А.А.Болотов, С.Б.Гашков, А.Б.Фролов,
А.А.Часовских

Алгоритмические основы эллиптической криптографии

Предполагается, что издание будет
допущено Министерством образования Российской Федерации
в качестве учебного пособия для студентов высших учебных заведений,
обучающихся по группе специальностей в области
информационной безопасности

Москва
2004

УДК 512.8
ББК
Б
А.А.Болотов, С.Б.Гашков, А.Б.Фролов, А.А.Часовских
Алгоритмические основы эллиптической криптографии. — М:Изд-во,2004. – 499 стр., ил.
ISSN
Книга содержит описание и анализ основных алгоритмов, на которых создаются криптографические системы эллиптической криптографии, алгебраическую теорию построения таких алгоритмов и анализ их сложности. Предназначена для студентов, преподавателей вузов и специалистов, создающих программные комплексы защиты информации на основе теории эллиптических кривых.

ББЛ

Учебное издание

БОЛОТОВ АНАТОЛИЙ АЛЕКСАНДРОВИЧ
ГАШКОВ СЕРГЕЙ БОРИСОВИЧ
ФРОЛОВ АЛЕКСАНДР БОРИСОВИЧ
ЧАСОВСКИХ АНАТОЛИЙ АЛЕКСАНДРОВИЧ

АЛГОРИТМИЧЕСКИЕ ОСНОВЫ ЭЛЛИПТИЧЕСКОЙ КРИПТОГРАФИИ

заведующий редакцией
Корректор
Компьютерная верстка

Издательство Изд-во РГСУ
Издательская лицензия
Адрес издательсва
Тел..факс:
Адрес в Интернет

Формат
Тираж
Отпечатано

©А.А.Болотов, С.Б.Гашков,
А.Б.Фролов, А.А.Часовских,2003

ISSN

Глава 0

Криптография в открытом обществе

До появления компьютеров как общедоступного средства вычислений, обработки и осмысливания больших объемов данных было одно понятие грамотности – овладение письменностью и искусством общения на основе познания культурного и научного наследия предшествующих поколений. Компьютеры вторгшиеся как в сферу профессиональной деятельности и в повседневные занятия человека неизбежно в силу естественного интереса к их использованию способствовали возникновению такого явления как в обществе как "вторая грамотность,"под которой и понимают владение компьютером и информационными технологиями. Расширение возможностей общения "вторично"образованных индивидуумов, обмена идеями, как и развитие электронной торговли сделало общество открытым. В открытом обществе "компьютерные"люди делятся на две категории – созидателей и расхитителей. Причем среди последних встречаются весьма грамотные специалисты, способные "расколоть"любую непрофессионально сделанную систему информационной защиты. Созидатели же подчас и не подозревают об опасностях, нависающих над их творениями или капиталами. Поэтому в открытом обществе одним из необходимых, даже определяющих условий использования информационных технологий в социальных системах и бизнесе, как и в других областях, является соблюдение условий и использование средств обеспечения информационной безопасности. При этом знакомство с криптографией потребуется каждому пользователю электронных средств обмена информацией. Поэтому криптография в будущем станет "третьей грамотностью"наряду со "второй грамотностью"Участники электронного информационного обмена должны владеть технологиями цифровой подписи, аутентификации и подтверждения целостности и подлинности электронных сообщений, обеспечения безопасности электронного бизнеса, защиты информации, передаваемой через Интернет и др. Актуальность этого подчеркивалась представительной научной конференцией "Московский университет и развитие криптографии в России,"МГУ, 17-18 октября 2002 г. Уважаемому читателю из числа

созидателей предлагается книга, содержащая фундаментальное изложение алгебраических и алгоритмических основ современного направления криптографии с открытым ключом (криптографии эллиптических кривых, позволяющей создавать наиболее защищенные и в то же время наиболее технологичные системы обеспечения информационной безопасности. Например, теория эллиптических кривых лежит в основе Российского стандарта электронной цифровой подписи. Отражая специфику криптографических применений, на самом деле, книга содержит алгоритмические основы конечной алгебры в широком понимании и содержит фундаментальный материал, способствующий более глубокому усвоению базовых алгебраических понятий и подходов к моделированию на их основе практических задач и в других областях, в частности, для безошибочных вычислений (error-free computing). Ее авторами являются весьма авторитетные ученые:

Болотов Анатолий Александрович, канд. физ.- мат. наук, доцент МЭИ, инициатор перевода и научный редактор одной из первых книг по криптографии с открытым ключом – А.Саломаа. Криптография с открытым ключом;

Гашков Сергей Борисович, доктор физ.- мат. наук, профессор кафедры дискретной математики МГУ, известный специалист по теории сложности вычислений;

Фролов Александр Борисович, доктор технических наук, профессор, профессор кафедры Информационной безопасности РГСУ (по совместительству), профессор кафедры Математического моделирования МЭИ.

Часовских Анатолий Александрович, доцент кафедры Математической теории интеллектуальных систем мех.-математа МГУ, известный специалист по теории сложности алгоритмов.

Значительная часть книги основана на публикациях авторов, а также на материалах лекций по дисциплинам "Математические основы криптологии"и "Криптографические методы обеспечения информационной безопасности,"прочитанных авторами в МГУ, РГСУ и МЭИ. Кроме того, в книге содержится систематизированное изложение современного состояния алгоритмических проблем этой области по многочисленным цитированным в работе публикациям. Существенно, что при специфике криптографической тематики авторами соблюдена преемственность с рядом изданных в последнее время учебников и научных изданий, ориентированных на криптографическое образование. За разработку на основе этого издания библиотеки программных функций ECCMSUMPEI авторы были отмечены в 2000 году дипломом компании INTEL в числе победителей конкурса по теме "Технологии электронного бизнеса"(наряду с участниками от МИФИ и МГТУ). Это свидетельствует о том, что полное издание будет хорошей теоретической основой для создания мощного лабораторного практикума, в частности, по технологиям электронного бизнеса.

Доктор физико-математических наук, профессор Бабаш А.В.

Глава 0

Предисловие

В последней четверти прошлого века на стыке теории сложности алгоритмов, алгоритмической теории чисел и компьютерной алгебры зародилась и в наши дни переживает настоящий бум по многим аспектам направление, известное сейчас как криптография с открытым ключом и позволяющее успешно решать разнообразные задачи защиты информации в компьютерных сетях [144]. В последние годы интенсивно развивается открытая независимо Н. Коблицем и Вю-Миллером в 1985 году криптография эллиптических кривых (мы предложили [11] краткий термин *эллиптическая криптография*), где роль основной криптографической функции выполняет скалярное умножение точки эллиптической кривой, то есть операция умножения точки на константу, аддитивный аналог возведения в степень в мультипликативной группе, реализуемый на основе операций сложения и удвоения точек эллиптической кривой. Последние, в свою очередь, описываются и выполняются на основе операций умножения. возведения в степень и инвертирования в в конечном поле, например $GF(p)$ или $GF(2^n)$.

Особый интерес к эллиптической криптографии обусловлен теми преимуществами, которые дает ее применение в беспроводных коммуникациях — высокое быстродействие и небольшая длина ключа. Например, в построенных на основе эллиптических кривых криптосистемах бинарной размерности от 150до 350 обеспечивается уровень криптографической стойкости, который требует в известных криптографических системах элементов бинарной размерности от 600 до 1400 и более.

Предлагаемое учебное пособие и посвящено этому современному направлению криптографии с открытым ключом — эллиптической криптографии.

В нем отражен опыт научного семинара, работающего в МЭИ под руководством авторов, а также опыт авторов преподавания математических основ криптологии и криптографических методов обеспечения информационной безопасности в Московском государственном университете, Московском энергетическом институте и Московском государственном социальном университете. При его подготовке использованы научные статьи авторов, многочисленные ис-

точники, приведенные в списке литературы, в частности, учебники [1, 41].

Наряду с фундаментальными алгебраическими аспектами эллиптических кривых особое внимание уделяется вопросам эффективной реализации базовых операций и основанных на на них криптографических протоколов с учетом особенностей и возможностей компьютера. По существу, речь идет о расширении его функциональных возможностей позволяющем эффективную реализацию операций в конечных полях и в группах точек эллиптических кривых. Это расширение допускает как чисто программные, так и технические, на основе синтеза логических схем, решения.

Мы используем все возможности достижения высокой скорости выполнения операций, как чисто программные, так и принципиально алгоритмические. Приемы первого рода мы называем программными «трюками», они позволяют повышать скорость выполнения операций «в разы», то есть уменьшают константу в оценке сложности операции. К таким «трюкам» относится, например, табулирование некоторых операций над байтами (умножение, возведение в квадрат, подсчет числа единиц, вычисление частного от деления на многочлен $1+x$, метод ускорения приведения по модулю неприводимого «малочлена» и другие). Заметим сразу, что мы не обращаемся к языкам типа Ассемблер, стремясь сохранить универсальность и сравнимость реализаций алгебраических преобразований. Приемы второго рода базируются на выдающихся научных открытиях в области алгоритмизации таких. как метод умножения Карацубы, позволивший существенно понизить сложность операции умножения как целых чисел, так и полиномов над конечным полем, получивший дальнейшее развитие в методе Шенхаге-Штрассена и других подходах к оптимизации операций в конечных алгебраических структурах. Именно в связи с ориентацией преимущественно на такого рода методы обеспечения эффективности в учебном пособии большое внимание уделяется оценкам сложности описываемых в нем алгоритмов. Мы исследуем преимущества имплементации алгебраических преобразований в тех или иных базисах конечного поля. В связи с этим неизбежно изучение тестов методов поиска неприводимых многочленов, допускающих нормальные, гауссовы и оптимальные нормальные базисы. Не ограничиваясь областью практической применимости, для демонстрации эффективности изучаемых методов мы работаем также и с алгебраическими структурами высоких порядков.

Следует отметить, что авторы не предлагают принципиально новых криптографических преобразований или криптосистем. В учебном пособии изучаются методы эффективной реализации базовых алгебраических и криптографических операций, на основе которых *могут* строиться как известные (соответствующие примеры приводятся в книге), так и перспективные криптографические системы (последние — с обязательным участием профессиональных криптографов).

Большинство изучаемых в книге понятий и алгоритмов иллюстрируется примерами. Наиболее существенные положения формулируются в виде теорем, наиболее сложные из них даются с доказательствами, а более очевидные фор-

мулируются в виде упражнений с указаниями к решению.

При подготовке примеров использована библиотека алгоритмов эллиптической криптографии и «визуализирующий» ее алгебраический процессор, разработанные под руководством авторов студентами МЭИ при выполнении дипломных проектов и магистерских диссертаций. Учебное пособие содержит все необходимое для создания таких библиотек и процессоров в любом вузе, где осуществляется подготовка по специальностям специалистов в области обеспечения информационной безопасности. Содержащиеся в учебном пособии теоремы, следствия, леммы, утверждения, уравнения (или алгебраические выражения), примеры, упражнения и рисунки нумеруются по главам, например, «Теорема 4.5» есть пятая теорема четвертой главы.

Алгебраическая и криптографическая терминология согласуется с учебниками [27, 1].

Учебное пособие состоит из шести глав.

Первая глава является алгебраическим введением, где рассматриваются основные алгебраические понятия и базовые алгоритмы, лежащие в основе теории, изучаемой в последующих главах и используемые в криптографической литературе.

Вторая глава посвящается современному состоянию теории тестирования и поиска неприводимых многочленов с заданными свойствами, в частности, допускающих построения нормального, гауссова или оптимального нормального базисов.

В третьей главе описываются методы реализации арифметических операций в конечных полях с использованием полиномиального (стандартного) базиса.

Четвертая глава содержит последние достижения в исследовании методов реализации арифметических операций в конечных полях с использованием нормальных базисов, в частности, гауссовых и оптимальных нормальных базисов, с акцентом на метода схемной реализации.

В пятой главе на основе даются основные алгоритмы на эллиптических кривых с особенностями их реализации в полях различных характеристик, а также методы эффективной реализации основной операции эллиптической криптографии – операции умножения точки эллиптической кривой на константу.

Шестая глава дает представление о реализации ряда известных криптографических протоколов (распределения ключей для классической криптосистемы, цифровой подписи и скрытой передачи) на основе изложенных и обоснованных в книге алгоритмов.

Авторы старались сочетать практическую и теоретическую направленность издания. Книга содержит материал и основанные на нем алгоритмы, систематическая реализация которых доступна каждому владеющему программированием на $C++$. Седьмая глава, по существу, и является путеводителем к созданию хорошей учебной лаборатории по математическим основам криптологии и криптографическим методам обеспечения информационной безопасности. Она написана с учетом положительного опыта создания Алгебраических процессо-

ров студентами и магистрантами МЭИ и МГУ.

В конце книги приведен список использованной учебной, монографической и научно-публицистической литературы, использованной авторами.

Для удобства пользования книгой приведен предметный указатель. Материалы, содержащие вспомогательные алгоритмы и таблицы вычисленных по описанным в книге алгоритмам параметрам алгебраических конструкций.

Учебное пособие предназначено, в первую очередь студентам,обучающимся по специальностям, а также для аспирантов по специальностям

«Криптография» – 075100б
«Компьютерная безопасность» – 075200б
«Организация и технология защиты информации» – 075300,
«Коплексная защита объектов информатизации» – 075400.

Авторы выражают благодарность за замечания всем специалистам, прочитавшим рукопись, а также студентам, также заметившим ряд неточностей, а главное, осуществившим экспериментальную проверку описаний алгоритмов и числовых параметров, приведенных в книге. Мы благодарны заведущему кафедрой информационной безопасности РГСУ профессору Бабашу А.В. за поддержку настоящего издания.

Глава 1

Алгебраические основы

1.1 Группы, кольца, поля

1.1.1 Группы

Группой называется множество G, на котором определена ассоциативная бинарная операция $\circ$,

которое содержит элемент e такой, что для любого элемента $a \in G$ выполняется

$$e \circ a = a \circ e = a,$$

и существует элемент $\bar{a}$ такой, что

$$a \circ \bar{a} = \bar{a} \circ a = e.$$

Указанный элемент e называется *единицей группы,* элемент $\bar{a}$ называется *симметричным* к элементу a. Легко показать, что единица группы единственна и что элемент, симметричный к данному элементу, также определяется однозначно. Как следствие в группе определена унарная операция $^-$ *инверсии,* которая сопоставляет каждому элементу a, $a \in G$, симметричный к нему элемент $\bar{a}$. Эта операция является биекцией. В разделе 1.1на стр. 9 В параграфе 1.1.1 на стр. 9 Если операция $\circ$ коммутативна, то группа называется *абелевой*. В дальнейшем мы будем изучать и использовать только абелевы группы.

Примерами абелевых групп являются

а) множества Z целых, Q рациональных, R действительных и C комплексных чисел с соответствующими операциями сложения;

б) множества $Q \setminus \{0\}$, $R \setminus \{0\}$ и $C \setminus \{0\}$ отличных от нуля рациональных, действительных и комплексных чисел с соответствующими операциями умножения.

Группы с операцией сложения называются *аддитивными,* группы с операцией умножения называются *мультипликативными.*

Бинарную операцию аддитивной группы принято обозначать знаком $+$, бинарную операцию мультипликативной группы обозначают знаком умножения $\times$, $\cdot$ или её обозначение, по умолчанию, опускают.

Единица аддитивной группы, как правило, обозначается 0 и называется *аддитивной* единицей. Результат операции аддитивной инверсии, то есть элемент $\bar{a}$ аддитивной группы, симметричный к элементу a, обозначатся $-a$ и называется *противоположным* к этому элементу. Единица мультипликативной группы обозначается 1 и называется *мультипликативной* единицей. Результат операции мультипликативной инверсии, то есть элемент $\bar{a}$ мультипликативной группы, симметричный к элементу a, обозначатся a^{-1} и называется *обратным* к этому элементу.

Ассоциативность операции $\circ$ позволяет записывать кратную композицию

$$(\cdots((a \circ a) \circ a) \circ \cdots a) \circ a,$$

опуская скобки:

$$a \circ a \circ a \circ \cdots \circ a.$$

Такие композиции называются *k-ой степенью* элемента a группы (k – число вхождений элемента a в формулу композиции). k-ая степень элемента a группы обозначается $k * a$ в аддитивных группах и a^k в мультипликативных группах.

Композиция

$$a + (-b)$$

операций инверсии и сложения аддитивной группы называется операцией *вычитания* элемента b из элемента a, обозначаемой $-$. Результат $a - b$ называется *разностью* элементов a и b. Аддитивно обратный относительно a элемент $-a$ получается как разность $0 - a$.

Композиция

$$a \times b^{-1}$$

операций инверсии и умножения мультипликативной группы называется операцией *деления* элемента a на элемент b, обозначаемой $/$. Результат a/b, или $\frac{a}{b}$ называется *частным* от деления элемента a на элемент b. Мультипликативно обратный относительно a элемент a^{-1} получается как частное $\frac{1}{a}$.

Рассмотренные выше примеры аддитивных и мультипликативных групп представляют бесконечные группы.

Группа на основе конечного множества G называется *конечной. Тривиальная* группа постоена на одноэлементном множестве $\{0\}$ (аддитивная тривиальная группа) или $\{1\}$ (мультипликативная тривиальная группа). Простейшие нетривиальные группы являются двухэлементными множествами с элементами, обозначаемыми обычно 0 и 1 (в аддитивной группе) или 1,2 (в мультипликативной группе).

Число элементов конечной группы называется *порядком* группы. Порядок тривиальной конечной группы равен 1, простейшая нетривиальная конечная группа имеет порядок 2.

Не трудно показать, что в конечной группе некоторая степень g^n любого элемента g группы равна единице e группы.

Порядком элемента g группы G называется наименьшее число n такое, что $g^n = e$, т.е. $g^n = 1$ в мультипликативной группе или $n * g = 0$ в аддитивной группе.

Элемент, порядок которого равен порядку группы, называется *образующим* элементом группы.

Группа, имеющая образующий элемент, называется *циклической*. В циклической группе каждый элемент представим как некоторая степень любого образующего элемента группы. Не трудно убедиться, что всякая циклическая группа абелева.

Часто конечная группа образуется как фактор-множество бесконечной группы по некоторому отношению эквивалентности. [1]

Так на множестве Z целых чисел относительно натурального числа m можно определить отношение

$$\{(x, y) \setminus x \equiv y \ (\mod m)\},$$

где $x \equiv y(\mod m)$ означает, что $m|(x - y)$, т.е. число m делит разность $(x - y)$. [2] Это отношение называется *отношением конгруэнтности по модулю* m, а классы эквивалентности по этому отношению – *классами конгруэнтности* (*или классами вычетов*) *по модулю* m.

Фактор-множество $Z/\equiv \mod m$ по этому отношению сокращенно обозначают Z_m, аналогично, классы конгруэнтности $[a]_{\equiv \mod m}$ обозначаются просто $[a]_m$.

Не трудно видеть, что $x \equiv y \ (\mod m)$ тогда и только тогда, когда $x \bmod m = y \bmod m$.

На фактор - множестве Z_m можно определить операцию сложения. Сумму классов эквивалентности определяют следующим образом:

$$[x]_m + [y]_m = [x + y]_m.$$

Удобно в качестве представителей классов $[x]_m$ использовать наименьшие неотрицательные элементы $x \bmod m$ классов. Тогда операцию сложения можно

[1] Отношением эквивалентности называется однородное бинарное отношение $\approx$, обладающее свойствами транзитивности, рефлексивности и симметричности. Множество, на котором задано это отношение, разбивается на классы эквивалентности $[a]_{\approx}$, где a – *представитель* класса. Совокупность классов эквивалентности есть *фактор-множество* множества A по данному отношению $\approx$, обозначается $A \setminus_{\approx}$.

[2] Согласно алгоритму деления целых чисел с остатком при заданном *делителе* d, $d \in Z\setminus\{0\}$ *делимое* a, $a \in Z$ единственным образом представляется формулой $a = qd + r$. $0 \le r < |d|$. Число q называется *частным*, а число r называется остатком от деления a на q. Например, если $a = 5$, а $d = -3$, то $a = (-2)(-2) + 1$, то есть $q = -2$, $r = 1$. Если остаток $r = 0$, то говорят, что d *делит* a, что обозначают $d|a$. Остаток обозначат также $r = a \bmod d$ или $rem(a, d)$ *Наибольшим общим делителем* целых чисел n и m называется наибольшее число d, являющееся делителем как m, так и n. Это число обозначается НОД(n, m) или просто (n, m). НОД(n, m) существует, тогда и только тогда, когда хотя бы одно из чисел n и m не равно 0. Если НОД$(n, m) = 1$, то числа n и m называются *взаимно простыми.*

описать в обозначениях этих представителей:

$$[x]_m + [y]_m = [(x + y) \bmod m]_m.$$

Не трудно убедиться, что фактор множество Z_m с только что описанной операцией сложения есть аддитивная группа. Аддитивной единицей является класс $[0]_m$, противоположный к этому элементу группы есть элемент $-[a]_m = [m - a]_m$.

Аналогично вводится операция умножения по модулю m.

$$[x]_m \times [y]_m = [x \times y]_m == [(x \times y) \bmod m]_m.$$

При этом множество ненулевых классов конгруэнтности $[a]_m,\ a \neq 0$, имеющих обратный класс $[a^{-1}]_m$, где $a \times a^{-1} \bmod m = 1$, образует мультипликативную группу, которая обозначается Z_m^*. Мультипликативной единицей является класс $[1]_m$. Класс $[a]_m$ принадлежит Z_m^* тогда и только тогда, когда числа a и m взаимно просты, то есть не имеют общих множителей. Эта группа содержит все ненулевые классы, то есть $Z_m^* = Z_m \setminus [0]_m$, тогда и только тогда, когда m есть простое число.

Замечание. Заметим, что рассмотренные аддитивная и мультипликативная группы, определённые на множествах Z_m и Z_m^* классов конгруэнтности по модулю m, изоморфны аддитивной и мультипликативной группам, заданным на множестве наименьших неотрицательных представителей этик классов, соответственно с операциями сложения и умножения по модулю m. Поэтому часто вместо группы на фактор множестве рассматривают группы на множестве представителей классов, при этом эти множества $\{0, 1, \ldots,\ m - 1\}$ и $\{a/a$ и m взаимно просты $\}$ представителей обозначают Z_m и Z_m^*, так же, как множества классов.

Подмножества $H,\ H \subseteq\ G$ множества группы G, замкнутое относительно операций группы, и являющееся группой с этими же операциями, называется *подгруппой* этой группы. Подгруппу удобно определить также следующим *критерием подгруппы:*

Подмножество $H,\ H \subset G$ *является подгруппой тогда и только тогда, когда* $\forall a, b \in H\ a \circ b^{-1} \in H$.

Например, тривиальные аддитивная или мультипликативная группы суть подгруппы любой аддитивной или мультипликативной группы соответственно, а простейшие нетривиальные аддитивная и мультипликативная группы (с точностью до обозначений элементов) являются подгруппами некоторых конечных нетривиальных аддитивных или мультипликативных групп соответственно.

Пусть H – подгруппа группы G. Тогда, как не трудно проверить, отношение $G_H = \{(a, b) \setminus \exists h \in H b = ah\}$ есть отношение эквивалентности на множестве G.

Классы эквивалентности $[a]_\approx$ по этому отношению называются *левыми смежными классами групп* G *по подгруппе* H. Очевидно, что $h_1 \neq h_2 \rightarrow ah_1 \neq ah_2$ для любых $h_1, h_2 \in H$. (Действительно, если $ah_1 = ah_2$, то умножая обе

части равенства слева на a^{-1}, получим $a^{-1}ah_1 = a^{-1}ah_2 \to h_1 = h_2$, противоречие). Значит, число элементов в каждом классе равно порядку подгруппы H. Если подгруппа H такова, что число смежных классов конечно, то это число называется *индексом подгруппы H в группе G* (обозначается $G : H$).

Учитывая, что суммарное число элементов в классах равно порядку группы, получаем для конечной группы следующее утверждение.

Теорема 1.1.1 (Лагранж) *Порядок и индекс подгруппы H в группе G делят порядок этой группы.*

Пример 1.1.1 Возьмем подгруппу $\{[0]_9, [3]_9, [6]_9\}$ аддитивной группы

$$Z_9 = \{[0]_9, [1]_9, [2]_9, [3]_9, [4]_9, [5]_9, [6]_9, [7]_9, [8]_9\}.$$

Левыми смежными классами являются множества

$$\{[0]_9, [3]_9, [6]_9\},$$
$$\{[1]_9, [4]_9, [7]_9\},$$
$$\{[2]_9, [5]_9, [8]_9\}.$$

Индекс $G : H$ равен 3.

Аналогично определяются правые смежные классы. Левые и правые смежные классы абелевой группы совпадают.

Далее не трудно заметить, что в конечной группе все различные степени

$$a^1, a^2, \ldots, a^\delta$$

(или, в аддитивной группе,

$$0 * a, 1 * a, \ldots, (\delta - 1) * a)$$

любого элемента составляют циклическую подгруппу, порядок которой равен порядку δ этого элемента (если $l > \delta$, то $a^l = a^{c\delta + (l \bmod \delta)} = a^{l \bmod \delta}$ или, в аддитивном случае, $l * a = (c\delta + (l \bmod \delta)) * a = (l \bmod \delta) * a$).

Отсюда получаем следующие утверждения:

Следствие 1.1.1 *Порядок любого элемента конечной группы делит порядок группы.*

Следствие 1.1.2 *Для любого элемента a конечной группы порядка m имеет место равенство $a^m = e$, т.е, $a^m = 1$ в мультипликативной группе и $m * a = 0$ в аддитивной группе.*

Следствие 1.1.3 *Если δ – порядок элемента a группы G, а $n \in N$, то*

$$a^n = e$$

тогда и только тогда, когда $\delta | n$.

Теорема 1.1.1 на стр. 13
1.1.3

Следствие 1.1.4 *Для любого элемента a конечной группы порядка m имеет место равенство $a^{m-1} = a^{-1}$ в мультипликативной группе и $(m-1) * a = -a$ в аддитивной группе.*

Следствие 1.1.5 *(Теорема Эйлера). Для всякого натурального n и всякого натурального a такого, что* НОД$(a, n) = 1$, *справедливо равенство*

$$a^{\varphi(n)} \bmod n) = 1.$$

Доказательство. Представители a классов конгруэнтности

$$[a]_{\equiv \bmod n} = \{x \setminus x \equiv a \pmod n\}$$

такие, что НОД $(a, n) = 1$, образуют группу с операцией умножения по модулю n. Порядок группы определяется функцией Эйлера $\varphi(n)$. (Напомним, что функция Эйлера $\varphi(n)$ как раз и определяет число указанных классов).

Пример 1.1.2 Если $n = 6$, то группа $G = (\{1, 5\}, \times_{\bmod 6}, 1)$, $\varphi(6) = 2$, $1^2 = 1 \bmod 6$, $5^2 = 1 \bmod 6$.

Следствие 1.1.6 *(Малая теорема Ферма). Для всякого простого числа p и натурального числа a, такого, что* НОД$(a, p) = 1$, *имеет место равенство*

$$a^{p-1} = 1 \bmod p.$$

По следствию 1.2, процесс возведения элемента конечной группы в большую степень можно упростить путём приведения показателя степени по модулю m порядка группы:

$$g^n = g^{n \bmod m}.$$

В частности, процесс возведения числа g, такого, что $(g, n) = 1$, в большую степень k по модулю n можно упростить путём приведения показателя степени по модулю $m = \varphi(n)$:

$$g^k \bmod n = g^{k \bmod \varphi(n)} \bmod n.$$

Теорема 1.1.2 *Если a – элемент порядка δ, то при любом $k \in N$*

$$\operatorname{ord} a^k = \frac{\delta}{(k, \delta)},$$

в частности,

$$\operatorname{ord} a^k = \delta \iff (k, \delta) = 1.$$

Доказательство. Обозначим $m = \text{ord } a^k$. Тогда $a^{km} = 1$. По следствию 1.3

$$\delta | km.$$

Отсюда

$$\frac{\delta}{(k,\delta)} \left| \frac{k}{(k,\delta)} \cdot m. \right.$$

Учитывая. что

$$\left(\frac{\delta}{(k,\delta)}, \frac{k}{(k,\delta)} \right) = 1,$$

получаем

$$\frac{\delta}{(k,\delta)} \left| m. \right. \tag{1.1}$$

С другой стороны,

$$(a^k)^{\frac{\delta}{(k,\delta)}} = (a^\delta)^{\frac{k}{(k,\delta)}} = 1.$$

Поэтому с учётом следствия 1.3

$$m \left| \frac{\delta}{(k,\delta)}, \right. \tag{1.2}$$

Объединяя отношения 1.1 и 1.2, получаем

$$m = \text{ord } a^k = \frac{\delta}{(k,\delta)}.$$

Рассмотрим методы вычисления порядка элемента группы и нахождения образующих элементов групп, а также элементов высокого порядка.

В приведенных ниже алгоритмах используется свойство, что порядок элемента делит порядок группы (Следствие 1.1). Алгоритмы записаны применительно к мультипликативным группам. Они применимы и к аддитивным группам, если в их описаниях мультипликативные степени элементов группы заменить аддитивными кратными.

Полезно также заметить, что возведение в степень элемента группы можно осуществить быстро, если воспользоваться разложением показателя степени по степеням основания системы счисления, например 2 и использовать следующий алгоритм.

Если вместо возведения в степень использовать операцию умножения, то получим алгоритм вычисления аддитивного кратного $k * a$ элемента a

Определение порядка элемента группы при известной факторизации порядка n группы. Порядок элемента группы можно определить по Алгоритму 1.2.

Поиск образующего элемента циклической группы. Приведём вероятностный Алгоритм 1.3 поиска образующего элемента циклической группы.

Алгоритм 1.1.1

ВХОД: Элемент a мультипликативной группы G;
коэффициенты $(d_0, d_1, \dots d_{n-1})$ бинарного разложения показателя степени $d = d_0 2^0 + d_1 2^1 + \cdots + d_{n-1} 2^{n-1}$.
ВЫХОД: Степень $b = a^d$ элемента a.

1. Присвоить $b \leftarrow 1$.
2. Для i от 1 до n выполнять:
   $b = (b(a^{d_{i-1}}))^2$.
3. Вернуть b.

Рис. 1.1: @@.

Эффективность алгоритма определяется тем, что группа содержит $\varphi(n)$ образующих элементов, и вероятность того, что случайно выбираемый элемент является образующим равна $\varphi(n)/n > \frac{1}{6 \ln \ln n}$.

Замечание. Трудности проблемы факторизации можно обойти выбором подходящей группы Z_p^*. При этом обеспечивается и присутствие большого множителя в разложении числа $p-1$: сначала выбирается достаточно большое простое число q. Затем случайно выбирают относительно малые числа R, пока не будет получено простое число $p = 2Rq + 1$. Поскольку $p - 1 = 2Rq$, факторизация сводится к факторизации числа R. Если выбирать $R = 1$, то факторизацией $p - 1$ является просто $2q$. Поскольку $\varphi(p-1) = \varphi(2q) = \varphi(2)\varphi(q) = q - 1$, вероятность того, что случайно выбранный элемент $\alpha \in Z_p^*$ является образующим элементом, есть $\frac{q-1}{2q} \approx \frac{1}{2}$. *Безопасным проствм числом* называется простое число вида $p = 2q + 1$, где q – простое.

Поиск элемента высокого порядка циклической группы. Иногда требуется элементы высокого порядка, не являющиеся образующими элементами.

Пусть α – образующий элемент циклической группы G порядка n и d делитель числа n. Тогда по теореме 1.7 элемент β порядка d можно получить как $\beta = \alpha^{n/d}$. Если q – простой делитель порядка n циклической группы G, элемент β порядка q можно найти без предварительного поиска образующего элемента α группы G. Для этого выбирают случайно $g \in G$ и вычисляют $\beta = g^{n/q}$, повторяя эти действия, пока $\beta \neq 1$.

Алгоритм 1.1.2

ВХОД: Мультипликативная группа G порядка n, элемент $a \in G$,
факторизация $n = p_1^{e_1} \cdot p_2^{e_2} \cdots p_k^{e_k}$,
показателя степени $d = d_0 2^0 + d_1 2^1 + \cdots + d_{n-1} 2^{n-1}$.
где $p_i,\ i = 1, \ldots, k$, – разные простые числа.
ВЫХОД: порядок t элемента a.
1. Присвоить $t \leftarrow n$.
2. Для i от 1 до k выполнять:
   2.1 Присвоить $t \leftarrow t/p_i^{e_i}$.
   2.2 Вычислить $a_1 \leftarrow a^t$.
   2.3 Пока $a_1 \neq 1$ выполнять $a_1 \leftarrow a_1^{p_i}$ и присваивать $t \leftarrow t \cdot p_i$.
3. Вернуть t.

Рис. 1.2: @@.

Алгоритм 1.1.3

ВХОД: Мультипликативная циклическая группа G порядка n,
факторизация $n = p_1^{e_1} \cdot p_2^{e_2} \cdots p_k^{e_k}$,
показателя степени $d = d_0 2^0 + d_1 2^1 + \cdots + d_{n-1} 2^{n-1}$.
где $p_i,\ i = 1, \ldots, k$, – разные простые числа.
ВЫХОД: образующий элемент α группы G.
1. Выбрать случайный элемент α группы G.
2. Для i от 1 до k выполнять:
   2.1 Вычислить $b \leftarrow \alpha^{n/p_i}$.
   2.2 Если $b = 1$, то перейти к 1.
3. Вернуть α.

Рис. 1.3: @@.

1.1.2 Кольца. Решетки на кольце

Кольцом называется множество R с двумя бинарными операциями $+$ и $\times$ такими, что R является абелевой группой относительно сложения и операция $\times$ ассоциативна и дистрибутивна относительно операции $+$:

$$(a \times b) \times c = a \times (b \times c);$$

$$a \times (b + c) = a \times b + a \times c \text{ и } (b + c) \times a = b \times a + c \times a.$$

Следствием определения кольца является свойство:

$$\forall a \; a \times 0 = 0 \times a = 0.$$

Примерами являются множества Z целых, Q рациональных и R действительных чисел с операциями сложения и умножения.

Кольцо, в котором $a \times b = 0 \rightarrow a = 0$ или $b = 0$ называется *областью целостности*. Если в кольце имеется мультипликативная единица 1, то кольцо называется *кольцом с единицей*. Ниже изучаются только области целостности с единицей.

Элемент a' кольца с единицей такой, что $a \times a' = 1$ называется *обратным к элементу* a. В общем случае ненулевой элемент кольца имеет не более одного обратного к нему элемента. Обратный к ненулевому элементу a кольца элемент обозначается a^{-1}.

На множестве элементов кольца определятся отношение $|$ делимости: $a|b$, если в кольце найдется элемент q такой, что $b = q \times a$, в частности, $a|b$, если ненулевой элемент a имеет обратный к нему элемент a^{-1} : в этом случае $b = q \times a$, где $q = b \times a^{-1}$. В кольце $\forall a \; a|0$ и $0|b \rightarrow b = 0$, в частности $0|0$. В кольце с единицей $\forall a \; 1|a$. Нулевой элемент кольца обратного к нему элемента не имеет.

Если $a \neq 0$, то в случае $a|b$ упомянутый элемент $q, b = q \times a$, единственный. Он называется *частным* от деления элемента b на элемент a. Если же $a = 0$, то каждый элемент кольца может выступать в роли такого элемента q, и частное от деления элемента b на элемент 0 не определено.

Не трудно показать, что это отношение является отношением предпорядка, то есть оно транзитивно и рефлексивно, но в общем случае не антисимметрично. Можно определить отношение эквивалентности на множестве элементов кольца $a \approx b \iff a|b$ и $b|a$.

В обозначениях $[a]_{\approx}$ классов эквивалентности по этому отношению в качестве представителя a класса указывают элемент, выбираемый по известному правилу, например, наибольший элемент этого класса по тому или иному, но определённому отношению линейного порядка на множестве R. Например, если R есть кольцо Z целых чисел, то классы содержат один или два элемента, из которых в качестве представителя в обозначении класса выбирается неотрицательный элемент.

Наряду с элементом a класс $[a]_{\approx} \in Z_{\approx}$ содержит противоположный по отношению к нему элемент $-a$. Для их различения будем использовать записи вида

$$(< \text{знак}>, < \text{элемент, представляющий класс}>).$$

Так элемент a, представляющий класс будем записывать в виде пары $(0, a)$, а противоположный к нему элемент $-a \in [a]_{\approx}$ представлять парой $(1, a)$.

На множестве классов эквивалентности $[a]_{\approx}$, определяется отношение частичного порядка $\leq$:

$$[a]_{\approx} \leq [b]_{\approx} =_{def} a|b.@[def?]^3$$

При этом классы $[1]_{\approx}$ и $[0]_{\approx}$ являются соответственно нижней и верхней универсальной границей получающегося при этом частично упорядоченного множества $(R_{\approx}, \leq)$.

Фрагмент диаграммы Хассе для такого частичного упорядочивания классов $Z_{\approx}$ дан на Рис.1.

Если фактор множество $R_{\approx}$ счетно или конечно, то относительно отношения $\leq$ его можно упорядочить топологически, то есть линейно и согласованно с этим отношением:

$$\forall [a]_{\approx}, [b]_{\approx} : \ [a]_{\approx} \leq [b]_{\approx} \rightarrow [a]_{\approx} \preceq [b]_{\approx},$$

где $\preceq$ обозначает отношение топологического порядка.

Пример 1.1.3 Топологическое упорядочение $Z/_{\approx}$:

$$[1]_{\approx} \preceq [2]_{\approx} \preceq [3]_{\approx} \preceq [4]_{\approx} \preceq [5]_{\approx} \preceq \ldots \preceq [0]_{\approx}$$

т.е.

$$\forall i, \ i > 0 \ [i]_{\approx} \preceq [i+1]_{\approx} \preceq [0]_{\approx}.$$

---

[3] Отношением частичного порядка называется однородное бинарное отношение, обладающее свойствами транзитивности, рефлексивности и антисимметричности. Множество, на котором задано такое отношение, называется частично упорядоченным множеством (ЧУМ). Элемент m, такой, что для всякого элемента a, некоторого подмножества A ЧУМ $a \leq m$ ($m \leq a$), называется *мажорантой* (*минорантой*) множества A. Совокупность всех мажорант (миноранта) составляет *верхний* (*нижний*) конус множества A. Мажоранта (миноранта) m множества A, принадлежащая этому множеству, называется его максимумом (минимумом). Минимум (максимум) верхнего (нижнего) конуса множества A называется *супремумом* (*инфимумом*) этого множества. Супремум (инфимум) одноэлементного подмножества $\{a\}$ есть элемент a. Элемент b ЧУМ *доминирует* над элементом a, если $a \leq b$ и нет другого элемента c, такого,что $a \leq c \leq b$. Графически ЧУМ удобно задавать *диаграммой Хассе,* графом, вершины которого соответствуют элементам ЧУМ, а ребра соответствуют отношению доминирования. Элемент $a, \ a \leq b$, размещается не выше элемента b.

Если супремум и инфимум имеет каждое двухэлементное подмножество ЧУМ, то ЧУМ называется решеткой. Мажоронта (миноранта) всего ЧУМ называется его *верхней* (*нижней*) *универсальной границей.* Всякая конечная решетка имеет как нижнюю O, так и верхнюю I универсальную границу. Элементы, домнирующие над O, называются *атомами* решетки.

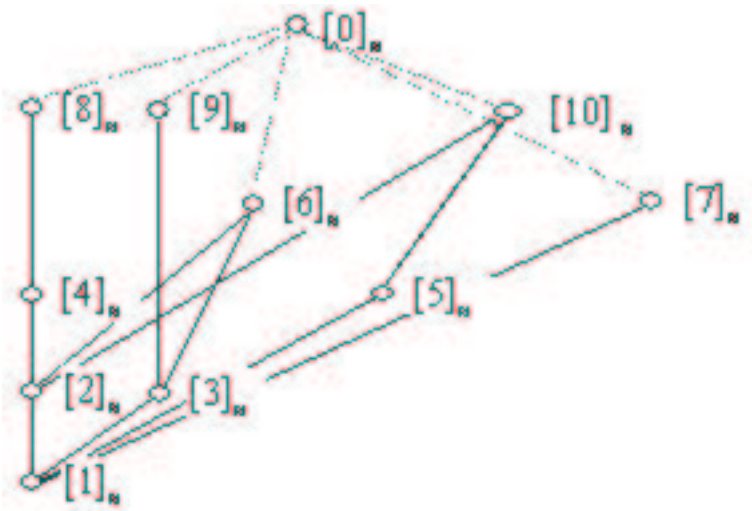

Рис. 1.4: Решетка.

(См. @Рис.1 по ярусам снизу вверх.)

Заметим, что множество $R/_{\approx}$ с отношением $\leq$ частичного порядка является решеткой. Операции инфимума и супремума на этой решетке будем обозначать $\wedge$ и $\vee$.

Наибольшим общим делителем НОД(x, y) элементов x и y кольца R называется элемент a, представляющий класс $[a]_{\approx} = [x']_{\approx} \wedge [y']_{\approx} \neq [0]_{\approx}$, $x \in [x']_{\approx}$, $y \in [y']_{\approx}$.

Замечание. В соответствии с этим определением НОД(0,0) не существует.

Пример 1.1.4 В кольце Z НОД$(-6,-4) = 2$: $-6 \in [6]_{\approx}$, $-4 \in [4]_{\approx}$, НОД$(-6,-4)$=$[6]_{\approx} \wedge [4]_{\approx} = [2]_{\approx}$.

Наименьшим общим кратным НОК(x, y) элементов x и y кольца R называется представитель a класса $[a]_{\approx} = [x']_{\approx} \vee [y']_{\approx} \neq [0]_{\approx}$, $x \in [x']_{\approx}$, $y \in [y']_{\approx}$.

В соответствии с этим определением, если хотя бы один из элементов x или y равен 0, то НОК(x, y) не существует. Легко установить, что

НОК$(a, b) = \frac{ab}{(a,b)}$, где (a, b)= НОД(a, b).

Пример 1.1.5 В кольце Z НОК$(-6,-4)$=12, так как $[6]_{\approx} \vee [4]_{\approx} = [12]_{\approx}$.

Элементы решетки $R_{\approx}$, доминирующие над классом $[1]_{\approx}$, являются ее *атомами,* и всякий ее элемент $[a]_{\approx}$ является супремумом степеней некоторых атомов. Соответственно элемент a, представляющий класс $[a]_{\approx}$ является произведением степеней элементов кольца R, представляющих эти атомы. На языке теории частично упорядоченных множеств эти атомы суть все атомы, являющиеся минорантами множества $\{[a]_{\approx}\}$, или атомы, принадлежащие нижнему конусу этого множества. Задача нахождения этих элементов и их степеней,

произведение которых равно элементу a, называется задачей факторизации в кольце R. Факторизация в кольце целых чисел до настоящего времени является задачей высокой вычислительной сложности. Некоторые рассматриваемые нами алгоритмы построены в предположении, что факторизация одного или более операндов известна. Одним из принципов криптографии с открытым ключом является использование известной факторизации при зашифровании в предположении, что факторизация сохраняется в тайне.

Как видно, элементы, представляющие классы эквивалентности $[a]_{\approx}$, составляют подмножество кольца R, являющееся решеткой с операциями $\inf(x,y)$, $\sup(x,y)$. Далее это подмножество обозначается $R_{\approx}$, как и множество классов эквивалентности. Имеется в виду, что $\inf(a,b) = a \wedge b$ есть элемент, представляющий класс $[a]_{\approx} \wedge [b]_{\approx}$, а $\sup(a,b) = a \vee b$ есть элемент, представляющий класс $[a]_{\approx} \vee [b]_{\approx}$.

Рассмотрим отношение эквивалентности $\equiv \pmod{\mathrm{m}}$ на множестве $R_{\approx}$, которое называется отношением *конгруэнтности по модулю элемента* $m \in R_{\approx}$:

$$a \equiv b \pmod{m} =_{def} m|(a-b),$$

где разность вычисляется применительно к элементам кольца R. При $m = 0$ оно является отношением равенства. Классы эквивалентности $[a]_{\equiv \bmod m}$ называются *классами конгруэнтности по модулю* m.

Если $c \in [a]_{\equiv \bmod m}$, то это обозначается как

$$a = c \bmod m$$

или $c = \mathrm{rem}(a,m)$. (При $m = 0$ $c \in [a]_{\approx \, mod \, m}$ равносильно $c = a$).

Заметим, что всегда

$$a|(b - b \bmod a)$$

и при $a \neq 0$ определено частное $q = \frac{b - b \bmod a}{a}$.

Представление элемента b кольца относительно элемента $a, a \neq 0$, в виде

$$b = q \times a + b \bmod a = q \times a + \ \mathrm{rem}(b,a)$$

называют *алгоритмом* или *операцией деления с остатком.*

Пример 1.1.6 В решетке $Z_{\approx}$ целых чисел $a = c \bmod m, m \neq 0$, есть остаток $r = \mathrm{rem}(c,m)$ от деления числа c на m, если же $m = 0$, то $\forall c \in Z_{\approx}$ $c \bmod m = c$. Так $\mathrm{rem}(6,5) = 1$, rem $(6,0) = 6$.

Используя эту операцию, можно описать алгоритм Евклида для операции инфимума на решетке $R_{\approx}$.

Обратим внимание, что $0 \wedge 0 = 0$, что и получается по этому алгоритму.

Утверждение 1.1. *Алгоритм* 1.4 *вычисляет представителя класса* $[d]_{\approx}$, $d = a \wedge b$.

Алгоритм 1.1.4

ВХОД: Представители a и b классов $[a]_{\approx}$ и $[b]_{\approx}$,
ВЫХОД: представитель d класса $[d]_{\approx}$, $d = a \wedge b$
1. Если $\mathrm{rem}(a, b) = a$, то присвоить $c \leftarrow a, a \leftarrow b, b \leftarrow c$.
2. Пока $b \neq 0$
   2.1. Присвоить $r \leftarrow \mathrm{rem}(a, b)$,
   2.2. Присвоить $a \leftarrow b$, $b \leftarrow r$.
3. Присвоить $d \leftarrow a$ и возвратить d

Рис. 1.5: @@.

Доказательство. Пусть п. 1 алгоритма уже выполнен. Значения переменных a, b и r по окончании i-ой итерации п.2 будем обозначать a^i, b^i и r^i соответственно. Начальные значения a и b обозначим a^0, b^0. Утверждение будет доказано, если мы покажем, что $\forall i\ a^i \wedge b^i = a^0 \wedge b^0$.

Докажем это индукцией по числу итераций п.2 алгоритма.

Базис индукции очевиден:

a) если $b^0 = 0$, то $r^1 = \mathrm{rem}(a^0, b^0) = a^0$, и $a^1 \wedge b^1 = b^0 \wedge r^1 = b^0 \wedge a^0 = a^0 \wedge b^0$;

b) если $b^0 \neq 0$, то

$$a^1 \wedge b^1 = b^0 \wedge r^1 =$$

$$= b^0 \wedge \frac{a^0 - r^1}{b^0} b^0 + r^1 =$$

$$= b^0 \wedge a^0 = a^0 \wedge b^0.$$

Если предположим, что

$$a^i \wedge b^i = a^0 \wedge b^0,$$

то тем же приемом получим доказательство индуктивного шага (учитывая, что $b^i \neq 0$) :

$$a^{i+1} \wedge b^{i+1} = b^i \wedge r^{i+1} =$$

$$= b^i \wedge \frac{a^i - r^{i+1}}{b^i} b^i + r^{i+1} =$$

$$= b^i \wedge a^i = a^i \wedge b^i = a^0 \wedge b^0.$$

Пример 1.1.7 Применим Алгоритм 1.4 к классам $[115]_{\approx}$ и $[25]_{\approx}$. Вычисления представлены в следующей таблице, где столбцы соответствуют последовательным шагам алгоритма.

Обозначения переменных	1	2.1^1	2.2^1	2.1^2	2.2^2	2.1^3	2.2^3	2.1^4	2.2^4	3
a	115		25		15		10		5	
b	25		15		10		5		0	
r		15		10		5		0		
d										5

Заметим, что для любых не равных 0 одновременно элементов a, b из $R_{\approx}$ существует единственная пара (x, y) элементов этого множества такая, что

$$a \times x + b \times y = a \wedge b. \tag{1.3}$$

Приведенный ниже расширенный алгоритм Евклида вычисляет $a \wedge b$ и элементы x и y, удовлетворяющие равенству (1.3)

Будем использовать также операцию деления $\mathrm{div}(a, b) = a/b$ в случае, когда выполняется отношение $b|a$ и $b \neq 0$.

В алгоритме участвуют как элементы $a = (0, a)$, представленные в обозначениях классов $[a]_{\approx}$, так и противоположные к ним элементы $-a = (1, a)$.

Заметим, что элементы a, представляющие классы $[a]_{\approx}$ в совокупности с противоположными к ним элементами образуют множество, являющееся подкольцом кольца R.

Операция сложения $(\mathrm{sign}(x), x) + (\mathrm{sign}(y), y) = (\mathrm{sign}(x + y), (x + y))$ этого подкольца может быть описана следующей таблицей.

$\mathrm{sign}(x)$	$\mathrm{sign}(y)$	$x > y$	$x = y$	$x < y$	$x + y$	$\mathrm{sign}(x + y)$
0	0	$\pm$	$\pm$	$\pm$	$x + y$	0
0	1	$+$	$-$	$-$	$x - y$	0
0	1	$-$	$+$	$-$	0	0
0	1	$-$	$-$	$+$	$y - x$	1
1	0	$+$	$-$	$-$	$x - y$	1
1	0	$-$	$+$	$-$	0	0
1	0	$-$	$-$	$+$	$y - x$	0
1	1	$\pm$	$\pm$	$\pm$	$x + y$	1

Операция вычитания описывается как композиция операции взятия противоположного элемента и сложения: $(\mathrm{sign}(x), x) - (\mathrm{sign}(y), y) = (\mathrm{sign}(x), x) + (\mathrm{sign}(y) \oplus 1, y))$.

Операция умножения умножения определяется выражением

$$(\mathrm{sign}(x), x) \times (\mathrm{sign}(y), y) = (\mathrm{sign}(x) \oplus \mathrm{sign}(y), xy).$$

Утверждение 1.2. *Алгоритм* 1.5 *вычисляет* $a \wedge b$ *и элементы* x *и* y, *удовлетворяющие равенству* (1.3).

Алгоритм 1.1.5

ВХОД: Представители a и b классов $[a]_{\approx}$ и $[b]_{\approx}$.
ВЫХОД: представитель d класса $[d]_{\approx}$, $d = a \wedge b$,
элементы x и y такие, что $ax + by = d$.

1. Если $\mathrm{rem}(a, b) = a$, то присвоить $c \leftarrow a, a \leftarrow b, b \leftarrow c$.
2. Присвоить $x_2 \leftarrow (0, 1)$, $x_1 \leftarrow (0, 0)$, $y_2 \leftarrow (0, 0)$, $y_1 \leftarrow (0, 1)$.
3. Пока $b \neq 0$
   3.1 $r \leftarrow \mathrm{rem}(a, b)$, $q \leftarrow (a - r)/b$,
   $x = x_2 - q \times x_1$,
   $y = y_2 - q \times y_1$.
   3.2 Присвоить $a \leftarrow b$, $b \leftarrow r$,
   $x_2 \leftarrow x_1$, $x_1 \leftarrow x$,
   $y_2 \leftarrow y_1$, $y_1 \leftarrow y$.
4. Присвоить $x \leftarrow x_2$, $y \leftarrow y_2$, $d \leftarrow a$ и возвратить (d, x, y).

Рис. 1.6: @@.

Доказательство. Заметим, что переменные a^i, b^i и r^i в каждой итерации вычисляются в соответствии с алгоритмом 1.3, причём $a^0 = a, b^0 = b$. Таким образом $\forall i\ (a^i \wedge b^i) = (a^0 \wedge b^0) = (a \wedge b)$. Осталось доказать, что значения x_2 и y_2 по окончании последней итерации п.3.2 таковы, что $a^0 x_2 + b^0 y_2 = (a^0 \wedge b^0)$. Для этого индукцией по числу итераций покажем, что

$$\forall i\ a^0 x_2^i + b^0 y_2^i = a^i.$$

Базис индукции очевиден:

$$a^0 x_2^1 + b^0 y_2^1 = a^0 x_1^0 + b^0 y_1^0 = a^0 \times 0 + b^0 \times 1 = b^0 = a^1.$$

Допустим, что

$$a^0 x_2^i + b^0 y_2^i = a^i.$$

Тогда

$$a^0 x_2^{i+1} + b^0 y_2^{i+1} =$$
$$= a^0 x_1^i + b^0 y_1^i$$
$$= a^0 (x_2^{i-1} - q^{i-1} x_1^{i-1}) + b^0 (y_2^{i-1} - q^{i-1} y_1^{i-1}) =$$
$$= (a^0 x_2^{i-1} + b^0 (y_2^{i-1}) - q^{i-1} (a^0 x_1^{i-1} + b^0 y_1^{i-1}) =$$
$$= a^{i-1} - q^{i-1} (a^0 x_2^i + b^0 y_2^i) = a^{i-1} - q^{i-1} a^i =$$
$$= r^{i-1} = a^{i+1}.$$

Алгоритм 1.1.6

ВХОД: Элементы a и b кольца R.
ВЫХОД: Элемент d =НОД(a,b) и элементы x, y такие,
что $ax + by = d$ или сообщение, что НОД(a,b) не существует
1. Применить Алгоритм 1.5.
2. Если $d \neq 0$, то возвратить d, x, y,
иначе возвратить "НОД(a,b) не существует".

Рис. 1.7: @@.

Пример 1.1.8 Вычислим элемент $d = 6 \wedge 5$, представляющий класс $[d]_{\approx} = [6]_{\approx} \wedge [5]_{\approx}$. Для этого применим Алгоритм 1.5 к паре классов $[6]_{\approx}$ и $[5]_{\approx}$. Вычисления представлены в таблице, где в столбцах 1, 2, 3.1^1, 3.2^1, 3.1^2 3.2^2 и 4 приведены результаты исполнения алгоритма по шагам. Напомним, что элементы a, b и d представляют классы $[a]_{\approx}$, $[b]_{\approx}$ и $[d]_{\approx}$ соответственно.

Обозначения переменных	1	2	3.1^1	3.2^1	3.1^2	3.2^2	4
a	$(0,6)$			$(0,5)$		$(0,1)$	
b	$(0,5)$			$(0,1)$		$(0,0)$	
r			$(0,1)$		$(0,0)$		
q	1		1		5		
x_2		$(0,1)$		$(0,0)$		$(0,1)$	
x_1		$(0,0)$		$(0,1)$		$(1,5)$	
x			$(0,1)$		$(1,5)$		$(0,1)$
y_2		$(0,0)$		$(0,1)$		$(1,1)$	
y_1		$(0,1)$		$(1,1)$		$(0,6)$	
y			$(1,1)$		$(0,6)$		$(1,1)$
d							$(0,1)$

Получили, $d = 1$, то есть $[6]_{\approx} \wedge [5]_{\approx} = [1]_{\approx}$. При этом $x \times a + y \times b = (0,1) \times (0,6) + (1,1) \times (0,5) = (0,1)$.

Применив алгоритм к паре $[0]_{\approx}, [0]_{\approx}$, получим $d = 0 \wedge 0 = 0$ – элемент, представляющий класс $[0]_{\approx}$

Алгоритмы 1.4 и 1.5 можно использовать в алгоритмах вычисления НОД двух элементов кольца R :

Аналогично получается алгоритм вычисления наибольшего общего делителя с использованием Алгоритма 1.3.

На фактор множестве $R/\equiv \bmod m$, используя операции кольца R, можно определить операции сложения $(+_{\equiv \bmod m})$ и умножения $(\times_{\equiv \bmod m})$ по модулю

m (более точно, эти операции определяются операциями сложения и умножения представителей классов по модулю m):

$$[a]_{\equiv \bmod m} +_{\bmod m} [b]_{\equiv \bmod m} = [a+b]_{\equiv \bmod m};$$

$$[a]_{\equiv \bmod m} \times_{\bmod m} [b]_{\equiv \bmod m} = [a \times b]_{\equiv \bmod m}.$$

Возможность такого определения операций и позволила назвать отношение эквивалентности $\equiv \bmod m$ отношением конгруэнтности.

Не трудно показать, что фактор множество $R_m = R/\equiv \bmod m$ с такими операциями также является кольцом, которое называется *фактор кольцом по отношению* $\equiv \bmod m$ (сокращенно – фактор кольцом по модулю m) Элементы $[a]_{\equiv \bmod m} \in R_m$ будем кратко обозначать $[a]_m$ Его нулевым элементом является класс $[0]_m$ мультипликативной единицей является класс $[1]_m$

Элемент $[a]_m^{-1} = [a']_m$ называется обратным элементом по отношению к элементу $[a]_m$, если $[a']_m \times [a]_m = [1]_m$.

Определить для элемента $[a]_m$ обратный к нему элемент или установить, что обратный элемент не существует, можно с помощью рассмотренного выше расширенного алгоритма Евклида, применив его к паре элементов $[m]_\approx$ и $[a]_\approx$ решетки $R_\approx$. Если по этому алгоритму получается $d = [1]_\approx$, то есть $[m]_\approx \wedge [a]_\approx = [1]_\approx$, то согласно выражениям 1.1 и 1.2 обратным к элементу $[a]_m$ является класс $[a]_m^{-1} = [x]_m$, где x – элемент, представляющий класс $[x]_\approx$, полученный по расширенному алгоритму Евклида. Если же $d \neq [1]_\approx$, то обратный элемент не существует.

Пример 1.1.9 Элемент $[5]_6$ кольца Z_6 является "самообратным,"то есть $[5']_6 = [5]_6$. Действительно, в примере 1.8 получили, что $d = 1$, следовательно элемент $y = -1 \bmod m$ является представителем класса $[5']_m$, $5' = -1 + 6$.

Применяя этот же алгоритм к классам $[6]_\approx$ и $[4]_\approx$, получим $d = 2$, что свидетельствует об отсутствии элемента, обратного к $[4]_\approx$.

Ненулевые элементы $[a]_m$ кольца R_m, имеющие обратные $[a]^{-1}$ к ним элементы (такие, что $[a]_m \wedge [a]^{-1} = [1]_m$), образуют мультипликативную группу R_m^*.

Последняя совпадает с множеством ненулевых элементов (т.е. каждый ненулевой элемент кольца R_m имеет обратный к нему элемент) тогда и только тогда, когда m не имеет делителей, отличных от 1 (т.е. $[a]_\approx \leq [m]_\approx \rightarrow 1 \in [a]_\approx$).

Далее классы $[a]_m$ для краткости мы обозначаем просто их представителями $\mathbf{a}$ жирным шрифтом, если значение m ясно из текста.

Пример 1.1.10 Множество $\{\mathbf{0}, \mathbf{1}, \mathbf{2}, \mathbf{3}, \mathbf{4}\}$ с операциями сложения и умножения по модулю 5 является кольцом, множество $\{\mathbf{1}, \mathbf{2}, \mathbf{3}, \mathbf{4}\}$ ненулевых элементов которого с операцией умножения по модулю 5 образует мультипликативную группу.

Множество $\{\mathbf{0}, \mathbf{1}, \mathbf{2}, \mathbf{3}, \mathbf{4}, \mathbf{5}\}$ с операциями сложения и умножения по модулю 6 является кольцом, но множество $\{\mathbf{1}, \mathbf{2}, \mathbf{3}, \mathbf{4}, \mathbf{5}\}$ его ненулевых элементов не является мультипликативной группой: элемент $\mathbf{2} \times \mathbf{3} \bmod m = \mathbf{0}$.

Множество элементов, представляющих классы кольца R_m, также является кольцом с операциями сложения и умножения по модулю m.

Алгоритм 1.1.7

ВХОД: два различных нечетных простых числа p и q,
факторизация чисел $p-1$ и $q-1$.
ВЫХОД: элемент α максимального порядка НОК$(p-1, q-1)$ группы $Z_n^*,\ n = p \cdot q$.
1. Применяя алгоритм 1 к $G = Z_p^*$ и факторизации числа $p-1$ найти образующий элемент a группы G_p^*.
2. Применяя алгоритм 2 к $G = Z_q^*$ и факторизации числа $q-1$ найти образующий элемент b группы G_q^*.
3. По китайской теореме об остатках и алгоритму Гаусса найти целое $\alpha,\ 1 \leq \alpha \leq n-1$, удовлетворяющее сравнениям
$\alpha \equiv a \pmod{p}$,
$\alpha \equiv b \pmod{p}$.
4. Вернуть α.

Рис. 1.8: @@.

В общем случае множество всех элементов кольца R_m, имеющих обратные к ним элементы, с этим кольцом может не совпадать, но как его подмножество оно является мультипликативной группой и обозначается R_m^*. Например, $Z_6^* = \{\mathbf{1}, \mathbf{5}\}$. Не все такие мультипликативные подгруппы являются циклическими, но они применяются в криптографии. Ввиду отсутствия образующих элементов используют элементы максимального порядка.

Приведем алгоритм поиска элемента максимального порядка группы $Z_{p \cdot q}^*$. Пусть $n = p \cdot q$, где p и q – различные нечетные простые числа. Тогда $Z_{p \cdot q}^*$ не циклическая группа порядка $\varphi(n) = (p-1)(q-1)$.

1.1.3 Поля. Многочлены над полем

Полем называется кольцо R с единицей, множество $R \setminus \{0\}$ ненулевых элементов которого с операцией $\times$ является мультипликативной абелевой группой.

Полем является первое кольцо, рассмотренное в примере 1.11.

Примерами бесконечных полей являются поля Q рациональных, R действительных и C комплексных чисел. В то же время кольцо целых чисел полем не является.

Подмножество поля, замкнутое относительно обеих операций и являющееся полем, называется *подполем*.

Поле, не имеющее подполя, не совпадающего с самим полем, называется

простым полем. Имеется единственное простое бесконечное поле – поле Q рациональных чисел.

Конечные поля называются *полями Галуа* по имени французского математика Эвариста Галуа (1811–1832), построившего теорию конечных полей в её современном представлении. Поля Галуа обозначают $GF(q)$ или F_q, где q – число элементов, или *порядок* поля. Мультипликативная группа поля F_q обозначается F_q^*, ее порядок на единицу меньше порядка поля: $|F_q^*| = q - 1$.

Простейшим полем является поле из двух элементов – поле $GF(2)$. Операции этого поля определяются таблицами, из которых следует, что сложение соответствует булевой функции сложения по модулю 2, а умножение – конъюнкции:

$+$	a	
b	0	1
0	0	1
1	1	0

$\times$	a	
b	0	1
0	0	0
1	0	1

Поля $GF(p)$, где p – простое число, являются *простыми* полями. Операциями поля $GF(p)$ являются операции сложения и умножения по модулю p. Существуют все простые поля $GF(p)$, более того, такие поля составляют класс всех простых конечных полей.

Два поля F^1 и F^2 называются *изоморфными*, если существует биекция $\varphi : F^1 \to F^2$, сохраняющая операции. Эта биекция и обратная к ней функция φ^{-1} называется *изоморфизмами.* Простое поле изоморфно полю, являющемуся фактор-кольцом Z_p кольца Z целых чисел по модулю простого числа p.

Понятие поля позволяет вводить и использовать большое разнообразие колец, элементы которых определяются как многочлены

$$f(X) = a_0 + a_1X + a_2X^2 + \dots a_nX^n$$

с коэффициентами a_i из данного поля F. Такие многочлены называются *многочленами над полем* F. Наибольшее число n, такое, что коэффициент $a_n \neq 0$, называется *степенью* многочлена $f(X)$. Многочлен степени $n-1$ над полем F однозначно представим упорядоченным набором $(a_0, a_1, \dots, a_{n-1})$ коэффициентов. Иногда удобно использовать наборы большей длины, полученные добавлением старших нулевых элементов. Если $a_{n-1} = 1$, то многочлен степени $n-1$ называется *нормированным.*

Кольцо многочленов над полем F обозначается $F[X]$. Так существуют многочлены над полями Q, R и C рациональных, действительных и комплексных чисел и соответствующие кольца $Q[X]$, $R[X]$ и $C[X]$ таких многочленов. В криптогафии и теории кодирования фундаментальное значение имеют многочлены над простыми полями Z_p и кольца $Z_p[X]$ таких многочленов. Формально операции над многочленами определяются теми же правилами, по которым складываются или перемножаются многочлены над действительным полем.

Операция *сложения* сопоставляет двум многочленам степени не выше, чем $n-1$ $p_1(X)=\sum_{i=0}^{n-1}a_iX^i$ и $p_2(X)=\sum_{i=0}^{n-1}b_iX^i$, их сумму

$$p_1(X)+p_2(X)=\sum_{i=0}^{n-1}(a_i+b_i)X^i. \tag{1.4}$$

Здесь и ниже в слагаемых формул, подобных формуле в правой части, имеются в виду операции сложения и умножения в поле F.

Результатом операции *умножения* многочленов $p_1(X)=\sum_{i=0}^{n-1}a_iX^i$ и $p_2(X)=\sum_{i=0}^{n-1}b_iX^i$ является многочлен

$$p(X)=p_1(X)\times p_2(X)=\sum_{i=0}^{2n-2}c_iX^i, \tag{1.5}$$

где $c_i=\sum_{t+l=i}a_tb_l$.

Аддитивной единицей кольца многочленов является многочлен 0, все коэффициенты которого нулевые, то есть равны аддитивной единице поля. Степень такого многочлена не определяется. Мультипликативной единицей кольца многочленов является многочлен 1 нулевой степени. Кольцо многочленов не является полем, так как не для всякого многочлена имеется обратный к нему многочлен.

В соответствии с общим определением отношения делимости в кольце, многочлен $f(X)$ делит многочлен $g(X)$, $(f(X)|g(X))$, если существует многочлен $q(X)$ такой, что $g(X)=q(X)\times f(X)$. Отношение делимости в кольце многочленов порождает отношение эквивалентности

$$f(X)\approx g(X)\iff f(X)|g(X)\text{ и }g(X)|f(X).$$

Классы эквивалентности по этому отношению содержат многочлены, получающиеся один из другого умножением на некоторый многочлен нулевой степени (элемент поля). Удобно в качестве представителей этих классов использовать нормированные многочлены, то есть многочлены, имеющие при старшей степени переменной коэффициент 1. Фактор множество $F[X]/_{\approx}$ кольца $F[X]$ по этому отношению, в свою очередь, является кольцом, а множество представителей его элементов есть подкольцо кольца $F[X]$. В обоих случаях (для фактор множества и подкольца) используется обозначение $F[X]_{\approx}$.

Отношение делимости на множестве $F[X]_{\approx}$ превращает это множество в решетку с универсальными границами. Атомы решетки представляются (или являются) нормированными *неприводимыми,* многочленами, т.е. многочленами, неразложимыми на нетривиальные (неконстантные) множители из того же кольца (подкольца).

Как и во всяком кольце, в кольце многочленов можно определить операцию деления с остатком, сопоставившую многочленам $f(X)$ и отличному от 0 многочлену $g(X)$ многочлен $r(X)$ степени меньшей степени многочлена $g(X)$, такой,

что существует многочлен $q(X)$, удовлетворяющий равенству

$$f(X) = q(X) \times g(X) + r(X),$$

где $r(X) = f(X) \bmod g(X) = \mathrm{rem}(f(X), g(X))$.

Операции сложения, вычитания, умножения и деления в простейшем случае можно выполнять по алгоритмам, применяемым при сложении, умножении и делении целых чисел, представляемых в позиционной системе счисления с основанием, равным порядку поля F с тем отличием, что при сложении и вычитании не учитываются переносы в старший разряд.

Инфимум двух нормированных многочленов $f(X)$ и $g(X)$ можно представить через эти многочлены следующим образом

$$d(X) = (f(X), g(X)) = p(X) \times f(X) + q(X) \times g(X). \tag{1.6}$$

Для вычисления $d(X) = (f(X), g(X))$ многочленов $d(X)$,$p(X)$ и $q(X)$, удовлетворяющих соотношению 1.4, можно использовать Алгоритм 1.4.

Для вычисления НОД$(f(X), g(X))$ применяется алгоритм 1.5.

Пример 1.1.11 Применим Алгоритм 1.5 для вычисления инфимума $[X^5 + X^2 + 1]_{\approx} \wedge [X^4 + X + 1]_{[\approx]}$ применительно к кольцу $GF[2][X]$. Вычисления представлены в таблице (ввиду того, что любой элемент кольца $GF(2)[X]$ совпадает с противоположным к нему элементом, в данном случае при вычислениях обозначения sign знаков элементов не используются)

	1	2	3.1^1	3.2^1	3.1^2	3.2^2
a	X^5+X^2+1			X^4+X+1		$X+1$
b	X^4+X+1			$X+1$		1
r			$X+1$		1	
q	1		X		X^3+X^2+X	
x_2		1		0		1
x_1		0		1		X^3+X^2+X
x			1		X^3+X^2+X	
y_2		0		1		X
y_1		1		X		$X^4+X^3+X^2+1$
y			X		$X^4+X^3+X^2+1$	

	3.1^3	3.2^3	4
a		1	
b		0	
r	0		
q	$X+1$		
x_2		X^3+X^2+X	
x_1		X^4+X+1	
x	X^4+X+1		X^3+X^2+X
y_2		$X^4+X^3+X^2+1$	
y_1		X^5+X^4+1	
y	X^5+X^2+1		$X^4+X^3+X^2+1$
d			1

Полученный многочлен X^3+X^2+X является обратным по модулю многочлена X^4+X+1 по отношению к многочлену X^5+X^2+1, а многочлен $X^4+X^3+X^2+1$ является обратным по модулю многочлена X^5+X^2+1 по отношению к многочлену X^4+X+1 в кольце $GF(2)[X]$. Действительно, не трудно проверить, что

$$((X^5+X^2+1)\times(X^3+X^2+X))+((X^4+X^3+X^2+1)\times(X^4+X+1))=1.$$

Применив вместо алгоритма 1.5 алгоритм 1.6, получим НОД$(X^5+X^2+1, X^4+X+1)=1$ и те же многочлены X^3+X^2+X и $X^4+X^3+X^2+1$.

Далее на множестве $F[X]_{\approx}$ можно определить отношение конгруэнтности

$$f_1(X) \equiv\ f_2(X)\ (\mathrm{mod\ g(X)}) \iff \mathrm{g(X)|(f_1(X)-f_2(X))},$$

разбивающее множество $F[X]_{\approx}$ на классы конгруэнтности $[c(X)]_{\equiv \mathrm{mod}g(X)}$ по модулю многочлена $g(X)$. Как и в общем случае будем использовать обозначение $f(X) \bmod g(X)=c(X)$, если $f(X)\in[c(X)]_{\equiv mod\ g(X)}$. Множество классов конгруэнтности с операциями, соответствующими операциям сложения и умножения по модулю $g(X)$ их представителей, образуют конечное кольцо. Соответственно кольцом является и множество самих представителей, назовем его кольцом нормированных многочленов по модулю многочлена $g(X)$.

1.2 Поля Галуа

1.2.1 Мультипликативная группа конечного поля

Приведем некоторые факты о мультипликативной группе конечного поля.

Порядок мультипликативной группы F_q^* конечного поля F_q равен $q-1$. Порядок $\mathrm{ord}\ a$ ненулевого элемента a конечного поля F_q определяется как порядок этого же элемента мультипликативной группы этого поля. Поэтому из теоремы Лагранжа и следствий вытекают утверждения

Теорема 1.2.1 *Для ненулевого элемента a поля $F_q=GF(q)$ справедливы следующие утверждения.*

(1) $\mathrm{ord}\ a|q-1$.

(2) $a^{q-1}=1$;

(3) $a^{q-2}=a^{-1}$;

(4) *для любого натурального числа n и любого элемента a поля F_q справедливо*

$$a^{q^n-1}=1;$$

(5) *для любого натурального числа n и любого элемента a поля F_q справедливо*

$$a^{q^n}=a.$$

Доказательство. (1,2,3) непосредственно получаем из Следствий 1.1, 1.2 и 1.4;

(4) справедливо по следствию 1.3, так как $q-1$ делит число q^n-1; (5) следует из (4).

Теорема 1.2.2 *Если a – ненулевой элемент порядка δ поля F_q; $n, m \in N$, то*

$$a^m = a^n \iff m \equiv n \bmod \delta.$$

Доказательство. По Следствию 1.3

$$a^{m-n} = 1 \iff \delta | m-n.$$

Отсюда следует

Теорема 1.2.3 *Если a – ненулевой элемент порядка δ поля F_q, то элементы*

$$1, a, a^2, \ldots, a^{\delta-1} \tag{1.7}$$

поля F_q все различны.

Теорема 1.2.4 *Если a – ненулевой элемент порядка δ поля F_q, то элементы* (2.1) *поля F_q суть все корни многочлена*

$$X^\delta - 1.$$

Доказательство. При любом натуральном k

$$(a^k)^\delta = (a^\delta)^k = 1.$$

Поэтому перечисленные элементы являются корнями многочлена. Других корней этот многочлен не имеет, так как число этих элементов равно степени многочлена и все они различны.

Теорема 1.2.5 *Если a – ненулевой элемент порядка δ поля, то порядок его степени $a^k, k \in \mathbf{N}$, равен*

$$\operatorname{ord}\, a^k = \frac{\delta}{(\delta, k)},$$

в частности, ord $a^k = \delta$, тогда и только тогда, когда $(\delta, k) = 1$.

Доказательство получаем как следствие Теоремы 1.2.

Теорема 1.2.6 *Если a – элемент порядка δ поля F_q, то поле содержит $\varphi(\delta)$ элементов порядка δ.*

Доказательство. По теореме 2.5 среди элементов (2.1) ровно $\varphi(\delta)$ имеют порядок δ.

Теорема 1.2.7 *Если δ – натуральный делитель числа $q-1$, то число элементов порядка δ поля F_q равно $\varphi(\delta)$.*

Доказательство. Обозначим $\psi(\delta)$ число элементов порядка δ поля F_q. По утверждению (1) теоремы 2.1 имеем

$$\sum_{\delta|q-1} \psi(\delta) = q-1. \tag{1.8}$$

Учитывая, что в силу тождества Гаусса для функции Эйлера

$$\sum_{\delta|q-1} \varphi(\delta) = q-1,$$

и принимая во внимание (2.2), получим, что

$$\sum_{\delta|q-1} (\varphi(\delta) - \psi(\delta)) = 0. \tag{1.9}$$

По теореме 2.6

$$\forall \delta \in N,\ \psi(\delta) \leq \varphi(\delta),$$

поэтому из (2.3) следует

$$\psi(\delta) = \varphi(\delta),$$

если $\delta|q-1$.

Следствие 1.2.1 *Мультипликативная группа конечного поля F_q – циклическая.*

Действительно, по теореме 2.7 при $\delta = q-1$ получаем, что группа F_q^* имеет $\varphi(q-1)$ образующих элементов.

Образующий элемент циклической группы F_q^* называется *примитивным элементом* поля F_q и поле F_q содержит $\varphi(q-1)$ примитивных элементов.

1.2.2 Характеристика поля

Характеристикой поля называется наименьшее натуральное число m, такое, что $m * 1 = 0$, или число 0, если такого числа m не существует.

Иными словами, характеристика поля определяется как аддитивный порядок мультипликативной единицы поля.

Следующие факты получим из этого определения непосредственно.

Следствие 1.2.2 *Характеристика конечного поля не равна нулю.*

Следствие 1.2.3 *Характеристика поля ненулевой характеристики есть число простое.*

Следствие 1.2.4 *Если p – характеристика поля F, а m, n, k и l – простые числа, то*

(1) $m * 1 = n * 1 \iff m \equiv n \pmod{p}$,
(2) $(m * 1) + (n * 1) = k * 1 \iff m + n \equiv k \pmod{p}$,
(3) $(m * 1) \cdot (n * 1) = l * 1 \iff m \cdot n \equiv l \pmod{p}$.

Следствие 1.2.5 *Если p – характеристика поля F, то $\forall a \in F \; p * a = 0$.*

Следствие 1.2.6 *Всякое конечное простое поле характеристики p изоморфно кольцу классов конгруэнтности кольца целых чисел по модулю p.*

Следствие 1.2.7 *Всякое конечное поле характеристики p содержит простое подполе из p элементов.*

В любом поле результат *операции деления* элемента a на ненулевой элемент b определяется как элемент $c = a \cdot b^{-1}$.

Следующее утверждение позволяет существенно упрощать алгоритмы выполнения арифметических операций в конечных полях.

Теорема 1.2.8 *Пусть H – поле характеристики p; $a, b \in H$. Тогда для любого натурального k*

$$(a + b)^{p^k} = a^{p^k} + b^{p^k},$$
$$(a - b)^{p^k} = a^{p^k} - b^{p^k}.$$

Если $b \neq 0$, то

$$\left(\frac{a}{b}\right)^{p^k} = \frac{a^{p^k}}{b^{p^k}}.$$

Доказательство. При $k = 1$ все члены разложения $(a+b)^p$ по формуле бинома Ньютона, кроме первого a^p и последнего b^p, имеют множитель p и равны 0 по теореме 2.1. Индуктивный переход очевиден. Второе утверждение для нечетного p следует из первого $(a + (-b))^{p^k} = a^{p^k} + (-b)^{p^k} = a^{p^k} - b^{p^k}$, а при $p = 2$ не отличается от первого. Свойство частного проверяется непосредственно по его определению:

$$\left(\frac{a}{b}\right)^{p^k} = (a \times b^{-1})^{b^k} a^{p^k} \times b^{-p^k} = \frac{a^{p^k}}{b^{p^k}}..$$

Следствие 1.2.8 *Возведение в степень p^k многочлена над полем характеристики p можно осуществить возведением в степень отдельных членов:*

$$(1 + a_1 X + \ldots + a_n X^n)^{p^k} = 1 + (a_1 X)^{p^k} + \ldots + (a_n X^n)^{p^k}.$$

Так при $p = 2$ получим $(1 + X^2 + X^3)^4 = 1 + X^8 + X^{12}$.

Пусть $f(X)$ есть многочлен над полем F. Элемент $x \in F$ такой, что $f(x) = 0$, называется *корнем многочлена* $f(X)$ *в поле* F.

Следствие 1.2.9 *Пусть* $f(X)$ *есть многочлен над полем характеристики* p *и* x *есть корень этого многочлена в том же поле. Тогда при любом натуральном* k *элемент* $x^{p^k} \in F$ *также является корнем этого многочлена.*

Действительно,

$$f(x^{p^k}) = f(x)^{p^k} = 0.$$

1.2.3 Конечное расширение поля

Пусть F – подполе поля H. Минимальное поле, содержащее F и элемент $\theta \in H,\ \theta \notin F$, называется *простым расширение поля* F и обозначается $F(\theta)$.

Пример 1.2.1 Множество $H = \{0,1\}^3$ с операциями $+$ и $\times$, представленными в следующих таблицах

$+$	**000**	**001**	010	011	100	101	110	111
000	**000**	**001**	010	011	100	101	110	111
001	**001**	**000**	011	010	101	100	111	110
010	010	011	000	001	110	111	100	101
011	011	010	001	000	111	110	101	100
100	100	101	110	111	000	001	010	011
101	101	100	111	110	001	000	011	010
110	110	111	100	101	010	011	000	001
111	111	110	101	100	011	010	001	000

$\times$	**000**	**001**	010	011	100	101	110	111
000	**000**	**000**	000	000	000	000	000	000
001	**000**	**001**	010	011	100	101	110	111
010	000	010	100	110	011	001	111	101
011	000	011	110	101	111	100	001	010
100	000	100	011	111	101	010	101	001
101	000	101	001	100	001	111	011	110
110	000	110	111	001	110	011	010	100
111	000	111	101	010	010	110	100	011

является полем H. (Эти таблицы составлены таким образом, чтобы операции обладали свойствами операций поля. что не трудно проверить).

Множество $\{000, 001\}$ с операциями, представленными в тех же таблицах жирным шрифтом также является полем. Обозначим это поле F. Поле H получается из поля F добавлением любого нового элемента θ из $\{0,1\}^3$, а затем последовательно и всех остальных элементов этого множества, поскольку каждый из них может быть получен из предыдущих с помощью операций поля H. Множество $\{0,1\}^3$ замкнуто относительно операций поля H, таким образом, это поле является минимальным полем, содержащим элементы поля F и элемент θ.

Вместо элемента (0,1,0) можно было выбрать любой другой отличный от (0,0,0) и (0,0,1) элемент из $\{0,1\}^3$.

Таким образом,

$$H = F(010) = F(011) = F(100) = F(101) = F(110) = F(111)$$

и H есть простое расширение поля F в поле H.

В данном случае H есть поле Галуа $GF(2^3)$, а F – поле Галуа $GF(2)$. Поле H содержит $\varphi(2^3) = 4$ примитивных элемента

$$010,\ 010^3,\ 010^5\ 010^7,$$

а поле $GF(2)$ имеет $\varphi(2) = 1$ примитивный элемент 001.

Если $x = \theta$ есть корень в поле H некоторого многочлена f степени n над полем F, то простое расширение $F(x)$ называется *простым алгебраическим расширением, полученным путем присоединения к полю F корня x многочлена f*. Если при этом многочлен f является неприводимым многочленом степени n, то расширение $F(x)$ называется *простым алгебраическим расширением поля F степени n*.

Пример 1.2.2 Поле $GF(2^2)$ можно получить присоединением к полю $GF(2)$ корня x неприводимого многочлена $1 + X + X^2$ и определив операцию умножения, отражающую, что $x^2 = x + 1$:

$$x \cdot 1 = x,$$

$$x \cdot x = 1 + x,$$

$$x \cdot (1 + x) = 1,$$

$$(1 + x)(1 + x) = x.$$

Заметим, что элемент $x = 010$ поля H из предыдущего примера есть корень неприводимого многочлена $f(X) = X^3 + X + 1$ над полем F (Можно проверить, что $f(x) = x^3 + x + 1 = 000$). Поле H можно получить присоединением к полю F этого корня: $H = F(010)$. Значит, H есть простое алгебраическое расширение поля F степени 3. Операция умножения отражает тот факт. что $X^3 = X^2 + 1$, что и было использовано при составлении таблицы умножения.

Пример 1.2.3 Поле комплексных чисел есть простое алгебраическое поле поля действительнвх чисел, получаемое присоединением корня i неприводимого многочлена $X : 2 + 1$ над полем действительных чисел.

Определение 2.3. Пусть F – подполе поля H. Поле H называется *конечным расширением поля F*, если поле H содержит элементы

$$h_1,\ h_2,\ \ldots,\ h_k,$$

такие, что любой элемент h из H линейно над полем F выражается через эти элементы, то есть уравнение

$$h = h_1 \cdot x_1 + h_2 \cdot x_2 + \ldots + h_k \cdot x_k$$

разрешимо в элементах $x_1, \ldots, x_k$, принадлежащих полю F.

Если при этом решение единственно, то указанный набор элементов называется *базисом поля H относительно поля F*, а число k элементов базиса называется *степенью конечного расширения поля H относительно поля F* и обозначается $k = [H : F]$.

Пример 1.2.4 Поле K комплексных чисел является конечным расширением степени 2 поля R действительных чисел, $[K : R] = 2$. Базисом поля комплексных чисел являются его элементы $h_1 = (1, 0) = 1 + 0i$ и $h_2 = (0, 1) = 0 + 1i$.

Пример 1.2.5 Поле $GF(2^2)$ является конечным расширением поля $GF(2)$ степени 2, $[GF(2^2) : GF(2)] = 2$. В качестве базиса $GF(2^2)$ можно выбрать, например, элементы $h_1 = 1$ и $h_2 = x$, где x – корень в поле $GF(2^2)$ неприводимого многочлена $f(X) = X^2 + X + 1$ над полем $GF(2)$.

Базис составляют также элементы x и x^2.

В качестве упражнений можно убедиться в справедливости следующих утверждений

а) Поле F является подполем любого конечного расширения H этого поля.

б) Характеристика конечного расширения поля равна характеристике его простого подполя.

в) Если H – конечное расширение поля W, а поле W – конечное расширение поля F, то H – конечное расширение поля F, при этом

$$[H : F] = [H : W] \cdot [W : F].$$

Пример 1.2.6 Поле $GF(2)$ есть подполе поля $GF(2^2)$. Характеристика поля $GF(2^2)$ равна 2: (1+1=0).

Пример 1.2.7 Поле $GF(2^4)$ можно рассматривать как конечное расширение степени 2 поля $GF(2^2)$, которое, в свою очередь, является конечным расширением степени 2 поля $GF(2)$. При этом $[GF(2^4) : GF(2)] = GF(2^4) : GF[2^2] \cdot [GF(2^2) : GF(2)]$.

Теорема 1.2.9 *Пусть F_q – конечное поле и $F_q(\theta)$ – его простое конечное расширение. Тогда $F_q(\theta)$ является простым алгебраическим расширением поля F_q,*

Доказательство. Пусть $F_q(\theta)$ – подполе поля H, $\theta \in H$. Рассмотрим максимальное множество линейно независимых над полем F_q элементов поля $F_q(\theta)$,

$$1, \theta, \ \theta^2, \ \ldots, \ \theta^{n-1}. \tag{1.10}$$

Элемент θ^n линейно зависит от этих элементов:

$$\theta^n = a_0 + a_1\theta + a_2\theta^2 + \ldots + a_{n-1}\theta^{n-1}, \ a_i \in F, \ i = 1, \ldots, \ n-1$$

Таким образом, θ есть корень в поле H многочлена

$$f(X) = X^n - a_{n-1}X^{n-1} - \ldots - a_1X - a_0$$

над полем F_q.

Ввиду того, что выбрано максимальное количество линейно независимых элементов, построенный многочлен неприводим.

Теорема 1.2.10 *Любой элемент α простого алгебраического расширения $F(\theta)$ степени n поля F однозначно представим в виде*

$$\alpha = a_0 + a_1\theta + a_2\theta^2 + \ldots + a_{n-1}\theta^{n-1}, \tag{1.11}$$

где θ корень в $F(\theta)$ некоторого неприводимого многочлена степени n над полем F.

Доказательство. Предположение о неоднозначности приводит к тому, что неприводимый многочлен f степени n имеет общий корень с многочленом меньшей степени, (то есть многочлен, корень которого присоединен, не является неприводимым).

Если элемент α принадлежит полю $F(\theta)$, то он может быть представлен формулой, содержащей константы из множества F и корень θ известного неприводимого многочлена степени n. Используя свойства операций поля, эту формулу можно привести к полиномиальному (относительно θ) виду. Затем итерациями можно преобразовать полученный полином к полиному степени не более $n-1$. Итерация применяется к полиному степени m не ниже n и заключается в замене *старшего* ненулевого члена $a_m\theta^m$ равным ему (поскольку $f(\theta) = 0$) выражением $a_m(\theta^{m-n}f(\theta) + \theta^m)$. После приведения подобных членов получается многочлен меньшей степени.

Множество $M(\theta)$ выражений (2. 5) замкнуто относительно операций сложения и умножения многочленов (с учетом того, что θ – корень неприводимого многочлена).

Следствие 1.2.10 *Всякое простое алгебраическое расширение поля F степени n является конечным расширением поля F степени n.*

Следствие 1.2.11 *Пусть H – конечное поле, содержащее подполе K, состоящее из q элементов. Тогда H состоит из q^m элементов, где $m = [H : K]$.*

Следствие 1.2.12 `Множество (2.4) составляет базис поля` $F(\theta)$.

1.2.4 Поле разложения многочлена. О числе элементов конечного поля

Определение 2.4. Пусть F – поле, f – многочлен над полем F. *Полем разложения многочлена f над полем F* называется минимальное расширение H поля F, в котором f разлагается в произведение линейных над полем F многочленов.

Пример 1.2.8 Многочлен $1 + X + X^2$ разлагается на линейные множители в поле $GF(2^2)$:

$$1 + X + X^2 = (X + x)(X + (1 + x)) = X^2 + X + Xx + Xx + 1,$$

где x – корень многочлена в поле $GF(2^2)$. В поле $GF(2)$ подобное разложение не существует (указанный многочлен неприводим) Следовательно, $GF(2^2)$ есть поле разложения многочлена $1 + X + X^2$.

Теорема 1.2.11 *(О существовании конечного поля) Для всякого простого числа p и всякого натурального числа n существует поле Галуа $GF(p^n)$ и таким полем является поле разложения H многочлена $X^q - X$, $q = p^n$ над полем F_p. Характеристика этого поля есть число p.*

Доказательство. Легко проверить, что по теореме 2.8 сумма, произведение и частное (при ненулевом делителе) корней многочлена $f(X) = X^q - X$ являются корнями этого же многочлена. Корнями являются. в частности, 0 и 1, при этом $p * 1 = 0$, $(p-1) * 1 = -1$. Каждому корню θ соответствует противоположный элемент $(p-1) * \theta$, также являющийся корнем указанного многочлена. Аналогично, обратный элемент θ^{-1} также является его корнем. Легко проверить, что выполняются и другие свойства операций поля. Таким образом, множество корней многочлена $f(X)$ есть подполе поля H. Заметим, что производная $f'(X)$ этого многочлена

$$f'(X) = q * X^{q-1} - 1 = -1 \neq 0.^4$$

Поэтому многочлен и его производная взаимно просты, откуда следует, что все корни многочлена $f(X)$ различны, то есть подполе корней многочлена совпадает с полем H.

Следствие 1.2.13 *Если F – конечное поле из q элементов, то каждый элемент $a \in F$ удовлетворяет равенству*

$$a^q = a.$$

Следствие 1.2.14 *Если K – конечное поле из q элементов и F – подполе поля K, то многочлен $X^q - X$ над полем F разлагается в поле K следующим образом*

$$X^q - X = \Pi_{a \in K}(X - a),$$

так что K является полем разложения многочлена $X^q - X$.

Пример 1.2.9 Многочлен $X^{2^2} + X$ разлагается на линейные множители в поле $GF(2^2)$:

$$X^4 + X = (X + 0)(X + 1)(X + \theta)(X + (1 + \theta)).$$

(В поле характеристики 2 минус и плюс взаимозаменяемы). Видим, что его корнями в поле $GF(2^2)$ являются $0, 1, \theta$ и $1+\theta$. ($x = \theta$ – корень в $GF(2^2)$ неприводимого многочлена $1+X+X^2$ над полем $GF(2)$). Элементы $0, 1, \theta$ и $1 + \theta$ являются корнями уравнения

$$X^4 = X.$$

1.2.5 Минимальные многочлены. Существование неприводимых многочленов

По следствию 2.13 каждый элемент θ поля H, являющегося конечным расширением поля F, есть корень многочлена $f(X) = X^q - X$ над полем F.

Если рассмотреть разложение этого многочлена на неприводимые над полем F множители, можно увидеть, что элемент θ является корнем точно одного из таких множителей $M(\theta)$, который называется *минимальным многочленом* элемента θ.

Минимальный многочлен примитивного элемента поля называется *примитивным многочленом.*

Справедливы следующие теоремы о примитивных многочленах.

---

[4]Производная многочлена определяется правилами дифференцирования непрерывной функции.

Теорема 1.2.12 *(Достаточное условие примитивности многочлена) Если $f(x)$ – неприводимый многочлен над полем $GF(2)$ степени m и m – простое число, то $f(x)$ – примитивный многочлен.*

Теорема 1.2.13 *(Необходимое и достаточное условие примитивности многочлена). Неприводимый многочлен $f(X)$ над полем $GF(p)$ примитивен тогда и только тогда, когда для любого простого делителя d числа p^m-1 многочлен*

$$X^{\frac{p^m-1}{d}}$$

не сравним с 1 по модулю $f(X)$.

Следствие 1.2.15 *Если число p^m-1 – простое, то неприводимый многочлен $f(X)$ степени m над полем $GF(p)$ примитивен.*

Элементы расширегия F_q поля F_p, имеющие один и тот же минимальный многочлен, называются *сопряженными* относительно поля F_p элементами.

Пример 1.2.10 Многочлен X^4+X разлагается над полем $GF(2)$ на неприводимые многочлены следующим образом:

$$X^4+X=X(1+X)(1+X+X^2).$$

Многочлен X является минимальным многочленом элемента 0 поля $GF(2^4)$, многочлен $1+X$ есть минимальный многочлен элемента 1 поля $GF(2^4)$, многочлен $1+X+X^2$ – минимальный многочлен элементов θ и $1+\theta$ поля $GF(2^2)$. Поскольку последние два элемента являются примитивными элементами поля $GF(2^2)$, то их минимальный многочлен $1+X+X^2$ является примитивным многочленом степени 2 над полем $GF(2)$. Элементы θ и $(1+\theta)$ являются сопряженными корнями этого многочлена и тем самым – сопряженными относительно поля $GF(2)$ элементами.

Пример 1.2.11 Приведём минимальные многочлены элементов поля $GF(2^4)$:

Элементы поля	Минимальный многочлен
0	X
1	$X+1$
α, α^2, α^4, α^8	X^4+X+1
$\alpha^{-1}=\alpha^{14}$, $\alpha^{-2}=\alpha^{13}$, $\alpha^{-4}=\alpha^{11}$, $\alpha^{-8}=\alpha^7$	X^4+X^3+1
α^3, α^6, α^{12}, $\alpha^{24}=\alpha^9$	$X^4+X^3+X^2+X+1$
α^5, α^{10}	X^2+X+1

Теорема 1.2.14 *Пусть F_{p^m} – расширение простого поля F_p, и пусть $\theta\in F_{p^m}$. Тогда элементы*

$$\theta,\theta^q,\theta^{q^2},\ldots,\theta^{p^{m-1}} \tag{1.12}$$

являются сопряженными относительно поля F_p элементами.

Доказательство. Пусть $f(X)$ – неприводимый многочлен над полем F, $x=\theta$ – его корень в конечном расширении $F(\theta)$ поля F. По теореме 2.8, x^{p^k} являются корнями того же многочлена $f(X)$.

Замечание. Элементы ряда (2.6) называют также *элементами, сопряженными с элементом θ* поля F_{q^m}.

Пример 1.2.12 Применительно к $GF(2^2)$ имеем ряды сопряженных элементов

$$0; 1; \theta,\ \theta^2 = 1 + \theta.$$

Теорема 1.2.15 *Сопряженные относительно поля F_p элементы поля F_{p^m} имеют один и тот же порядок.*

Пример 1.2.13 Для рассмотренных в предыдущем примере элементов поля $GF(2^2)$ имеем

$$\theta^3 = 1,\ (1+\theta)^3 = 1.$$

$$\theta^2 \neq 1,\ (1+\theta)^2 \neq 1.$$

Следствие 1.2.16 *Если θ – примитивный элемент поля F_q, то примитивными будут и все сопряженные с ним относительно любого подполя поля F_q элементы.*

Пример 1.2.14 Рассмотренные в предыдущем примере сопряженные элементы θ и $1+\theta$ являются примитивными элементами поля $GF(2^2)$.

Следствие 1.2.17 *Поля F_{q^m} изоморфны.*

Доказательство. Изоморфизм одного поля в другое однозначно определяется выбором пары примитивных элементов, по одному в каждом поле. Эти примитивные элементы могут быть корнями разных примитивных многочленов.

Пример 1.2.15 Можно рассмотреть два изоморфных расширения поля $GF(2)$. Первое получается присоединением корня θ неприводимого многочлена $1 + X + X^2$, а второе – присоединением сопряженного корня $1 + \theta$. Изоморфизм ψ отображает первое поле на второе следующим образом:

$$\psi(0) = 0,\ \psi(1) = 1,\ \psi(\theta) = 1 + \theta,\ \psi(1 + \theta) = \theta.$$

Пример 1.2.16 В одном варианте поля $GF(2^4)$ можно взять в качестве примитивного элемента корень α примитивного многочлена $X^4 + X + 1$, а в другом варианте поля – корень $\alpha^{-1} = \alpha^{14}$ примитивного многочлена $X^4 + X^3 + 1$. Варианты поля изоморфны, $\psi(\alpha) = \psi(\alpha^{14})$. Остальные значения функции ψ определяются однозначно.

Теорема 1.2.16 *Сопряженные относительно поля F_q элементы поля F_{q^m} различны тогда и только тогда, когда минимальный многочлен этих элементов над F_q имеет степень m. Если же это не так, то степень d минимального многочлена элемента θ является собственным делителем числа m, и тогда среди сопряженных с θ относительно F_q элементов различными будут лишь*

$$\theta, \theta^q, \theta^{q^2}, \ldots, \theta^{q^{d-1}},$$

каждый из которых повторяется в ряду сопряженных m/d раз, поскольку $\theta^{q^m} = \theta$.

Пример 1.2.17 В поле $GF(2^2)$ $1 = 1^2;\ 0 = 0^2$. В то же время $\theta,\ \theta^2$, как и $(1+\theta), (1+\theta)^2$, различны.

Следствие 1.2.18 *Множество различных сопряженных относительно поля F_p элементов поля F_{p^m}, являющихся корнями данного минимального многочлена над полем F_p содержит все корни этого минимального многочлена в поле F_{p^m}.*

Теорема 1.2.17 *Для каждого конечного поля F_q и каждого натурального числа n в кольце $F_p[X]$ существует неприводимый многочлен степени n.*

Доказательство. Можно показать, что число $N_p(n)$ нормированных неприводимых многочленов степени n над простым полем F_p определяется формулой

$$N_p(n) = \frac{1}{n}\sum_{d|n}\mu(d)p^{n/d},$$

где μ есть функция Мебиуса,

$$\mu(n) = \begin{cases} 1, & \text{если } n = 1, \\ 0, & \text{если } n \text{ делится на квадрат простого числа,} \\ (-1)^k, & \text{если } n \text{ есть произведение } k \text{ различных простых чисел.} \end{cases}$$

Методы тестирования многочленов на неприводимость и методы генерации неприводимых во второй главе.

Пример 1.2.18 $N_2(2) = 1$

Замечание. Число $N_p(n)$ неприводимых многочленов удовлетворяет неравенству

$$\frac{1}{2n} \le \frac{N_p(n)}{p^n} \approx \frac{1}{n}$$

Теорема 1.2.18 *Пусть $f \in F_q[X]$ – неприводимый многочлен степени m над конечным полем F_q. Тогда $f(X)$ делит многочлен $X^{q^n} - X$ в том и только том случае, если число m делит n.*

Пример 1.2.19 Делителями многочлена $X^{2^2} - X$ являются многочлены X, $1 + X$, $X + X^2$, $1 + X + X^2$.

Теорема 1.2.19 *Если $f \in F_q[X]$ – неприводимый многочлен степени m, то в поле F_{q^m} содержится любой корень θ многочлена f. Более того, все корни многочлена f просты и ими являются m различных элементов θ, θ^q, $\theta^{q^2}, \ldots, \theta^{q^{m-1}}$ поля F_{q^m}.*

См. пример 2.13.

Следствие 1.2.19 *Пусть F – поле и f – неприводимый многочлен степени n над полем F. Тогда*

(1) *Существует поле разложения многочлена f над полем F;*
(2) *Любые поля разложения многочлена f над полем F изоморфны.*
(3) *Поле разложения многочлена f есть конечное расширение поля F степени n.*

Доказательство. Если n – степень неприводимого многочлена над полем F, содержащим q элементов, то его полем разложения является поле разложения многочлена $X^{q^n} = X$. Все такие поля изоморфны. Корни многочлена f составляют базис его поля расширения.

Теорема 1.2.20 *Критерий подполя. Пусть F_q – конечное поле из $q = p^n$ элементов (p –простое число). Тогда каждое подполе поля F_q имеет порядок p^m, где m является положительным делителем числа n. Обратно, если m – положительный делитель числа n, то существует ровно одно подполе поля F_q из p^m элементов.*

из p^m элементов.

1.2.6 След и норма элемента конечного поля

Пусть $P = F_q,\ K = F_{q^m}$ и $\alpha \in K$. *След* $\mathrm{Tr}_{K/P}(\alpha)$ элемента α из поля K в поле P определяется равенством

$$\mathrm{Tr}_{K/P}(\alpha) = \alpha + \alpha^q + \alpha^{q^2} + \ldots + \alpha^{q^{m-1}}.$$

Если P – простое подполе поля K, то $\mathrm{TR}_{K/P}$ называется *абсолютным следом* элемента α и обозначается $\mathrm{Tr}_K(\alpha)$.

Пример 1.2.20 Пусть $P = F = GF(2),\ K = GF(2^2)$. Тогда

$$Tr_K(0) = 0.$$

$$Tr_K(1) = 1 + 1 = 0,$$

$$Tr_K(\theta) = \theta + \theta^2 = 1,$$

$$Tr_K(1+\theta) = (1+\theta) + (1+\theta)^2 = 1.$$

Характеристическим многочленом элемента α над полем P называется многочлен $g(X) = f(X)^{m/d}$, где $f(X)$ – минимальный многочлен элемента α над полем $P,\ d$ – степень многочлена $f(X)$. Корнями многочлена $f(X)$ являются элементы

$$\alpha, \alpha^q, \alpha^{q^2}, \ldots, \alpha^{q^{d-1}}.$$

Корнями многочлена $g(X)$ в поле F являются те и только те элементы, которые сопряжены с элементом α относительно поля P. Отсюда

$$g(X) = X^m + a_{m-1}X^{m-1} + \ldots + a_1X_1 + a_0 = (X-\alpha)(X-\alpha^q)\ldots(X-\alpha^{q^{m-1}}).(*)$$

Сравнение коэффициентов дает

$$\mathrm{Tr}_{F/P}(\alpha) = -a_{m-1}.$$

Получаем, что след $\mathrm{Tr}_{F/P}(\alpha)$ всегда является элементом поля P.

Для $\alpha \in F = F_{q^m}$ и $P + F_q$ *норма* $N_{F/P}(\alpha)$ элемента α над полем P определяется равенством

$$N_{F/P}(\alpha) = \alpha \cdot \alpha^q \cdot \alpha^{q^2} \cdot \ldots \cdot \alpha^{q^{m-1}} = \alpha^{(q^m-1)/(q-1)}.$$

Сравнение в равенстве $(*)$ свободных членов приводит к заключению, что норма $N_{F/P}$ является элементом поля P :

$$N_{F/P}(\alpha) = (-1)^m a_0.$$

1.2.7 Алгоритмическое представление поля Галуа $GF(2^n)$

Алгоритмически поле Галуа $GF(q^n)$ удобно описывать, имея в виду его изоморфный образ в виде фактор множества $F_q[X]_{\mathrm{mod}\ f(X)}$ кольца $F_q[X]$ нормированных многочленов над полем F_q по модулю некоторого неприводимого многочлена $f(X)$ степени n над полем F_q. В этом случае поле F_{q^n} рассматривается как алгебраическое расширение $F_q(x)$ поля F_q степени n, где x – корень этого многочлена $f(X)$ в $GF(q^n)$, а его элементы – как представители этих классов.

Тогда элементы поля определяются упорядоченными наборами $\alpha = (a_0, a_1, a_2, \ldots, a_{n-1})$ коэффициентов многочленов $\alpha(X)$, представляющих классы $[\alpha(X)]_{\mathrm{mod}\ f(X)}$, они же – коэффициенты многочленов $\alpha(x)$, определяющих элементы расширения $F_q(x)$ поля F_q.

Операция сложения в поле F_{q^n} определяется как покомпонентное сложение указанных векторов с использованием аддитивной операции поля F_q :

$$< a_0, a_1, \ldots, a_{n-1} > + < b_0, b_1, b_2, \ldots, b_{n-1} >=< a_0 + b_0, a_1 + b_1, \ldots, a_{n-1} + b_{n-1} >,$$

то есть как набор коэффициентов суммы многочленов

$$\alpha(X) = \sum_{i=0}^{n-1} a_i X^i + \beta(X) = \sum_{i=0}^{n-1} b_i X^i = \sum_{i+0}^{n-1} (a_i + b_i X^i). \tag{1.13}$$

Определить произведение

$$< a_0, a_1, a_2, \ldots, a_{n-1} > \cdot < b_0, b_1, b_2, \ldots, b_{n-1} >$$

элементов $\alpha =< a_0, a_1, \ldots, a_{n-1} >$ и $\beta =< b_0, b_1, \ldots, b_{n-1} >$ в поле F_{q^n} можно двумя эквивалентными способами:

1) по теореме о делении с остатком

$$\alpha(X) \times \beta(X) = f(X) \times q(X) + \rho(X),\ \deg\ \rho(X) < n,$$

где $\rho(X) =\mathrm{rem}(\alpha(X) \times \beta(X), f(X))$ откуда

$$\alpha \times \beta = f \times q + \rho = \rho = (c_0, c_1, \ldots, c_{n-1}),$$

где f, q и ρ – наборы коэффициентов многочленов $f(X),\ q(X)$ и $\rho(X)$ (вектор ρ определяет класс эквивалентности, которому принадлежит произведение многочленов над полем $GF(2)$)

2) подставляя вместо формальной переменной XС корень x неприводимого многочлена $f(X)$:

$$\alpha(x) \times \beta(x) = f(x) \times q(x) + \rho(x) = \rho(x),$$

так как $f(x) = 0$.

Таким образом, произведение

$$< a_0, a_1, a_2, \ldots, a_{n-1} > \cdot < b_0, b_1, b_2, \ldots, b_{n-1} >$$

элементов α и β в поле F_{q^n} представляется вектором $\sigma =< c_0, c_1, c_2, \ldots, c_{n-1},$ который можно вычислить

$$\sigma =< c_0, c_1, c_2, \ldots, c_{n-1} >$$

можно вычислить как коэффициенты многочлена $\rho(X) = \mathrm{rem}(\alpha(X) \times \beta(X), f(X))$, например, делением многочленов "столбиком"произведения многочленов над полем Галуа $GF(2)$

$$\alpha(X) \times \beta(X) = \sum_{k=0}^{2n-2} \sum_{i+j=k} a_i b_j X^k, \tag{1.14}$$

на многочлен $f(X)$. или как коэффициенты многочлена $\alpha(x) \times \beta(x)$. В последнем случае возникшие в процессе формального умножения ненулевые члены

$$a_s \times x^s \tag{1.15}$$

степени $s > n-1$ в порядке убывания степеней переменной x заменяются выражениям выражениями $a_s x^s = a_s(x^s + x^{s-n} \times f(x))$ и приведении после каждой такой замены подобных членов ($x^s + x^{s-n} \times f(x) = x^s$, т.к. $f(x) = 0$). Этот процесс соответствует процессу вычисления коэффициентов многочлена $\rho(X)$, а по заменяемым в указанных итерациях ненулевым старшим членам (2.5) можно определить все ненулевые коэффициенты $q_i,\ 0 < i \leq\ n - m, q_i \neq 0$, частного $q(X)$:

$$q_i = a_s \times f_{n-1}^{-1},$$

где f_{n-1}-старший коэффициент полинома-делителя.

Напомним. что подобный прием был применен нами при доказательстве теоремы 2.10.

Рассмотренный способ формализуется в виде так называемого «школьного» алгоритма деления (См. 3.4.1).

1.2.8 Поле Галуа как векторное пространство

Векторное пространство над полем Галуа. В соответствии с общим определением векторного пространства векторное пространство над полем Галуа $GF(q)$ определяется как множество векторов

$$\mathbf{a} = (a_1, \ldots, a_i, \ldots, a_n)$$

определённой длины n, состоящих из элементов поля, на котором определена операция сложения как покомпонентное сложение векторов по правилам операции сложения в поле $GF(q)$ и (коммутативная) операция умножения вектора

$\mathbf{a}$ на на произвольный элемент α поля $GF(q)$, результатом которой является вектор произведений (по правилам умножения в $GF(q)$) соответствующих элементов исходного вектора на этот элемент поля:

$$\alpha \cdot \mathbf{a} = (\alpha a_1, \ldots, \alpha a_i, \ldots, \alpha a_n).$$

Учитывая свойства операций в поле $GF(q)$, не трудно проверить, что данное определение корректно, то есть удовлетворяет всем аксиомам общего определения векторного пространства: операция сложения в нём коммутативна и ассоциативна, имеется нулевой элемент, для любого вектора $\mathbf{a}$ можно указать противоположный вектор $\mathbf{b}$ такой, что $\mathbf{a} + \mathbf{b} = \mathbf{0}$, умножение на мультипликативную единицу поля $GF(q)$ не приводит к изменению вектора, умножение на элемент поля $GF(q)$ ассоциативно $(\alpha(\beta\mathbf{a})) = ((\alpha \cdot \beta)\mathbf{a})$, выполняется дистрибутивность относительно векторного множителя и дистрибутивность относительно "полевого"множителя

$$(\alpha + \beta) \cdot \mathbf{a} = \alpha \cdot \mathbf{a} + \beta \cdot \mathbf{a},$$

$$\alpha(\mathbf{a} + \mathbf{b}) = \alpha \cdot \mathbf{a} + \alpha \cdot \mathbf{b}.$$

Как и в общем случае выражение

$$\alpha_1 \mathbf{e_1} + \cdots + \alpha_i \mathbf{e_i} + \cdots \alpha_n \mathbf{e_n}$$

называется *линейной комбинацией* векторов $\mathbf{e_1}, \ldots, \mathbf{e_i}, \ldots, \mathbf{e_n}$ с коэффициентами

$$\alpha_1, \ldots, \alpha_i, \ldots \alpha_n.$$

Если хотя бы один из коэффициентов отличен от нуля, то линейная комбинация называется *нетривиальной.* Векторы $\mathbf{e_1}, \ldots, \mathbf{e_i}, \ldots, \mathbf{e_n}$ называются *линейно зависимыми,* если существует хотя бы одна их тривиальная комбинация, равная нулю, иначе векторы называются *линейно независимыми.* Векторное пространство называется n-мерным, если существует n линейно независимых векторов, а любые $n+1$ векторов линейно зависимы. Любые n линейно независимых векторов n-мерного векторного пространства образуют его *базис.* Если $\mathbf{e_1}, \ldots, \mathbf{e_i}, \ldots, \mathbf{e_n}$ есть базис векторного пространства, то любой вектор представим линейной комбинацией базисных векторов.

Не трудно заметить, что n-мерное векторное пространство над полем $GF(q)$ состоит из элементов поля $GF(q^n)$, являющегося алгебраическим расширением степени n поля $GF(q)$. Поэтому базис такого векторного пространства иногда называют *базисом поля* $GF(q^n)$.

Два базиса. Пусть задан неприводимый многочлен $p(X)$ над полем $GF(p)$ степени n. Тогда конечное расширение $GF(p)(x) = GF(p^n)$ степени n поля $GF(p)$, где x – корень многочлена $p(X)$, можно рассматривать как векторное пространство размерности n. По следствию 2.11, множество элементов

$$1,\ x,\ x^2,\ \ldots,\ x^{n-1}$$

составляет базис n-мерного векторного пространства и одновременно базис поля $GF(p^n)$. Действительно, это множество линейно независимо и позволяет представить любой элемент

$$q(x) = a_0 + a_1 \cdot x + a_2 \cdot x^2 + \ldots + a_{k-1}x^{n-1}. \tag{1.16}$$

из $GF(2^n)$.

Этот базис называется *полиномиальным* или *стандартным* базисом и обозначается S.

Базисом поля $GF(p^n)$ может оказаться также множество сопряженных с многочленом (элементом поля) x многочленов (элементов)

$$\{x,\ x^p,\ x^{p^2}, \ldots, x^{p^{n-1}}\}.$$

Это множество называется *нормальным* множеством.

Действительно, все эти элементы различны (Теорема 2.16), поскольку они являются корнями неприводимого многочлена $p(x)$, и если они окажутся линейно независимыми, то любой элемент поля $GF(2^n)$ можно будет представить линейной комбинацией

$$a = a_0 x + \ a_1 x^p + \ a_2 x^{p^2} + \ldots + \ a_{n-1}x^{p^{n-1}} = \sum_{i=0}^{n-1} a_i x^{p^i}.$$

Тогда это множество называется *нормальным* базисом поля $GF(2^n)$ и обозначается N.

Теорема 1.2.21 *Во всяком n-мерном векторном пространстве над полем $GF(p)$ конечном поле $GF(2^n)$ существует полиномиальный (стандартный) и нормальный базисы.*

Заметим, что в нормальном базисе возведение в степень характеристики поля p равносильно циклическому сдвигу векторного представления элемента поля (учитывая, что $x^{p^n} = x$):

$$a^p = (a_0 x + \ a_1 x^p + \ a_2 x^{p^2} + \ldots + \ a_{n-1}x^{p^{n-1}})^p =$$

$$= a_0 x^p + \ a_1 x^{p^2} + \ a_2 x^{p^3} + \ldots + \ a_{n-2}x^{p^{n-1}} + \ a_{n-1}x^{p^n} =$$

$$= a_{n-1}x + a_0 x^p + \ a_1 x^{p^2} + \ a_2 x^{p^3} + \ldots + \ a_{n-2}x^{p^{n-1}}.$$

Таким образом,

$$a^p = (a_0, a_2, \ldots, a_{n-2}, a_{n-1})^p = (a_{n-1}, a_1, a_2 \ldots, a_{n-2}).$$

В нормальном базисе просто вычислить след элемента a, $a \in GF(2^n)$: значение $Tr(a) = a + a^2 + a^{2^2} + \ldots + a^{2^{n-1}}$ равно сумме по модулю два элементов бинарного вектора a. Действительно, рассмотрим матрицу, строками которой

являются элементы $a,\ a^2,\ a^{2^2},\ldots,\ a^{2^{n-1}}$. Её столбцами являются те же векторы. То есть j-ый столбец представляет собой циклический сдвиг на j позиций первого столбца. Следовательно, сумма строк матрицы есть нулевой вектор или вектор из элементов 1. Это соответствует представлению в нормальном базисе элементов 0 и 1 простого подполя $GF(2)$.

Пример 1.2.21 Вычислим $Tr(10110)$ элемента (10110) поля $FG(2^5)$.

$$\begin{aligned} a &= 10110 \\ a^2 &= 01101 \\ a^{2^2} &= 11010 \\ a^{2^3} &= 10101 \\ a^{2^4} &= 01011 \end{aligned}$$

Сумма строк равна $(1,1,1,1,1)$, то есть $TR(10110)=1$ в подполе $GF(2)$.

Вычисление следа в полиномиальном базисе несколько сложнее, но может быть ускорено за счёт предварительного вычисления так называемого вектора следа.

Составим матрицу, строками которой являются представленные в полиномиальном базисе элементы $x^0,\ x^1,\ x^2, x^{n-1},\ x^n,\ \ldots,\ x^{n-2}$.

Пример 1.2.22 В поле $FG(2^5)$, порождаемом неприводимым многочленом x^5+x^2+1, такая матрица имеет вид

$$\begin{aligned} x^0 &= 10000 \\ x^1 &= 01000 \\ a^2 &= 00100 \\ a^3 &= 00010 \\ a^4 &= 00001 \\ a^5 &= 10100 \\ a^6 &= 01010 \\ a^7 &= 00101 \\ a^8 &= 10110 \end{aligned}$$

Элементы $t_i,\ i=0,\ldots,n-1$ вектора следа $(t_0,\ldots,\ t_{n-1})$ образуются суммированием по модулю два элементов матрицы, расположенных по диагонали (слева направо и сверху вниз) в соседних векторах $a_i,\ a_{i+1},\ldots,a_{j+n-1}$.

Например, $t_0 = 1\oplus 1\oplus 1\oplus 1\oplus 1$. Вектор следа в целом следующий:

$$(t_0,t_1,t_2,t_3,t_4)=(1,0,0,1,0).$$

Значение функции следа для элемента a поля $GF(2^n)$ равно сумме (дизъюнкции) элементов вектора, получаемого как поразрядное произведение (конъюнкция) элементов вектора a и вектора следа.

Пример 1.2.23 Вычислим значение функции следа элемента $(x^0,x^1,x^2,x^3,x^4)=(01101)$.

Вычислим покомпонентное произведение вектора следа и элемента $a=(01101)$:

$$(t_0,t_1,t_2,t_3,t_4)\&(01101)=(10010)\&(01101)=(00000).$$

Сумма элементов полученного вектора равна 0. След элемента (11000) является суммой элементов вектора (10010)&(11100) = (10000), т.е. $Tr(11100)=1$.

Умножение в полиномиальном базисе выполняется, как описано в предыдущем параграфе, например, умножением многочленов над полем Галуа и последующим приведением по модулю неприводимого многочлена, порождающего этот базис.

Умножение в нормальном базисе выполняется по правилам умножения многочленов

$$q(x) \cdot s(x) = \sum_{i=0}^{n-1} \sum_{j=0}^{n-1} a_i b_j x^{p^i} x^{p^j},$$

где $q(x) = a_0 x + a_1 x^p + \ldots + a_{n-1} x^{p^{n-1}}$, $s(x) = b_0 x + b_1 x^p + \ldots + b_{n-1} x^{p^{n-1}}$, при этом приведения результата по модулю неприводимого многочлена не требуется. В то же время слагаемые

$$c_{i,j} = a_i b_j x^{p^i} x^{p^j}$$

в указанной двойной сумме в общем случае не являются элементами нормального множества, так как должны представляться как линейные комбинации элементов базиса. Это усложняет операцию умножения в нормальном базисе по сравнению с операцией умножения в полиномиальном базисе. Недостатком последней является необходимость приведения результата умножения по модулю неприводимого многочлена. Иногда используют оба базиса, предусматривая перевод представлений элементов поля из одного базиса в другой. [5]. В случае применения полиномиального базиса используют базисы, порождаемые степенями корня примитивного многочлена с малым числом слагаемых (называемого «малочленом», английский термин - fewnomial). В этом случае возможно ускорение приведения результата умножения многочленов над полем GF(p) применением рассмотренного выше алгоритма. Существенную роль имеет также оптимизация умножения в полиномиальном базисе [6] многочленов над полем $GF(2)$ как первого этапа умножения в поле Галуа.

Проверка нормального множества на базисность. Переход от нормального базиса к стандартному. В стандартном базисе $S = \{1, x, x^2, \ldots, x^{k-1}\}$ любой многочлен из $GF(2^k)$ выражается некоторой линейной комбинацией элементов множества S с коэффициентами из $\{0, 1\}$. Таким образом, любой многочлен из $GF(2^k)$ в базисе S задается набором $t_1, t_2, \ldots t_n$ целых неотрицательных чисел, представляющих в совокупности все коэффициенты многочлена (см. приложение 1). Покажем, как определять такие наборы для элементов множества N.

Для многочлена x соответствующий набор имеет вид $1, 0, 0, \ldots, 0$. Пусть $t_1, t_2, \ldots t_n$ - набор чисел, задающих x^{2^i}. Рассмотрим элементарные многочлены $t_j(x), \quad j = 1, 2, \ldots, n$, задаваемые числами $t_j, \quad j = 1, 2, \ldots, n$. Тогда для многочлена $g(x) = \sum_{j=1}^{n} t_j x^{s(j-1)}$ имеем $x^{2^i} \equiv g(x) \bmod p(x)$. Умножив многочлен

[5] Методы умножения, возведения в степень и инвертирования в нормальных базисах рассматриваются в четвертой главе.

[6] Изложению и анализу этих методов посвящена третья глава.

$g(x)$ на себя и взяв остаток от деления результата на многочлен $p(x)$, получим набор чисел $t'_1, t'_2, \dots t'_n$ задающих многочлен $x^{2^{i+1}}$ в стандартном базисе.

Рассмотрим таблицу T с k строчками и n столбцами, $(i+1)$-я строчка которой, читаемая справа налево, совпадает с набором чисел $t_1, t_2, \dots t_n$, задающих многочлен x^{2^i} в стандартном базисе $i = 0, 1, \dots, k-1$.

Нетрудно видеть, что множество N является базисом в точности тогда, когда матрица из нулей и единиц соответствующая таблице T невырождена. Этот факт можно проверить с использованием алгоритма NONSIGN-MATRIX, описанного в Приложении 1.

Таблица T задает переход от нормального базиса к стандартному.

Пример 1.2.24 Пусть $k = 3, \quad s = 2$. Рассмотрим многочлен $p(x) = x^3 + x^2 + 1$. Этот многочлен неприводим. Далее,

$$x \equiv x \pmod{p(x)},$$

$$x^2 \equiv x^2 \pmod{p(x)},$$

$$x^4 = (x+1)p(x) + x^2 + x + 1 \equiv x^2 + x + 1 \pmod{p(x)}.$$

Тогда матрица перехода от нормального базиса к стандартному имеет вид

$$M = \begin{matrix} 0 & 1 & 0 \\ 1 & 0 & 0 \\ 1 & 1 & 1. \end{matrix}$$

Матрица M невырождена. Поэтому многочлены x, x^2, x^4 образуют нормальный базис. Рассмотрим многочлен $x^2 + x^4$, заданный в нормальном базисе. Тогда имеет место равенство $v = (0, 1, 1)$. Перемножая вектор v на матрицу M, получим вектор $v' = (0, 1, 1)$. Поэтому в стандартном базисе многочлен $x^2 + x^4$ представляется в виде $1 + x$.

Этот пример можно проследить, если при $s = 2$ преобразовать матрицу M к таблице T

$$T = \begin{matrix} 0 & 2 \\ 1 & 0 \\ 1 & 3, \end{matrix}$$

а вектор v к $(2, 1)$. Результатом умножения вектор-строки v на таблицу T, будет $(2, 1)$, что соответствует многочлену $x + 1$.

Переход от стандартного базиса к нормальному. Пусть таблица T задает невырожденную матрицу из нулей и единиц M перехода от нормального базиса к стандартному, т.е. нормальное множество является базисом. Используя алгоритм INV-MATRIX (см. Приложение 1), в этом случае построим таблицу T', задающую бинарную матрицу обратную к M (M^{-1}). Рассмотрим набор целых неотрицательных чисел $t'_1, t'_2, \dots t'_n$, который задает многочлен $f(x), \quad f(x) = \sum_{i=1}^{k} a_i x^{i-1}, \quad a_i \in \{0, 1\}$ в стандартном базисе. Тогда произведение вектор-строки u на таблицу T' задает этот же многочлен, но уже в стандартном базисе.

Пример 1.2.25 Пусть число $k = 3$, многочлен $p(x)$ и матрица M такие же, как и в предыдущем примере. Для матрицы M^{-1} обратной к M, имеем

$$\begin{matrix} 0 & 1 & 0 \\ 1 & 0 & 0 \\ 1 & 1 & 1. \end{matrix}$$

Найдем, например, представление многочлена $x+1$ в нормальном базисе. Для этого перемножим вектор-строку $v = (0, 1, 1)$ на матрицу M^{-1}. Получим строку $(0, 1, 1)$, соответствующую многочлену $x^2 + x^4$.

Если перемножать вектор $(2, 1)$, соответствующий v при $s = 2$ на таблицу T, задающую матрицу M, то получается вектор $(2, 1)$, который в нормальном базисе задает многочлен $x^2 + x^4$.

Оптимальные и гауссовы нормальные базисы[7]. Для оптимизации времени умножения или схемной реализации умножения в нормальных базисах используют оптимальные или близкие к ним гауссовы нормальные базисы. Согласно [138], сложностью C_B произвольного нормального базиса

$$B = \{x, x^q, x^{q_2}, \ldots, x^{q^{n-1}}\}$$

называется число ненулевых элементов в матрице T, i-я строка которой есть вектор коэффициентов элемента xx^{q^i} поля $GF(q^n)$ относительно базиса B, то есть

$$xx^{q^i} = \sum_{j=0}^{n-1} t_{i,j} x^{q^j}.$$

Это определение мотивируется следующим алгоритмом умножения в нормальном базисе B (алгоритмом Massey-Omura, см.,например, [109]):

пусть

$$\xi = \sum_{i=0}^{n-1} x_i x^{q^i}, \quad \zeta = \sum_{j=0}^{n-1} y_j x^{q^j},$$

произвольные элементы поля $GF(q^n)$, разложенные по нормальному базису B, тогда их произведение можно вычислить по формуле:

$$\pi = \xi\zeta = \sum_{i,j=0}^{n-1} x_i y_j x^{q^j+q^i} = \sum_{i,j=0}^{n-1} x_i y_j x^{(q^{i-j}+1)q^j},$$

где разность $i - j$ вычисляется по модулю n, а так как

$$x^{(q^{i-j}+1)q^j} = \left(x^{q^{i-j}+1}\right)^{q^j} = \left(\sum_{k=0}^{n-1} t_{i-j,k} x^{q^k}\right)^{q^j} =$$

$$= \sum_{k=0}^{n-1} t_{i-j,k} x^{q^{k+j}} = \sum_{m=0}^{n-1} t_{i-j,m-j} x^{q^m},$$

[7] Изучению оптимальных и гауссовых нормальных базисов посвящена вторая часть второй главы.

где разность $m-j$ и сумма $k+j$ тоже вычисляются по модулю n, то

$$\pi = \sum_{m=0}^{n-1} p_m x^{q^m},$$

где

$$p_m = \sum_{i,j=0}^{n-1} t_{i-j,m-j} x_i y_j$$

некоторая билинейная форма над полем $GF(q)$.

Так как при возведении элементов ξ, ζ в степень q происходит циклический сдвиг переменных в каждом из векторов $x_i, i = 1, \ldots, n$ и $y_i, i = 1, \ldots, n$ на одну позицию вправо, а $\pi^q = \xi^q \zeta^q$, то координаты элемента π^q вычисляются по формулам $P_i = p_i(S(x), S(y))$. Но при возведении элемента π в степень q происходит такой же циклический сдвиг координат , т.е. координата p_i переходит в координату $p_{i+1 \bmod n}$, значит $p_{i-1 \bmod n}(x,y) = p_i(S(x), S(y)), i = 0, \ldots n-1$ откуда следует, что $p_{i-k \bmod n}(x,y) = p_i(S^k(x), S^k(y)), i = 0, \ldots n-1$ т.е. все остальные формы формы получаются из формы p_0 по формуле $p_{m \bmod n}(x,y) = p_0(S^{n-m}(x), S^{n-m}(y)), k = 1, \ldots, n-1$ где S^{n-m} – операция циклического сдвига координат вектора вправо на $n-m$ позиций, или, что равносильно, влево на m позиций. Этот сдвиг можно явно определить формулой

$$S^{n-m}(x_0, \ldots, x_{n-1}) = (x_m, \ldots, x_{n-1}, x_0, \ldots x_{m-1}) = (x_{i+m \bmod n}, i = 0, \ldots, n-1).$$

Определив матрицу A равенствами $a_{i,j} = t_{i-j,-j}$, где $i-j$ и $-j$ вычисляются по модулю n, замечаем, что предыдущую формулу можно переписать в виде

$$p_m = \sum_{i,j=0}^{n-1} t_{i-j,m-j} x_i y_j = p_0(S^{n-m}(x), S^{n-m}(y)),$$

где

$$p_0(x,y) = A(x,y) = \sum_{i,j=0}^{n-1} a_{i,j} x_i y_j.$$

В отличие от T матрица A симметрическая, но число ее ненулевых элементов, а также их сумма такие же, как и у матрицы T. Для вычисления билинейной формы $A(x,y)$ достаточно выполнить $2C_B + n - 1$ сложений и умножений в поле $GF(q)$. Если пренебречь временем выполнения циклических сдвигов, то сложность выполнения умножения над нормальным базисом поля $GF(q^n)$ оценивается сверху как $n(2C_B + n - 1)$ операций в поле $GF(q)$, что видно из следующей формулы:

$$\xi\zeta = A(\xi,\zeta)x + A(\xi^{q^{n-1}}, \zeta^{q^{n-1}})x^q + A(\xi^{q^{n-2}}, \zeta^{q^{n-2}})x^{q^2} + \ldots + A(\xi^q, \zeta^q)x^{q^{n-1}}.$$

Таким образом, сложность умножения зависит только от количества ненулевых элементов C_B в матрице A.

Упражнение 1.2.1 Докажите, что в случае $q = 2$ сложность умножения оценивается как $n(C_B + n - 1)$.

Матрица A («таблица умножения» в базисе B) однозначно определяет операцию умножения в рассматриваемом поле.

О сложности нормальных базисов известно следующее ([138]).

Теорема 1.2.22 *Для любого нормального базиса B поля $GF(q^n)$ его сложность C_B не меньше $2n-1$. Более того, если $q = 2$, то сложность – нечетна.*

Доказательство. Напомним, что в матрице T произвольная i-я строка есть просто вектор коэффициентов элемента xx^{q^i} поля $GF(q^n)$ относительно базиса B. Таким образом, сумма всех строк матрицы T есть вектор коэффициентов следующего элемента $x(x + x^q + \ldots + x^{q^{n-1}}) = xTr(x)$, то есть $(Tr(x), 0, \ldots, 0)$. Заметим, что след не равен нулю, так как элементы базиса B линейно независимы. С другой стороны, строки T тоже линейно независимы, таким образом, вектор $x, xx^q, \ldots, xx^{q^{n-1}}$ тоже является базисом в $GF(q^n)$, таким образом, столбцы в матрице тоже линейно независимы, в частности нет нулевых строк в матрице. Но суммы элементов в каждом столбце, кроме первого, равны нулю. Таким образом, в каждом столбце, кроме первого, стоит, как минимум, по два ненулевых элемента. А в первом должен стоять, как минимум, один ненулевой элемент, чтобы их сумма была ненулевой. Поэтому $C_B \geq 2n-1$. В частном случае $q = 2$ количество единиц в первом столбце – нечетно, а во всех остальных – четно. Итак, установлено, что для любого нормального B его сложность не меньше $2n - 1$.

Нормальные базисы, для которых достигается эта граница, называют *оптимальными*.

В [80] были приведены примеры нормальных базисов B, у которых функция сложности C_B является линейной. Стандартный алгоритм умножения для таких базисов имеет квадратичную оценку сложности. Эти базисы B^α, получившие название *гауссовых нормальных базисов* (GNB),

1.2.9 Линейные рекуррентные последовательности (ЛРП)

ЛРП и линейные регистры сдвига (ЛРС). Будем рассматривать бесконечные последовательности над простым конечным полем F_p

$$< u >= u_0, u_1, \ldots, u_n, \ldots,$$

то есть функции $u : N_0 \to F_p$ на множестве N_0 целых неотрицательных чисел, принимающие значения в поле F_p.

Последовательность $< u >$ называется *линейной рекуррентной последовательностью* (ЛРП) *порядка* k *над полем* F_p, если существуют константы $a_0, \ldots, a_{k-1} \in F_p$ такие, что

$$u_{n+k} = \sum_{j=0}^{k-1} a_j \cdot u_{n+j} + a, \ n \geq 0. \tag{1.17}$$

Замечание. Ниже будем изучать только *однородные* ЛРП, определяемые рекуррентным соотношением вида

$$u_{n+k} = \sum_{j=0}^{k-1} a_j \cdot u_{n+j}, \ n \geq 0,$$

то есть соотношением (2.1), в котором свободный член $a = 0$.

Это равенство, выражающее зависмость между членами последовательности, называется *законим рекурсии,* а опредляющий этот закон многочлен

$$f(x) = x^k - \sum_{j=0}^{k-1} a_j \cdot x^j \tag{1.18}$$

называется *характеристическим многочленом* ЛРП. Вектор

$$\mathbf{u}_0 = (u_0, \ldots, u_{k-1})$$

называется *начальным вектором* ЛРП.

Периодом ЛРП $< u >$ называется наименьшее натуральное число t такое, что при некотором неотрицательном числе η для всех $i \geq 0$ выполняется равенство

$$u_{\eta+i+t} = u_{\eta+i}.$$

Если η может быть равно 0, то последовательность называется *строго периодической.* Последовательность строго периодическая тогда и только тогда, когда коэффициент a_0 ее характеристического многочлена не равен 0. В этом случае многочлен называется несингулярным. (Если $a_0 = 0$, то характеристический многочлен называется *сингулярным*). Линейные рекуррентные последовательность удобно изучать (и практически использовать) как последовательности выходных сигналов *линейных регстров сдвига* (ЛРС).

ЛРС, формирующий ЛРП порядка k над полем F_p представляется как автономный структурный автомат

$$V = (\emptyset, F_p^k, F_p, \varphi, \psi),$$

представляемый функциональной схемой с памятью. Функциональная схема содержит k элементов задержки

$$g_0, \ g_1, \ \ldots, \ g_{k-1},$$

с начальными состояниями

$$\mathbf{q}(\mathbf{0}) = (q_0(0) = u_0,\ q_1(0) = u_1,\ \ldots,\ q_{k-1}(0) = u_{k-1}).$$

Функционирование автомата описывается следующей канонической системой:

$$q_{k-1}(t+1) = \sum_{i=0}^{k-1} a_i \cdot q_i(t),$$

$$q_i(t+1) = q_{i+1}(t),\ i = (0, k-2),$$

$$y(t) = q_0(t).$$

Не трудно видеть, что последовательность $< y >$ выходных сигналов такого автомата в точности совпадает с ЛРП $< u >$ с начальным вектором, совпадающим с вектором начальных состояний автомата, то есть ЛРС. Из автоматной интерпретации ЛРП порядка k следует, что ее период не превышает $p^k - 1$, где p – порядок поля P. Действительно, автомат имеет $p^k - 1$ ненулевых состояний и в процессе функционирования через не более чем $p^k - 1$ моментов времени автомат перейдет в одно из состояний, в котором он уже находился.

Если при этом окажется, что период равен $p^k - 1$, то ЛРП порядка k называется *последовательностью максимального периода*, или просто *максимальной* ЛРП.

Автоматная интерпретация подсказывает понятие *состояния ЛРП* как вектора $\mathbf{u}_n = (u_n, u_{n+1}, \ldots, n_{n+k-1})$, определяющего состояние $\mathbf{q}(\mathbf{n})$ структурного автомата в момент n дискретного времени. При этом начальный вектор $\mathbf{u}_0 = (u_0, \ldots u_{k-1})$ (он же вектор начального состояния $\mathbf{q}(\mathbf{0})$ конечного автомата) ЛРП рассматривается как ее начальное сотояние.

Не трудно видеть, что векторы $\mathbf{u}_{n+1}$ и $\mathbf{u}_n$ соседних состояний ЛРП как векторы соседних состояний конечного автомата удовлетворяют матричному уравнению

$$\mathbf{u}_{n+1} = \mathbf{u}_n A,$$

где A есть матрица над полем F_p размера $k \times k$ следующего вида

$$A = \begin{pmatrix} 0 & 0 & 0 & \ldots & 0 & a_0 \\ 1 & 0 & 0 & \ldots & 0 & a_1 \\ 0 & 1 & 0 & \ldots & 0 & a_2 \\ \cdot & \cdot & \cdot & \ldots & \cdot & \cdot \\ 0 & 0 & 0 & \ldots & 1 & a_{k-1} \end{pmatrix}$$

Лемма 1.2.1 *Для векторов состояний* $\mathbf{u}_0, \mathbf{u}_1, \ldots, \mathbf{u}_n, \ldots$ *ЛРП справедливо равенство*

$$\mathbf{u}_n = \mathbf{u}_0 A^n,\ n = 0, 1, \ldots\ .$$

Если характеристический многочлен несингулярный, то матрица A обратима. Можно показать, что обратной матрицей является матрица

$$A^{-1} = \begin{pmatrix} a_{k-1}^* & 1 & 0 & 0 & 0 & \dots & 0 & 0 \\ a_{k-2}^* & 0 & 1 & 0 & 0 & \dots & 0 & 0 \\ \cdot & \cdot & \cdot & \cdot & \cdot & \dots & \cdot & \cdot \\ a_2^* & 0 & 0 & 0 & 0 & \dots & 1 & 0 \\ a_1^* & 0 & 0 & 0 & 0 & \dots & 0 & 1 \\ a_0^* & 0 & 0 & 0 & 0 & \dots & 0 & 0 \end{pmatrix}$$

Здесь a_i^* – коэффициенты *возвратного многочлена*

$$f^*(X) = X^n \times a_0^{-1} \times f(\frac{1}{X}).$$

Пример 1.2.26 Возьмем многочлен $f(X) = X^3 + X + 1$ над полем F_2, тогда $f^*(X) = X^3 + X^2 + 1$. Матрицы A и A^{-1} имеют вид

$$A = \begin{pmatrix} 0 & 0 & 1 \\ 1 & 0 & 1 \\ 0 & 1 & 0 \end{pmatrix}, \qquad A^{-1} = \begin{pmatrix} 1 & 1 & 0 \\ 0 & 0 & 1 \\ 1 & 0 & 0 \end{pmatrix}.$$

Все обратимые матрицы размере $k \times k$ над полем F_p образуют *общую линейную группу* $GL(k, F_p)$.

Теорема 1.2.23 *Период ЛРП делит порядок матрицы A, рассматриваемой как элемент группы $GL(k, F_p)$.*

По лемме 2.1 начальное состояние $\mathbf{u}_0$ ЛРП можно вычислить, если известно состояние $\mathbf{u}_n$:

$$\mathbf{u}_0 = \mathbf{u}_n (A^n)^{-1}.$$

Формула общего члена ЛРП

Выведем формулу общего члена ЛРП, заданной характеристическим многочленом $f(x)$ степени k. При этом будем использовать функцию $tr_P^Q : Q \to P$ следа из расширения $Q = GF(p^k)$ поля $P = GF(p) = F_p$ в простое поле $P = F_p$:

$$tr_P^Q(x) = x + x^p + x^{p^2} + \dots + x^{p^{k-1}}.$$

Сокращенно эту функцию будем обозначать tr. Свойства операций конечного поля влекут следующие свойства функции след:

$$tr(a \cdot x + b \cdot y) = a \cdot tr(x) + b \cdot tr(y), \ a, b \in P;$$

$$tr(x) = tr(x^{p^k}) = (tr(x))^{p^k}.$$

Лемма 1.2.2 *Для любого ненулевого $\alpha \in Q$ и любого $b \in P$ число N_b решений уравнения $tr(\alpha \cdot x) = b$ равно p^{k-1}*

Доказательство: $N_b \leq p^{k-1}$, т.к. p^{k-1} – степень уравнения. Но $\sum_{b \in P} N_b = p^k$, так как при любом x $tr(\alpha \cdot x) \in P$. Отсюда $N_b = p^{k-1}$.

Теорема 1.2.24 *Для ЛРП $< u >$, определяемой неприводимым характеристическим многочленом (2.2) с корнем λ в поле Q существует единственная константа $\alpha \in Q$ такая, что*

$$u_n = tr(\alpha \cdot \lambda^n), \ n \geq 0. \tag{1.19}$$

Доказательство. Прежде всего заметим, что последовательность (2.3) является линейной рекуррентной последовательностью, определяемой характеристическим многочленом (2.2)

$$\sum_{j=0}^{k-1} a_j \cdot u_{n+j} = \sum_{j=0}^{k-1} a_j \cdot tr(\alpha \cdot \lambda^{n+j}) = tr\left(\alpha \cdot \lambda^n \cdot \sum_{j=1}^{k-1} a_j \cdot \lambda^j\right) =$$
$$tr(\alpha \cdot \lambda^n \cdot \lambda^k) = tr(\alpha \cdot \lambda^{n+k}) = u(n+k).$$

Здесь $\sum_{j=1}^{k-1} a_j \cdot \lambda^j = \lambda^k$, так как λ есть корень многочлена (2.2)

Далее покажем, что различным константам соответствуют разные последовательности.

Заметим, что векторы

$$\mathbf{u}_{\lambda^0} = (tr(1\lambda^0), tr(1\lambda^1), \ldots, tr(1\lambda^{k-1})),$$
$$\mathbf{u}_{\lambda^1} = (tr(\lambda\lambda^0), tr(\lambda\lambda^1), \ldots, tr(\lambda\lambda^{k-1})),$$
$$\ldots$$
$$\mathbf{u}_{\lambda^{k-1}} = (tr((\lambda^{k-1}\lambda^0), tr(\lambda^{k-1}\lambda^1), \ldots, tr(\lambda^{k-1}\lambda^{k-1})),$$

определяют начальные состояния

$$\mathbf{u}_{\lambda^0}, \mathbf{u}_{\lambda^1}, \ldots \mathbf{u}_{\lambda^{k-1}}$$

последовательностей, соответствующих линейно независимым значениям

$$1, \lambda, \ldots, \lambda^{k-1} \tag{1.20}$$

константы α.

Множество этих начальных состояний также линейно независимо. Допустим, что это не так, тогда покажем, что линейно зависимо множество констант (2.4), составляющих полиномиальный базис поля Q.

Допустим, что некоторая линейная комбинация указанных начальных состояний равна нулю:

$$c_0\mathbf{u}_1 + c_1\mathbf{u}_\lambda + \ldots + c_{k-1}\mathbf{u}_{\lambda^{k-1}} = 0.$$

Тогда линейной комбинации базисных констант

$$c_0 + c_1\lambda + \cdots + c_{k-1}\lambda^{k-1}$$

соответствует нулевое начальное состояние последовательности и, следовательно, нулевая последовательность:

$$tr((c_0 + c_1\lambda + \cdots + c_{k-1}\lambda^{k-1}) \cdot \lambda^i) = 0, i = 0, 1, \ldots.$$

Если λ есть корень примитивного, многочлена, то уравнение

$$tr((c_0 + c_1\lambda + \cdots + c_{k-1}\lambda^{k-1}) \cdot x) = 0, \tag{1.21}$$

удовлетворяется при любом x, то есть имеет $p^k > p^{k-1}$ корней (0 и $p^k - 1$ степеней корня λ), что при

$$((c_0 + a_1\lambda + \cdots + c_{k-1}\lambda^{k-1}) \neq 0$$

противоречит лемме 2.2.

Если λ есть корень многочлена, не являющегося примитивным, представим через него примитивный элемент θ поля Q :

$$\theta = b_0 + b_1\lambda + \cdots + b_{k-1}\lambda$$

и образуем последовательность

$$tr((c_0 + c_1\lambda + \cdots + c_{k-1}\lambda^{k-1}) \cdot \theta^i), i = 0, 1, \ldots \ .$$

Используя свойства линейности функции tr, получим

$$\begin{array}{l} tr(c_0 + c_1\lambda + \cdots + c_{k-1}\lambda^{k-1}) \cdot \theta^i) = \\ = tr((c_0 + c_1\lambda + \cdots + c_{k-1}\lambda^{k-1}) \cdot (\sum_{j=0}^{k-1} b_j\lambda^j)^i) = \\ = (tr(c_0 + c_1\lambda + \cdots + c_{k-1}\lambda^{k-1}))^{-(i-1)} \cdot \\ \cdot(\sum_{j=0}^{k-1} b_j tr(c_0 + c_1\lambda + \cdots + c_{k-1}\lambda^{k-1})\lambda^j))^i = 0. \end{array}$$

Как видим, и в этом случае уравнение (2.5) удовлетворяется при любом x (при x, равном 0 или любой степени корня θ примитивного многочлена), что противоречит линейной независимости элементов полиномиального базиса поля Q.

Таким образом, имеется взаимно однозначное соответствие между множеством возможных значений констант α и начальных состояний последовательностей.

Пример 1.2.27 Пусть $k = 2$, λ – корень многочлена X^2+X+1 над полем $GF(2)$. Константам 1 и λ соответствуют базисные начальные состояния

$$\mathbf{u}_1 = tr(1\lambda^0), tr(1\lambda^1) = (tr(1), tr(\lambda)) = (1+1, \lambda+\lambda^2) = (0, 1);$$

$$\mathbf{u}_\lambda = (tr(\lambda)\lambda^0), tr(\lambda\lambda^1) = (tr(\lambda), tr(\lambda^2)) = (1, 0),$$

Отсюда получаем, что константам (0,0) и (1,1) соответствуют начальные состояния

$$\mathbf{u}_0 = (0, 0) \text{ и } \mathbf{u}_1 = (1, 1).$$

Упражнение 1.2.2 Сформулируйте алгоритм вычисления начального вектора ЛРП, определяемой известным неприводимым многочленом, по ее отрезку из k элементов.

Указание. Сначала следует найти элемент

$$\alpha = \alpha_0 + \alpha_1\lambda + \cdots + \alpha_{k-1}\lambda^{k-1},$$

упоминаемый в формулировке теоремы. Для этого с использованием заданных k элементов $u_n, u_{n+1}, \ldots, u_{k-1}$ составить и решить систему из k линейных относительно коэффициентов этого элемента уравнений

$$u_{n+j} = tr\left((\alpha_0 + \alpha_1\lambda + \cdots + \alpha_{k-1}\lambda^{k-1})\lambda^{n+j}\right),\ j = 0, 1 \ldots, k-1.$$

После того, как элемент α найден, k начальных элементов последовательности вычисляются по формуле

$$u_j = tr(\alpha_0 + \alpha_1\lambda + \cdots + \alpha_{k-1}\lambda^{k-1})\lambda^j,\ j = 0, 1 \ldots, k-1.$$

Пример 1.2.28 Пусть $k = 2,\ \lambda$ есть корень многочлена $X^2 + X + 1$. Даны элементы $u_2 = 0,\ u_3 = 1$ последовательности

$$< u >= u_0, u_1, u_2, u_3, \ldots,$$

характеристическим многочленом которой является $X^2 + X + 1$.

Составим два уравнения

$$u_2 = tr((\alpha_0 + \alpha_1\lambda)\lambda^2) = \alpha_0 + \alpha_1 tr(\lambda^3) = \alpha_0 + \alpha_1 \cdot 0 = 0,$$

$$u_3 = tr((\alpha_0 + \alpha_1\lambda)\lambda^3) = \alpha_0 + \alpha_1 tr(\lambda^4) = \alpha_0 + \alpha_1 tr(\lambda) = \alpha_0 + \alpha_1 \cdot 1 = 1.$$

Из этих уравнений получим $\alpha_0 = 0, \alpha_1 = 1$, то есть $\alpha = (0, 1)$.

Теперь можно определить u_0 и u_1 :

$$u_0 = tr(a\lambda^0) = tr(0 + \lambda) = \lambda + \lambda^2 = \lambda + \lambda + 1 = 1,$$

$$u_1 = tr(a\lambda^1) = tr((0 + \lambda)\lambda) = tr(\lambda^2) = tr(\lambda) = \lambda + \lambda^2 = 1.$$

Следствие 1.2.20 *Период рекуррентной последовательности равен порядку корня λ ее характеристического многочлена и она является последовательностью максимального периода тогда и только тогда, когда ее характеристический многочлен примитивен.*

Минимальный и аннулирующий многочлены ЛРП. ЛРП из элементов поля P, заданная некоторым рекуррентным соотношением, может удовлетворять и многим другим рекуррентным соотношениям. Так если t есть период ЛРП $< u >= u_0, u_1, \ldots$, то выполняются рекуррентные соотношения $u_{n+t} = u_n,\ n = 0, 1, \ldots,\ u_{n+2t} = u_n,\ n = 0, 1, \ldots$ и т.д. Подобные соотношения связаны между собой, как это определяет следующая теорема.

Теорема 1.2.25 *Пусть* $< u >= u_0, u_1, \ldots$ *– ЛРП над полем* P*. Тогда существует однозначно определяемый нормированный многочлен* $m(x)$ *над полем* P *такой, что любой нормированный многочлен* $f(x)$ *положительной степени над* P *является характеристическим многочленом этой последовательности* $< u >$ *тогда и только тогда, когда* $f(x)$ *делится на* $m(x)$*.*

Определяемый этой теоремой многочлен $m(x)$ является, очевидно характеристический многочленом ЛРП $< u >$, имеющим наименьшую степень, он называется *минимальным* многочленом ЛРП, степень минимального многочлена определяет *линейную сложность* ЛРП.

Очевидно, что минимальный многочлен ЛРП является неприводимым многочленом над полем P. Если при этом минимальный многочлен примитивен, то ЛПР имеет максимальный период.

Линейной сложностью $L(u^n)$ конечной последовательности $< u^n >= u_0, u_1, \ldots, u_{n-1}$ называется сложность бесконечной ЛРП

$$< u >= u_0, u_1, \ldots u_{n-1}, u_n \ldots,$$

имеющей минимальную линейную сложность.

Алгебра степенных рядов. Произвольной последовательности $u_0, u_1, \ldots, u_n, \ldots$ из элементов поля P свяжем формальный степенной ряд от формальной переменной x.

$$G(x) = u_0 + u_1 x + u_2 x^2 + \cdots + u_n x^n + \cdots = \sum_{n=0}^{\infty} u_n x^n. \tag{1.22}$$

Степенной ряд последовательности иногда называют производящей функцией этой последовательности. Однако в данном случае ни область определения, на область значений "функции"не могут быть указаны. Рассматриваемая конструкция является лишь формальным символом, отражающим линейный порядок элементов последовательности. Элементы последовательности выступают в качестве коэффициентов формального степенного ряда.

Два формальных степенных ряда

$$B(x) = \textstyle\sum_{n=0}^{\infty} b_n x^n \text{ и } \sum_{n=0}^{\infty} c_n x^n$$

считаются равными, если $b_n = c_n,\ n = 0, 1, \ldots$.

Использование формальных степенных рядов позволяет рассматривать многочлен над полем P

$$p(x) = p_0 + p_1 x + \cdots + p_k x^k$$

также как формальный степенной ряд

$$p(x) = p_0 + p_1 x + \cdots + p_k x^k + 0 \cdot x^{k+1} + 0 \cdot x^{k+2} + \cdots.$$

На множестве степенных рядов определяют операции сложения и умножения по правилам, аналогичным правилам сложения и умножения многочленов:

$$B(x) + C(x) = \sum_{n=0}^{\infty} (b_n + c_n)x^n,$$

$$B(x)C(x) = \textstyle\sum_{n=0}^{\infty}(d_n)x^n, \text{ где } \ d_n = \sum_{k=0}^{n} b_k c_{n-k}, \ n = 0, 1, \ldots.$$

Если $B(x)$ и $C(x)$ – многочлены, то эти операции имеют обычный смысл сложения и умножения многочленов. В то же время, как видно, можно складывать и перемножать обычные многочлены и формальные степенные ряды смешанным образом(один операнд – многочлен, а другой – формальный степенной ряд).

Множество формальных степенных рядов с двумя рассмотренными операциями образует кольцо. Аддитивной единицей кольца является формальный ряд, соответствующий последовательности

$$< 0 >= 0, 0, \ldots$$

из аддитивных единиц 0 поля P, а мультипликативной единицей – формальный ряд, соответствующий последовательности

$$< 1 >= 1, 0, 0 \ldots,$$

начинающейся мультипликативной единицей 1 поля P и продолжающийся аддитивными единицами этого поля.

Теорема 1.2.26 *Формальный степенной ряд*

$$B(x) = \sum_{n=0}^{\infty} b_n x^n$$

имеет обратный относительно операции умножения элемент $B(x)^{-1}$ *тогда и только тогда, когда* $b_0 \neq 0$.

Доказательство. Пусть $C(x) = B(x)^{-1}$, то есть

$$B(x)C(x) =< 1 > .$$

в кольце степенных рядов. Тогда коэффициенты $c_0, c_1, \ldots,$ $b_0, b_1, \ldots$ степенных рядов $C(x)$ и $B(x)$ удовлетворяют соотношениям

$$\begin{aligned} &d_0 = b_0 c_0 = 1, \\ &d_1 = b_0 c_1 + b_1 c_0 = 0. \\ &\ldots \\ &d_n = b_0 + c_n + b_1 c_{n-1} + \ldots + b_n c_0 = 0. \ldots \end{aligned}$$

Из первого соотношения следует, что $b_0 \neq 0$, и c_0 однозначно определяется как b_0^{-1} в поле P. Остальные коэффициенты $c_i, \ i = 1, 2, \ldots$ при этом определяются однозначно по рекурсивной схеме.

Если $B(x)$ имеет обратный элемент, то можно определить операцию деления $\frac{A(x)}{B(x)} = A(x)B(x)^{-1}$. Формально результат можно получить делением "углом."

Пусть $u_0, u_1, \ldots$ линейная последовательность k-го порядка над полем P, удовлетворяющая рекуррентному соотношению

$$u_{n+k} = a_{k-1}u_{n+k-1} + a_{k-2}u_{n+k-2} + \cdots + a_0u_n,\ n = 0, 1, \ldots.$$

Многочлен

$$f^*(x) = 1 - a_{k-1}x - a_{k-1}x^2 - \cdots - a_0x^k$$

над полем P называется *возвратным, или двойственным многочленом* этой последовательности. Характеристический многочлен

$$f(x) = x^k - a_{k-1}x^{k-1} - a_{k-1}^{k-1} - \cdots - a_0$$

и возвратный характеристический многочлен последовательности порядка k связаны соотношением

$$f^*(x) = x^k f(x^{-1}).$$

Отсюда

$$f(x) = x^k f^*(x^{-1}).$$

Еще одной операцией на множестве степенных рядов является операция T сдвига, сопоставляющая степенному ряду (2.?) степенной ряд

$$T(G(x)) = u_1 + u_2x + u_3x^2 + \cdots + u_{n+1}x^n + \cdots = \sum_{n=0}^{\infty} u_{n+1}x^n. \tag{1.23}$$

k-кратному применению этой операции соответствует оператор T^k :

$$T^k(G(x)) = u_k + u_{k_1}x + \cdots + u_{n+k}x^n + \cdots = \sum_{n=0}^{\infty} u_{n+k}x^n. \tag{1.24}$$

(при $k = 0$ $T^k(G(x)) = G(x)$). Многочлену $f(X) = \sum_{k=0}^{r} a_kx^k$ соответствует композиция операторов сдвига

$$f^T = \sum_{k=0}^{r} a_kT^k.$$

В результате ее применение к степенному ряду $G(x)$ получается степенной ряд

$$f^T(G(k)) = \sum_{k=0}^{r} T^k(a_kG(x)).$$

Многочлен $f(X)$ называется *аннулирующим многочленом* для степенного ряда $G(X)$, если

$$f^T(G(k)) = \sum_{k=0}^{r} T^k(G(x)) = 0.$$

Упражнение 1.2.3 *Покажите, что степенной ряд имеет аннулирующий многочлен тогда и только тогда, когда он является линейной рекуррентной последовательностью и что аннулирующим ее многочленом является ее характеристический многочлен.*

Алгоритм Берлекэмпа-Месси. Пусть задан отрезок ЛРП с неизвестным минимальным многочленом степени не более k, содержащий не менее $2k$ элементов. Приведём одну из модификаций алгоритма Берлекэмпа-Месси построения минимального многочлена $m(x)$

Если степень получаемого минимального многочлена равна k, то этот многочлен есть аннулирующий многочлен для этой последовательности.

Пусть $u_0, u_1, \ldots$ – последовательность над конечным полем P и $G(x) = \sum_{n=0}^{\infty} u_n x^n$ – представляющий эту последовательность формальный степенной ряд. Для $j = 0, 1, \ldots$ определим многочлены $g_j,\ h_j$ над полем P, целые числа m_j и элементы b_j из поля P следующим образом. Положим

$$g_0(x) = 1,\ h_0(x) = x,\ m_0 = 0,\ b_0 = u_0.$$

Далее для $j = (0, 2k-1)$ выполнить

$$1. g_{j+1}(x) = g_j(x) - b_j h_i(x);$$
$$2. h_{j+1}(x) = \begin{cases} b_j^{-1} x g_j(x), & \text{если } b_j \neq 0,\ m_j \geq 0, \\ x h_j(x) & \text{в противном случае;} \end{cases}$$
$$3. m_{j+1} = \begin{cases} -m_j, & \text{если } b_j \neq 0,\ m_j \geq 0, \\ m_j + 1 & \text{в противном случае;} \end{cases}$$
4. Присвоить b_{j+1} значение коэффициента при x^{j+1} формального ряда $g_{j+1}(x)G(x)$.

Замечание. Поскольку в вычислениях используются только первые $2k$ членов последовательности, то вместо формального ряда $G(x)$ можно использовать многочлен

$$G_{2k-1}(x) = \sum_{n=0}^{2k-1} u_n x^n.$$

Если $u_0, u_1, \ldots$ – ЛРП с минимальным многочленом степени k, то после выполнения указанных действий получим многочлен $g_{2k}(x)$, равный возвратному минимальному многочлену. Искомый минимальный многочлен в этом случае может быть получен как

$$m(x) = x^k g_{2k}(1/x).$$

Если же заранее известно лишь, что $\deg m(x) \leq k$, то минимальный многочлен определяется равенством

$$m(x) = x^r g_{2k}(1/x),$$

где $r = \lfloor k + 1/2 - m_{2k}/2 \rfloor$.

Пример 1.2.29 Пусть 8 членов ЛРП над полем $GF(3)$ порядка $k \leq 4$ образуют её начальный отрезок

$$0, 2, 1, 0, 1, 2, 1, 0,$$

тогда

$$G_7(x) = 2x + x^2 + x^4 + 2x^5 + x^6.$$

Работа алгоритма представлена в следующей таблице

j	$g_j(x)$	$h_j(x)$	m_j	b_j
0	1	x	0	0
1	1	x^2	1	2
2	$1 + x^2$	$2x$	-1	1
3	$1 + x + x^2$	$2x^2$	0	0
4	$1 + x + x^2$	$2x^3$	1	2
5	$1 + x + x^2 + 2x^3$	$2x + 2x^2 + 2x^3$	-1	2
6	$1 + x^3$	$2x^2 + 2x^3 + 2x^4$	0	1
7	$1 + x^2 + 2x^3 + x^4$	$x + x^4$	0	1
0	$1 + 2x + x^2 + 2x^3$		0	

В данном случае $r = \lfloor 4 + 1/2 - m_8/4 \rfloor = 4$. Поэтому

$$m(x) = x^4 + 2x^3 + x^2 + 2x.$$

Рекуррентное соотношение наименьшего порядка, которому удовлетворяет данная последовательность, имеет вид

$$u_{n+4} = u_{n+3} + 2u_{n+2} + u_{n+1}.\ n = 0, 1, \ldots.$$

Пример 1.2.30 Пусть первые 8 членов ЛРП над полем $GF(2)$ следующие:

$$1, 1, 0, 0, 1, 0, 1, 1.$$

Используем многочлен $G_7(x) = 1 + x + x^4 + x^6 + x^7$ над полем $GF(2)$ вместо формального степенного ряда $G(x)$ последовательности. Применим алгоритм Берлекэмпа–Мэсси, чтобы найти ЛРП наименьшего порядка, с указанными первыми элементами. Работу алгоритма представим в таблице:

j	$g_j(x)$	$h_j(x)$	m_j	b_j
0	1	x	0	1
1	$1 + x$	x	0	0
2	$1 + x$	x^2	1	1
3	$1 + x + x^2$	$x + x^2$	-1	1
4	1	$x^2 + x^3$	0	1
5	$1 + x^2 + x^3$	x	0	0
6	$1 + x^2 + x^3$	x^2	1	0
7	$1 + x^2 + x^3$	x^3	2	0
0	$1 + x^2 + x^3$		3	

В этом примере $r = \lfloor 4 + 1/2 - m_8/2 \rfloor = 3$ и, следовательно,

$$m(x) = x^3(1 + (1/x)^2 + (1/x)^3) = x^3 + x + 1.$$

Таким образом, заданные элементы образуют начальный отрезок ЛРП, удовлетворяющей рекуррентному соотношению

$$u_{n+3} = u_{n+1} + u_n, n = 0, 1, \ldots,$$

и не существует ЛРП меньшего порядка, имеющей тот же начальный отрезок. Не трудно проверить, что многочлен $x^3 + x + 1$ является аннулирующим для рекуррентной последовательности с указанным начальным отрезком:

$$(T^3 + T + 1)(1 + x^2 + x^3 + x^5 + x^6 + \cdots) = 0.$$

Пример 1.2.31 Построим ЛРП над полем $GF(2)$ наименьшего порядка, не превышающего 5, первые 10 членов которой образуют отрезок

$$0, 0, 1, 1, 0, 1, 1, 1, 0, 1.$$

Используем многочлен

$$G_{10} = x^2 + x^3 + x^5 + x^6 + x^7 + x^9,$$

представляющий указанный отрезок. Работа алгоритма Берлекэмпа-Месси представлена в таблице

j	$g_j(x)$	$h_j(x)$	m_j	b_j
0	1	x	0	0
1	1	x^2	1	0
2	1	x^3	2	1
3	$1 + x^3$	x	-2	1
4	$1 + x + x^3$	x^2	-1	1
5	$1 + x + x^2 + x^3$	x^3	0	1
6	$1 + x + x^2$	$x + x^2 + x^3 + x^4$	0	0
7	$1 + x + x^2$	$x^2 + x^3 + x^4 + x^5$	1	1
0	$1 + x + x^3 + x^4 + x^5$	$x + x^2 + x^3$	0	1
9	$1 + x^2 + x^4 + x^5$	$x^2 + x^3 + x^4$	0	1
10	$1 + x^3 + x^5$	$x + x^3 + x^5 + x^6$	0	0

В данном случае $r = \lfloor 5+1/2+m_{10}/2 \rfloor = \lfloor 5+1/2+0 \rfloor = 5$. Следовательно,

$$m(x) = x^5(1 + (x^{-1})^3 + (x^{-1})^5) = x^5 + x^2 + 1.$$

Указанный отрезок является начальным отрезком ЛРП, определяемой рекуррентным соотношением

$$u_{n+5} = u_{n+2} + u_n.$$

Многочлен $x^5 + x^2 + 1$ есть аннулирующий многочлен этой последовательности.

1.2.10 Минимальный (аннулирующий) многочлен ЛРП. Алгоритм Берлекемпа-Месси

Минимальный многочлен и линейная сложность ЛРП. ЛРП из элементов поля P, заданная некоторым рекуррентным соотношением, может

удовлетворять и многим другим рекуррентным соотношениям. Так если t есть период ЛРП $< u >= u_0, u_1, \ldots$, то выполняются рекуррентные соотношения $u_{n+t} = u_n,\ n = 0, 1, \ldots$, $u_{n+2t} = u_n,\ n = 0, 1, \ldots$ и т.д. Подобные соотношения связаны между собой, как это определяет следующая теорема.

Теорема 1.2.27 *Пусть $< u >= u_0, u_1, \ldots$ – ЛРП над полем P. Тогда существует однозначно определяемый нормированный многочлен $m(x)$ над полем P такой, что любой нормированный многочлен $f(x)$ положительной степени над P является характеристическим многочленом этой последовательности $< u >$ тогда и только тогда, когда $f(x)$ делится на $m(x)$.*

Определяемый этой теоремой многочлен $m(x)$ является, очевидно характеристический многочленом ЛРП $< u >$, имеющим наименьшую степень, он называется *минимальным* многочленом ЛРП, степень минимального многочлена определяет *линейную сложность* ЛРП.

Очевидно, что минимальный многочлен ЛРП является неприводимым многочленом над полем P. Если при этом минимальный многочлен примитивен, то ЛПР имеет максимальный период.

Линейной сложностью $L(u^n)$ конечной последовательности $< u^n >= u_0, u_1, \ldots, u_{n-1}$ называется сложность бесконечной ЛРП

$$< u >= u_0, u_1, \ldots u_{n-1}, u_n \ldots,$$

имеющей минимальную линейную сложность.

Алгебра степенных рядов. Произвольной последовательности $u_0, u_1, \ldots, u_n, \ldots$ из элементов поля P свяжем формальный степенной ряд от формальной переменной x.

$$G(x) = u_0 + u_1 x + u_2 x^2 + \cdots + u_n x^n + \cdots = \sum_{n=0}^{\infty} u_n x^n. \tag{1.25}$$

Степенной ряд последовательности иногда называют производящей функцией этой последовательности. Однако в данном случае ни область определения, на область значений "функции"не могут быть указаны. Рассматриваемая конструкция является лишь формальным символом, отражающим линейный порядок элементов последовательности. Элементы последовательности выступают в качестве коэффициентов формального степенного ряда.

Два формальных степенных ряда

$$B(x) = \textstyle\sum_{n=0}^{\infty} b_n x^n \text{ и } \sum_{n=0}^{\infty} c_n x^n$$

считаются равными, если $b_n = c_n,\ n = 0, 1, \ldots$.

Использование формальных степенных рядов позволяет рассматривать многочлен над полем P

$$p(x) = p_0 + p_1 x + \cdots + p_k x^k$$

также как формальный степенной ряд

$$p(x) = p_0 + p_1 x + \cdots + p_k x^k + 0 \cdot x^{k+1} + 0 \cdot x^{k+2} + \cdots .$$

На множестве степенных рядов определяют операции сложения и умножения по правилам, аналогичным правилам сложения и умножения многочленов:

$$B(x) + C(x) = \sum_{n=0}^{\infty} (b_n + c_n) x^n,$$

$$B(x)C(x) = \textstyle\sum_{n=0}^{\infty} (d_n) x^n, \text{ где } d_n = \sum_{k=0}^{n} b_k c_{n-k},\ n = 0, 1, \ldots .$$

Если $B(x)$ и $C(x)$ – многочлены, то эти операции имеют обычный смысл сложения и умножения многочленов. В то же время, как видно, можно складывать и перемножать обычные многочлены и формальные степенные ряды смешанным образом(один операнд – многочлен, а другой – формальный степенной ряд).

Множество формальных степенных рядов с двумя рассмотренными операциями образует кольцо. Аддитивной единицей кольца является формальный ряд, соответствующий последовательности

$$< 0 >= 0, 0, \ldots$$

из аддитивных единиц 0 поля P, а мультипликативной единицей – формальный ряд, соответствующий последовательности

$$< 1 >= 1, 0, 0 \ldots ,$$

начинающейся мультипликативной единицей 1 поля P и продолжающийся аддитивными единицами этого поля.

Теорема 1.2.28 *Формальный степенной ряд*

$$B(x) = \sum_{n=0}^{\infty} b_n x^n$$

имеет обратный относительно операции умножения элемент $B(x)^{-1}$ *тогда и только тогда, когда* $b_0 \neq 0$.

Доказательство. Пусть $C(x) = B(x)^{-1}$, то есть

$$B(x)C(x) =< 1 > .$$

в кольце степенных рядов. Тогда коэффициенты $c_0, c_1, \ldots$, $b_0, b_1, \ldots$ степенных рядов $C(x)$ и $B(x)$ удовлетворяют соотношениям

$$\begin{aligned} &d_0 = b_0 c_0 = 1, \\ &d_1 = b_0 c_1 + b_1 c_0 = 0. \\ &\ldots \\ &d_n = b_0 + c_n + b_1 c_{n-1} + \ldots + b_n c_0 = 0. \ldots \end{aligned}$$

Из первого соотношения следует, что $b_0 \neq 0$, и c_0 однозначно определяется как b_0^{-1} в поле P. Остальные коэффициенты $c_i,\ i = 1, 2, \ldots$ при этом определяются однозначно по рекурсивной схеме.

Если $B(x)$ имеет обратный элемент, то можно определить операцию деления $\frac{A(x)}{B(x)} = A(x)B(x)^{-1}$. Формально результат можно получить делением "углом."

Пусть $u_0, u_1, \ldots$ линейная последовательность k-го порядка над полем P, удовлетворяющая рекуррентному соотношению

$$u_{n+k} = a_{k-1}u_{n+k-1} + a_{k-2}u_{n+k-2} + \cdots + a_0u_n,\ n = 0, 1, \ldots.$$

Многочлен

$$f^*(x) = 1 - a_{k-1}x - a_{k-1}x^2 - \cdots - a_0x^k$$

над полем P называется *возвратным, или двойственным многочленом* этой последовательности. Характеристический многочлен

$$f(x) = x^k - a_{k-1}x^{k-1} - a_{k-1}^{k-1} - \cdots - a_0$$

и возвратный характеристический многочлен последовательности порядка k связаны соотношением

$$f^*(x) = x^k f(x^{-1}).$$

Отсюда

$$f(x) = x^k f^*(x^{-1}).$$

Алгоритм Берлекэмпа–Месси. Пусть задан отрезок ЛРП с неизвестным минимальным многочленом степени не более k, содержащий не менее $2k$ элементов. Приведём одну из модификаций алгоритма Берлекэмпа-Месси построения минимального многочлена $m(x)$

Пусть $u_0, u_1, \ldots$ – последовательность над конечным полем P и $G(x) = \sum_{n=0}^{\infty} u_n x^n$ – представляющий эту последовательность формальный степенной ряд. Для $j = 0, 1, \ldots$ определим многочлены $g_j,\ h_j$ над полем P, целые числа m_j и элементы b_j из поля P следующим образом. Положим

$$g_0(x) = 1,\ h_0(x) = x,\ m_0 = 0,\ b_0 = u_0.$$

Далее для $j = (0, 2k-1)$ выполнить

1. $g_{j+1}(x) = g_j(x) - b_j h_i(x)$;
2. $h_{j+1}(x) = \begin{cases} b_j^{-1} x g_j(x), & \text{если } b_j \neq 0,\ m_j \geq 0, \\ x h_j(x) & \text{в противном случае;} \end{cases}$
3. $m_{j+1} = \begin{cases} -m_j, & \text{если } b_j \neq 0,\ m_j \geq 0, \\ m_j + 1 & \text{в противном случае;} \end{cases}$
4. Присвоить b_{j+1}значение коэффициента при x^{j+1} формального ряда $g_{j+1}(x)G(x)$.

Замечание. Поскольку в вычислениях используются только первые $2k$ членов последовательности, то вместо формального ряда $G(x)$ можно использовать многочлен

$$G_{2k-1}(x) = \sum_{n=0}^{2k-1} u_n x^n.$$

Если $u_0, u_1, \ldots$ – ЛРП с минимальным многочленом степени k, то после выполнения указанных действий получим многочлен $g_{2k}(x)$, равный возвратному минимальному многочлену. Искомый минимальный многочлен в этом случае может быть получен как

$$m(x) = x^k g_{2k}(1/x).$$

Если же заранее известно лишь, что $\deg m(x) \le k$, то минимальный многочлен определяется равенством

$$m(x) = x^r g_{2k}(1/x),$$

где $r = \lfloor k + 1/2 - m_{2k}/2 \rfloor$.

Пример 1.2.32 Пусть 8 членов ЛРП над полем $GF(3)$ порядка $k \le 4$ образуют её начальный отрезок

$$0, 2, 1, 0, 1, 2, 1, 0,$$

тогда

$$G_7(x) = 2x + x^2 + x^4 + 2x^5 + x^6.$$

Работа алгоритма представлена в следующей таблице

j	$g_j(x)$	$h_j(x)$	m_j	b_j
0	1	x	0	0
1	1	x^2	1	2
2	$1 + x^2$	$2x$	-1	1
3	$1 + x + x^2$	$2x^2$	0	0
4	$1 + x + x^2$	$2x^3$	1	2
5	$1 + x + x^2 + 2x^3$	$2x + 2x^2 + 2x^3$	-1	2
6	$1 + x^3$	$2x^2 + 2x^3 + 2x^4$	0	1
7	$1 + x^2 + 2x^3 + x^4$	$x + x^4$	0	1
0	$1 + 2x + x^2 + 2x^3$		0	

В данном случае $r = \lfloor 4 + 1/2 - m_8/4 \rfloor = 4$. Поэтому

$$m(x) = x^4 + 2x^3 + x^2 + 2x.$$

Рекуррентное соотношение наименьшего порядка, которому удовлетворяет данная последовательность, имеет вид

$$u_{n+4} = u_{n+3} + 2u_{n+2} + u_{n+1}.\ n = 0, 1, \ldots.$$

Пример 1.2.33 Пусть первые 8 членов ЛРП над полем $GF(2)$ следующие:

$$1, 1, 0, 0, 1, 0, 1, 1.$$

Используем многочлен $G_7(x) = 1 + x + x^4 + x^6 + x^7$ над полем $GF(2)$ вместо формального степенного ряда $G(x)$ последовательности. Применим алгоритм Берлекэмпа–Мэсси, чтобы найти ЛРП наименьшего порядка, с указанными первыми элементами. Работу алгоритма представим в таблице:

j	$g_j(x)$	$h_j(x)$	m_j	b_j
0	1	x	0	1
1	$1+x$	x	0	0
2	$1+x$	x^2	1	1
3	$1+x+x^2$	$x+x^2$	-1	1
4	1	x^2+x^3	0	1
5	$1+x^2+x^3$	x	0	0
6	$1+x^2+x^3$	x^2	1	0
7	$1+x^2+x^3$	x^3	2	0
0	$1+x^2+x^3$		3	

В этом примере $r = \lfloor 4 + 1/2 - m_8/2 \rfloor = 3$ и, следовательно,

$$m(x) = x^3(1 + (1/x)^2 + (1/x)^3) = x^3 + x + 1.$$

Таким образом, заданные элементы образуют начальный отрезок ЛРП, удовлетворяющей рекуррентному соотношению

$$u_{n+3} = u_{n+1} + u_n, n = 0, 1, \ldots,$$

и не существует ЛРП меньшего порядка, имеющей тот же начальный отрезок.

Пример 1.2.34 Построим ЛРП над полем $GF(2)$ наименьшего порядка, не превышающего 5, первые 10 членов которой образуют отрезок

$$0, 0, 1, 1, 0, 1, 1, 1, 0, 1.$$

Используем многочлен

$$G_{10} = x^2 + x^3 + x^5 + x^6 + x^7 + x^9,$$

представляющий указанный отрезок. Работа алгоритма Берлекэмпа-Месси представлена в таблице

j	$g_j(x)$	$h_j(x)$	m_j	b_j
0	1	x	0	0
1	1	x^2	1	0
2	1	x^3	2	1
3	$1+x^3$	x	-2	1
4	$1+x+x^3$	x^2	-1	1
5	$1+x+x^2+x^3$	x^3	0	1
6	$1+x+x^2$	$x+x^2+x^3+x^4$	0	0
7	$1+x+x^2$	$x^2+x^3+x^4+x^5$	1	1
0	$1+x+x^3+x^4+x^5$	$x+x^2+x^3$	0	1
9	$1+x^2+x^4+x^5$	$x^2+x^3+x^4$	0	1
10	$1+x^3+x^5$	$x+x^3+x^5+x^6$	0	0

В данном случае $r = \lfloor 5+1/2+m_{10}/2 \rfloor = \lfloor 5+1/2+0 \rfloor = 5$. Следовательно,

$$m(x) = x^5(1 + (x^{-1})^3 + (x^{-1})^5) = x^5 + x^2 + 1.$$

Указанный отрезок является начальным отрезком ЛРП, определяемой рекуррентным соотношением

$$u_{n+5} = u_{n+2} + u_n.$$

1.3 Эллиптические кривые

1.3.1 Алгебраические кривые и эллиптические кривые

Теория эллиптических кривых является одним из важнейших разделов алгебраической геометрии, точнее, ее раздела, изучающего плоские алгебраические кривые (вообще об алгебраической геометрии см. фундаментальную монографию [69], а о плоских кривых — довольно доступно написанную книгу [62].) Она тесно связана также с комплексным анализом (а именно, с теорией эллиптических функций, см., например, [44]) и с теорией чисел (в частности, с диофантовым анализом, см., например, [61]), до сих пор интенсивно развивается и чрезвычайно обширна и сложна. В ее создании принимали участие многие крупнейшие математики прошлого, а начинается она (в определенном смысле) с последнего из великих древнегреческих математиков — Диофанта. Структуру группы на эллиптических кривых определил знаменитый французский математик Анри Пуанкаре. Долгое время теория эллиптических кривых являлась чистейшей областью математики, не имеющей за ее пределами никаких приложений. Представление об этой теории и ее взаимосвязях с другими разделами математики можно получить, заглянув, например, в книги [40], [42], [56], [21].

В восьмидесятые годы прошлого века теория эллиптических кривых получила приложения в области построения алгоритмов факторизации больших чисел (см., например [17], [60]) и через эти приложения вошла в криптографию (как в ее области, связанные с построением криптосистем с открытым ключом, протоколов распределения ключей, и протоколов цифровой подписи, а также в области классической криптографии, связанные с генерацией псевдослучайных последовательностей).

В криптографии с открытым ключом эллиптические кривые являются основой ряда алгоритмов ECC – криптографии на эллиптических кривых. Далее мы для краткости будем иногда называть ее просто эллиптической криптографией. Идея создания эллиптической криптографии была выдвинута в 1985 году независимо в работах В. Миллера и Н. Коблица. Интерес в криптографии к эллиптическим кривым обусловлен, с одной стороны, тем, что они являются богатым источником конечных абелевых групп, обладающих полезными структурными свойствами, так и тем, что на их основе обеспечиваются те же криптографические свойства, которыми обладают числовые или полиномиальные криптосистемы, но при существенно меньшем размере ключа.

Далее мы даем краткое и ориентированное только на последующие приложения введение в теорию эллиптических кривых.

Алгебраической кривой порядка n над полем $\mathcal{F}$ называется множество точек $(x, y), x, y \in \mathcal{F}$, удовлетворяющих уравнению $F(X, Y) = 0$, где $F(X, Y)$ многочлен степени n с коэффициентами из $\mathcal{F}$. Уточним, что под *степенью одночлена* понимается *сумма* степеней входящих в него переменных, а под *степенью многочлена* — максимальная степень составляющих его одночленов.

Пример 1.3.1 Степень одночлена $7X^3Y^2$ равна 5, а степень многочлена $7X^3Y^2 + 2X^4 + 6Y^2 3X^2Y^4$ равна 6.

Пары $(x, y) \in \mathcal{F}^{\in}$, элементов поля $\mathcal{F}$, удовлетворяющие уравнению кривой, называются ее *точками*.

Кривая, определяемая уравнением первой степени $aX + bY + c = 0$, называется прямой.

Кривая, определяемая уравнением второй степени

$$a_{11}X^2 + a_{12}XY + a_{22}Y^2 + a_1X + a_2Y + a_0 = 0,$$

называется кривой второго порядка.

Пример 1.3.2 Над полем действительных чисел среди таких кривых, кроме эллипсов, гипербол и парабол, встречаются и вырожденные кривые, которые представляют собой пару прямых (иногда совпадающих), они тоже определяются уравнением второго порядка

$$(a_1X + b_1Y + c_1)(a_2X + b_2Y + c_2) = 0.$$

Точка (x_0, y_0) кривой $F(X, Y) = 0$ называется *неособой*, если в ней не равны нулю обе частные производные многочлена $F(X, Y)$.

Поясним, что здесь частные производные $\frac{\partial F}{\partial x}, \frac{\partial F}{\partial y}$ определяются известными формальными правилами дифференцирования, применяемыми к многочленами над произвольным полем, а именно

$$\frac{\partial(A(y)F + B(y)G)}{\partial x} = A(y)\frac{\partial F}{\partial x} + B(y)\frac{\partial G}{\partial x}$$

(линейность дифференцирования), и

$$\frac{\partial(FG)}{\partial x} = \frac{\partial(F}{\partial x}G + \frac{\partial G}{\partial x}F$$

(правило Лейбница), из которых следует, что если $F(x, y)$ записан по степеням x как

$$F(x, y) = A_0(y) + A_1(y)x + \ldots + A_n(y)x^n,$$

то

$$\frac{\partial F}{\partial x} = A_1(y) + 2A_2(y)x + \ldots + nA_n(y)x^{n-1}$$

и аналогичные правила справедливы для частного дифференцирования по переменной y. Кривая называется *неособой,* или *гладкой*, если все ее точки неособые. В любой такой точке (x, y) к ней можно провести *касательную* , т.е прямую, определяемую уравнением

$$(X - x)\frac{\partial F}{\partial x} + (Y - y)\frac{\partial F}{\partial y} = 0.$$

Неособая кривая третьего порядка над полем $\mathcal{F}$ и называется *эллиптической кривой* над тем же полем, если на ней есть хотя бы одна точка. Но если даже таких точек нет, то они могут появиться, если рассмотреть эту кривую над каким-нибудь расширением поля $\mathcal{F}$. Рассмотрение той же кривой над расширением поля далее будет часто использоваться.

Ньютон доказал, что над полем действительных чисел любую эллиптическую кривую можно преобразовать к виду $y^2 = x^3 + ax + b$ (получившему позже название *формы Вейерштрасса*) с помощью замены координат вида

$$X = \frac{l_1(x,y)}{l_3(x,y)}, Y = \frac{l_2(x,y)}{l_3(x,y)}, l_i \neq 0$$

(*проективной замены координат*).

Упражнение 1.3.1 Проверьте, что *кривая Ферма* $X^3 + Y^3 = 1$ заменой $X = 3x/y, Y = (y-9)/y$ сводится к кривой $y^2 - 9y = x^3 - 27$, и далее к кривой вида $y^2 = x^3 - 27/4$.

В случае произвольного поля характеристики, отличной от двух, произвольную эллиптическую кривую можно преобразовать к виду

$$Y^2 + a_1XY + a_3Y = X^3 + a_2X^2 + a_4X + a_6,\ a_i \in \mathcal{F},$$

также называемому формой Вейерштрасса. Далее для произвольного поля используем следующее определение.

Эллиптической кривой $E\mathcal{E}$ над полем $\mathcal{F}$ называется гладкая кривая, задаваемая уравнением вида

$$Y^2 + a_1XY + a_3Y = X^3 + a_2X^2 + a_4X + a_6,\ a_i \in \mathcal{F}. \tag{1.26}$$

Будем обозначать $\mathcal{EF}$ множество точек $(x,y) \in \mathcal{F}^2$, удовлетворяющих этому уравнению и содержащее кроме того *бесконечно удаленную* точку, обозначаемую O. Пока что мы ограничиваемся представлением о бесконечно удаленной точке O как о точке, расположенной бесконечно далеко в положительном направлении оси y и рассматриваемой в качестве третьей точки пересечения эллиптической кривой любой вертикальной линией, всякая такая линия пересекает кривую в точках $(x_1,y_1), (x_2,y_2), O$.

Если $\mathcal{K}$ – расширение поля $\mathcal{F}$, то $\mathcal{EK}$ обозначает множество точек $(x,y) \in \mathcal{K}^2$, удовлетворяющих (3.1), вместе с точкой O. В определение эллиптической кривой удобно включить и требование ее гладкости и для *алгебраически замкнутого расширения* $\mathcal{K}$ поля $\mathcal{F}$ (т.е. такого расширения, в котором любой многочлен имеет корень). Иными словами, два уравнения

$$a_1Y - 3X^2 - 2a_2X - a_4 = 0,$$

$$2Y + a_1X + a_3 = 0 \tag{1.27}$$

не должны удовлетворяться одновременно ни в одной точке $(x,y) \in \mathcal{EK}^2$.

Странная, на первый взгляд, нумерация индексов и странное название (эллиптическая кривая) объясняется следующим образом. В частном случае, когда $a_1 = a_3 = a_3 = a_4 = a_6 = 0$ эллиптическая кривая

$$Y^2 = X^3 - X,$$

над полем $\mathcal{R}$ действительных чисел при больших X ведет себя как функция $Y = X^{3/2}$, которая может быть параметризирована подстановкой $X = T^2$ и $Y = T^3$. Если считать, что, что X имеет степень 2, а Y имеет степень 3, индексы коэффициентов в (3.1) определяют степени, которые должны быть даны коэффициентам, чтобы это уравнение стало однородным, то есть, чтобы степень каждого члена была равна 6 (в общем случае эллиптическая кривая не имеет рациональной параметризации). Кривой $Y^2 = f(X)$ соответствует *эллиптический интеграл*

$$\int \frac{dX}{\sqrt{f(X)}},$$

не берущийся в элементарных функциях. Эллиптические интегралы, в свою очередь, связаны с вычислением длин дуг эллипсов [56],[42]. Их изучение началось еще в 18 веке в работах Фаньяно, Эйлера и Лежандра.

Дискриминат и инвариант эллиптической кривой. В зависимости от характеристики $char\mathcal{F}$ поля $\mathcal{F}$ общее уравнение эллиптической кривой может быть упрощено. Если поле $\mathcal{F}$ не является полем характеристики 2, то без потери общности можно полагать, что $a_1 = a_3 = 0$, то есть вместо уравнения (3.1) рассматривать уравнение

$$Y^2 = X^3 + a_2X^2 + a_4X + a_6,\ a_i \in \mathcal{F}. \tag{1.28}$$

Если характеристика поля не равна $2, 3$, то после упрощения левой части (3.4), линейной заменой переменной (а именно, $X \to X - 1/3a_2$) можно также удалить член X^2 и без потери общности полагать, что кривая задана уравнением вида

$$Y^2 = X^3 + aX + b,\ a, b, \in \mathcal{F},\ \mathrm{char}\mathcal{F} \neq 2, 3. \tag{1.29}$$

В частности, в таком виде представимы эллиптические кривые над полем нулевой характеристики, например, эллиптические кривые над полем $\mathcal{R}$ действительных чисел. Последние, хотя и не применяются в криптографии, допускают графическую интерпретацию самой кривой и наглядное объяснение важных ее свойств.

С уравнением (3.4) эллиптической кривой E можно связать *дискриминант*

$$\Delta(E) = \left(\frac{a}{3}\right)^3 + \left(\frac{b}{2}\right)^2 = \frac{(4a^3 + 27b^2)}{4 \times 27} \tag{1.30}$$

многочлена $x^3 + ax + b$ и не изменяющийся при линейных преобразованиях *j-инвариант*

$$j(E) = \frac{1728(4a^3)}{\Delta}. \tag{1.31}$$

Понятие дискриминанта и j-инварианта в общем случае кривой 3.1 выглядят более громоздко. А именно,

$$\Delta = -b_2^2 b_8 - 8b_4^3 - 27b_6^2 + 9b_2 b_4 b_6,$$

где $b_2 = a_1^2 + 4a_2, b_4 = 2a_4 + a_1 a_3, b_6 = a_3^2 + 4a_6, b_8 = a_1^2 a_6 + 4a_2 a_6 - a_1 a_3 a_4 + a_2 a_3^2 - a_4^2$,

$$j(E) = \frac{c_4^3}{\Delta}, c_4 = b_2^2 - 24b_4.$$

Если $\Delta = 0$, то указанный многочлен имеет кратные корни и в точке $(x, 0)$ нарушается условие гладкости кривой. Вообще справедлива

Теорема 1.3.1 *Кривая E гладкая тогда и только тогда, когда ее дискриминант ненулевой.*

Отметим еще, что при том же условии эта кривая не имеет рациональной параметризации.

Уточним, что мы имели в виду выше под возможностью упрощения вида уравнений для эллиптических кривых. Две кривые E и E' над полем $\mathcal{F}$ называются *изоморфными*, если они переходят друг в друга при *допустимой* замене координат

$$X = u^2 x + r, Y = u^3 y + u^2 s x + t.$$

Упражнение 1.3.2 Проверьте, что множество допустимых преобразований координат образует группу относительно операции композиции преобразований.

Упражнение 1.3.3 Проверьте, что множество всех эллиптических кривых над данным полем разбивается на классы эквивалентности относительно отношения изоморфизма.

Упражнение 1.3.4 Докажите, что любая эллиптическая кривая над полем $\mathcal{F}$ характеристики $\neq 2$ изоморфна кривой вида

$$Y^2 = X^3 + a_2 X^2 + a_4 X + a_6,\ a_i \in \mathcal{F}.$$

Указание: сделать замену переменных

$$X = x, Y = y - \frac{a_1}{2} x - \frac{a_3}{2}.$$

Упражнение 1.3.5 Докажите, что любая эллиптическая кривая над полем $\mathcal{F}$ характеристики $\neq 2, 3$ изоморфна кривой вида

$$Y^2 = X^3 + a_4 X + a_6,\ a_i \in \mathcal{F}.$$

Указание: применить предыдущее утверждение и сделать замену переменных

$$X = x - a_2/3, Y = y.$$

Упражнение 1.3.6 Докажите, что любая эллиптическая кривая над полем $\mathcal{F}$ характеристики 2 изоморфна кривой вида

$$Y^2 + XY = X^3 + a_2X^2 + a_6,\ a_i \in \mathcal{F}$$

или кривой вида

$$Y^2 + a_3Y = X^3 + a_4X + a_6,\ a_i \in \mathcal{F}.$$

Указание. Если $a_1 \neq 0$, сделать замену переменных

$$X = a_1^2x + a_3/a_1, Y = a_1^3y,$$

и получить уравнение вида

$$Y^2 + XY = X^3 + a_2X^2 + a_4X + a_6,\ a_i \in \mathcal{F},$$

а потом сделать замену переменных вида

$$X = x, Y = y + a_4.$$

Если $a_1 = 0$, сделать замену переменных

$$X = x + a_2, Y = y.$$

Эллиптические кривые неизоморфные над данным полем $\mathcal{F}$ могут быть изоморфными над его расширением. Справедлива

Теорема 1.3.2 *Две кривые изоморфны над алгебраическим замыканием поля $\mathcal{F}$ тогда и только тогда, когда они имеют равные j-инварианты.*

Для доказательства теоремы достаточно проверить, что при выполнении произвольной допустимой замены координат коэффициенты уравнения преобразуются по следующим формулам

$$ua_1' = a_1 + 2s, u^2a_2' = a_2 - sa_1 + 3r - s^2, u^3a_3' = a_3 + ra_1 + 2t,$$

$$u^4a_4' = a_4 - sa_3 + 2ra_2 - (t + rs)a_1 + 3r^2 - 2st,$$

$$u^6a_6' = a_6 + ra_4 + r^2a_2 + r^3 - ta_3 - t^2 - rta_1,$$

$$u^2b_2' = b_2 + 12r, u^4b_4' = b_4 + rb_2 + 6r^2, u^6b_6' = b_6 + 2rb_4 + r^2b_2 + 4r^3,$$

$$u^8b_8' = b_8 + 3rb_6 + 3r^2b_4 + r^3b_2 + 3r^4, u^4c_4' = c_4, u^6c_6' = c_6, u^{12}\Delta' = \Delta.$$

Справедлива также

Теорема 1.3.3 *Для каждого $j \in \mathcal{F}$ существует эллиптическая кривая над $\mathcal{F}$ с инвариантом j. Если $j \neq 0, 1728$, то такой кривой является*

$$Y^2 + XY = X^3 - \frac{36}{j - 1728}x - \frac{1}{j - 1728},$$

для которой $c_4 = \frac{j}{j-1728}, \Delta = \frac{j^2}{(j-1728)^3}$. Группа автоморфизмов любой кривой конечна. При $j \neq 0, 1728$ она состоит только из двух автоморфизмов. Нетривиальный автоморфизм задается преобразованием

$$X = x, Y = y - a_1 x - a_3.$$

Эллиптические кривые с нулевым $j-$инвариантом называются *суперсингулярными*, если $j-$инвариант не равен нулю, то *несуперсингулярными.*

Характерные формы эллиптической кривой над полем действительных чисел приведены на Рис. 3.1 (кривые с отрицательным дискриминантом) и на Рис.3.2 (кривая с положительным дискриминантом). В случае характеристики 2 левая часть уравнения (3.1) имеет вид

$$Y^2 + a_3 Y$$

в случае суперсингулярных эллиптических кривых и

$$Y^2 + a_1 XY, \ a_1 \neq 0,$$

в случае *несуперсингулярных* эллиптических кривых. Действительно, тогда $ua_1' = a_1, c_4 = b_2^2 = a_1^4$, значит

$$j = 0 \leftrightarrow a_1 = 0.$$

Для полей характеристики 2 можно, выбирая подходящие r, s, t в допустимом преобразовании координат, положить $a_1 = 1$ в несуперсингулярном случае:

$$Y^2 + XY = X^3 + a_2 X^2 + a_6, \ a_i \in \mathcal{F}. \tag{1.32}$$

Тогда $b_2 = 1, b_4 = b_6 = 0, b_8 = a_6, c_4 = 1, \Delta = a_6, j = 1/a_6$.

Завершая обзор типов уравнений эллиптических кривых заметим, что кривые характеристики 3 (char $\mathcal{F} = 3$) описываются уравнением (3.2), которое при $a_2 \neq 0$ к виду (3.3) не приводится.

1.3.2 Группа точек эллиптической кривой

На множестве $\mathcal{EF}$, состоящем из точек эллиптической кривой (3.1) и еще одного элемента — бесконечно удаленной точки кривой (формально пока не являющейся точкой кривой), можно определить операцию, обладающую свойствами

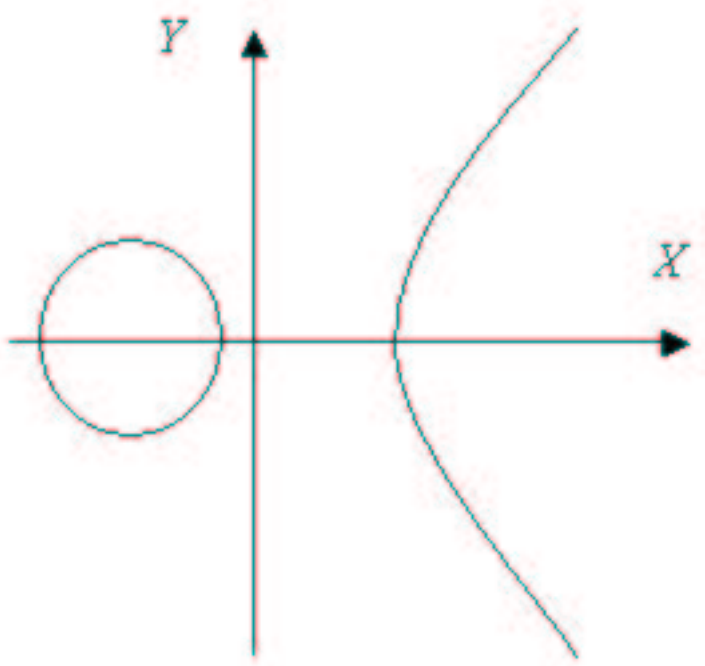

Рис. 1.9: Эллиптическая кривая с отрицательным дискриминантом.

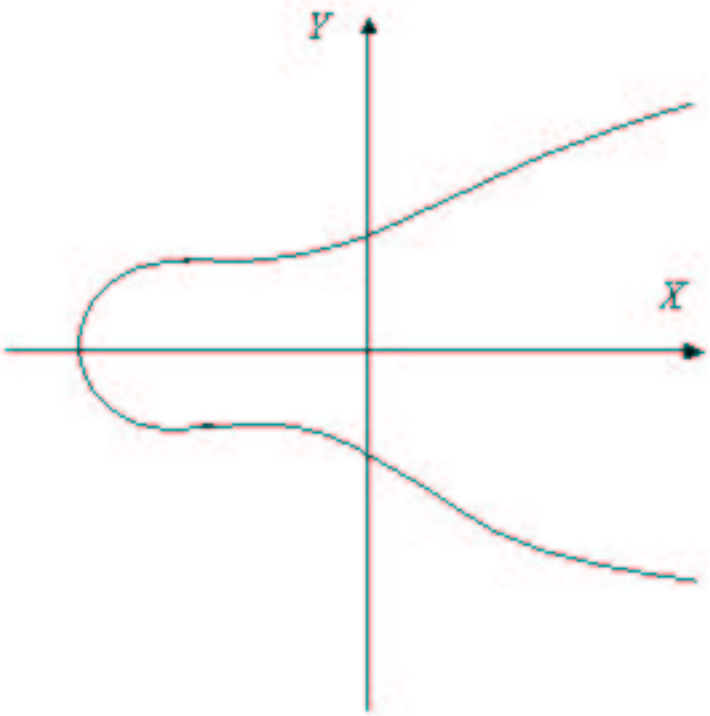

Рис. 1.10: Эллиптическая кривая с положительным дискриминантом.

операции абелевой группы. Принято получающуюся при этом группу рассматривать как аддитивную группу, а операцию называть операцией сложения и обозначать, как обычно, знаком плюс. Упомянутая дополнительная точка выполняет роль нейтрального элемента (в аддитивной записи — нуля) этой группы и обозначается O.

Если $\mathcal{K}$ – расширение поля $\mathcal{F}$, то $\mathcal{EK}$ обозначает множество точек $(x, y) \in \mathcal{K}^2$, удовлетворяющих (3.1), вместе с точкой O. Будем использовать обозначение $\mathcal{EK}$ лишь в том случае, когда рассматриваются как поле $\mathcal{F}$, так и его расширение $\mathcal{K}$.

По определению, полагаем для любой точки $(x, y) \in \mathcal{E}(\mathcal{F})$

$$(x, y) + O = O + (x, y) = (x, y), O + O = O.$$

Чтобы определить в общем случае операцию сложения абелевой группы, сначала покажем, что каждой точке (x, y) эллиптической кривой можно сопоставить в определенном смысле симметричную точку (далее будет ясно, что такая точка и будет точкой $-(x, y)$, противоположной к (x, y) точкой в группе данной кривой). Заметим, что вместе с точкой (x, y) кривая имеет и точку

$$(x, \tilde{y}) = (x, -a_1x - a_3 - y). \tag{1.33}$$

В этом не трудно убедиться непосредственным вычислением левой и правой части уравнения (3.1) при $X = x$, $Y = -a_1x - a_3 - y$ и учитывая, что при $X = x$ и $Y = y$ меет место равенство. Симметричность проявляется в том, что, как не трудно проверить, по тому же правилу точке $(x, \tilde{y})$ соответствует исходная точка, так как имеет место *инволютивный закон*:

$$(x, y) = (x, \tilde{\tilde{y}}).$$

Обратим внимание, что если уравнение приведено к виду (3.2) (что возможно, если $3 \neq char\mathcal{F}$), то

$$(x, \tilde{y}) = (x, -y). \tag{1.34}$$

В частности, если кривая определена над полем $\mathcal{R}$ действительных чисел, то точки (x, y) и $(x, -y)$ располагаются на прямой $Y = x$ симметрично относительно оси абсцисс (Рис.3.3).

Для суперсингулярных и несуперсингулярных кривых характеристики 2 симметричная точка $(x, \tilde{y})$ определяется соответственно уравнениями (частные случаи (3.8) при $a_1 = 0, a_3 = 1$ и $a_1 = 1, a_3 = 0$)

$$(x, \tilde{y}) = (x, y + 1) \tag{1.35}$$

$$(x, \tilde{y}) = (x, x + y) \tag{1.36}$$

Будем считать, что

$$(x, y) + (x, \tilde{y}) = O$$

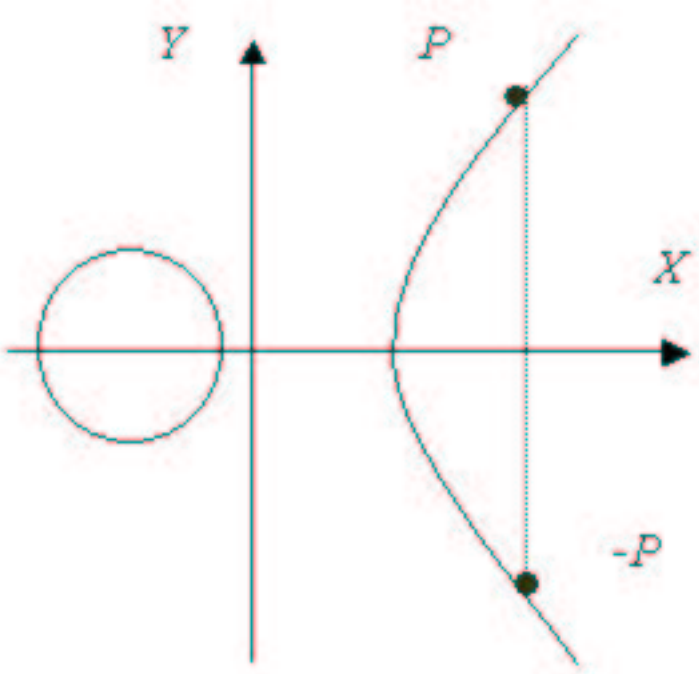

Рис. 1.11: Противоположные точки

и обозначать

$$(x, \tilde{y}) = -(x, y).$$

Как видим, множество $\mathcal{E}(\mathcal{F})$ удовлетворяет двум аксиомам группы (существует нулевой элемент и каждому элементу соответствует противоположный элемент)

Операция сложения пока что определена нами для случаев, когда хотя бы одно слагаемое есть O или слагаемые (x_1, y_1)и (x_2, y_2) таковы, что

$$x_1 = x_2, \text{ и } y_2 = \tilde{y}_1 \text{ или, что то же самое, } y_1 = \tilde{y}_2.$$

Осталось определить сумму $(x_1, y_1) + (x_2, y_2)$ для остальных случаев, когда

$$x_1 \neq x_2 \tag{1.37}$$

или

$$x_1 = x_2, \text{ и } y_2 \neq \tilde{y}_1 \text{ (или } y_1 \neq \tilde{y}_2). \tag{1.38}$$

Пусть $P = (x_1, y_1)$ и $Q = (x_2, y_2)$ две точки эллиптической кривой, удовлетворяющие условию (3.13), и ни одна из них не есть O. Обозначим $\lambda(x_1, x_2, y_1, y_2) \neq 0$ элемент поля F, такой, что прямая на плоскости F^2

$$\mathcal{L} = \{(X, Y) \setminus Y - y_1 = \lambda(P, Q)(X - x_1)\} \tag{1.39}$$

содержит эти две точки эллиптической кривой $\mathcal{E}(\mathcal{F})$.

Такой элемент легко вычислить:

$$\lambda(P,Q) = \lambda(x_1, x_2, y_1, y_2) = \frac{y_2 - y_1}{x_2 - x_1}; \tag{1.40}$$

Если же $P = Q = (x', y')$ (то есть имеет место условие (3.13)), вместо прямой (3.14) будем использовать прямую

$$\mathcal{L}' = \{(X,Y) \setminus Y - y' = \lambda'(P)(X - x')\}, \tag{1.41}$$

где

$$\lambda'(P) = \left.\frac{\partial F(X,Y)/\partial X}{\partial F(X,Y)/\partial Y}\right|_{X=x', Y=y'} =$$

$$= \left.\frac{(a_1 Y - 3X^2 - 2a_2 X - a_4}{2Y + a_1 X + a_3}\right|_{X=x', Y=y'}. \tag{1.42}$$

Очевидно, она содержит точку $P = Q$. Отметим, что знаменатель в выражении (3. 17) в рассматриваемом случае не может быть нулевым

Упражнение 1.3.7 Проверьте это.

Покажем, что кроме точек P и Q прямая $\mathcal{L}$ (3.14), как и множество $\mathcal{L}'$ (3.16) содержит еще одну точку R эллиптической кривой (3.1). В случае прямой (3.14) эта дополнительная точка может совпасть с точкой P или с точкой Q (Такая точка называется *точкой инфлекции*).

Уравнения прямых (3.14) и (3.16) равносильны соответственно уравнениям

$$Y = \lambda X + \beta, \tag{1.43}$$

где

$$\lambda = \lambda(P,Q),\ \beta = y_1 - \lambda x_1. \tag{1.44}$$

и

$$Y = \lambda' x + \beta', \tag{1.45}$$

где

$$\lambda' = \lambda'(P),\ \beta' = y_1 - \lambda' x_1. \tag{1.46}$$

Точка $(x, \lambda x + \beta) \in \mathcal{L}$ (или точка $(x, \lambda' x + \beta') \in \mathcal{L}'$ лежит на эллиптической кривой только в том случае, когда

$$(\lambda x + \beta)^2 + a_2 x(\lambda x + \beta) + a_3(\lambda x + \beta) = x^3 + a_2 x^2 + a_4 x + a_6$$

(или соответственоо

$$(\lambda' x + \beta')^2 + a_2 x(\lambda' x + \beta') + a_3(\lambda' x + \beta') = x^3 + a_2 X^2 + a_4 x + a_6)$$

Отсюда следует, что кубическое уравнение

$$(\lambda X + \beta)^2 + a_2 X(\lambda X + \beta) + a_3(\lambda X + \beta) = X^3 + a_2 X^2 + a_4 X + a_6$$

(или соответственоо

$$(\lambda' X + \beta')^2 + a_2 X(\lambda' X + \beta') + a_3(\lambda' X + \beta') = X^3 + a_2 X^2 + a_4 X + a_6)$$

имеет (с учетом кратности) три корня, среди них x_1 и x_2 (или дважды x), так как $(x_1, \lambda x_1 + \beta)$ и $(x_2, \lambda x_2 + \beta)$ (или $(x, \lambda' x + \beta')$) являются точками P и Q (точкой P) кривой.

Воспользовавшись теоремой Виета, согласно которой сумма корней нормированного многочлена равна взятому со знаком минус коэффициенту γ (или γ') при степени, предшествующей старшей степени, мы можем определить и третий корень

$$x_3 = \gamma - x_1 - x_2 \text{ (или } \ x_3 = \gamma' - 2x)$$

уравнения, а затем вторую координату

$$y_3 = y_1 + \lambda(x_3 - x_1) \text{ (или } \ y_3 = y + \lambda'(x_3 - x_1)$$

третьей точки эллиптической кривой, принадлежащей прямой (3.14) (или (3.16)).

Это позволяет получить выражение для x_3 и, следовательно, для обеих координат третьей точки

$$R = (x_3, y_3) = (\gamma - x_1 - x_2, y_1 + \lambda(x_3 - x_1)) \tag{1.47}$$

эллиптической кривой на прямой (3.15) через координаты x_1, x_2, y_1, y_2.

Аналогично определяются координаты точки

$$R = (x_3, y_3) = (\gamma' - 2x, y + \lambda'(x_3 - x)) \tag{1.48}$$

на прямой (3.16).

Определение 3.2. При условиях (3.12) или (3.13) суммой двух возможно совпадающих точек эллиптической кривой объявляется точка

$$P + Q = -R = -(x_3, y_3) \tag{1.49}$$

или

$$P + P = 2P = 2R = -(x_3, y_3), \tag{1.50}$$

где (x_3, y_3) – третья точка (3.22) или (3.23) принадлежащая множеству (3.14) или (3.16) соответственно.

Заметим, что есть соблазн назвать суммой точек P, Q саму точку R. Но так определенная операция сложения не будет иметь иметь нейтрального элемента.

Общая схема алгоритма сложения или удвоения для группы точек эллиптической кривой и конкретные формулы для вычисления координат третьей точки, когда ни одно из слагаемых не есть точка $\mathcal{O}$ и когда эти слагаемые не взаимно противоположны, рассматриваются в главе 5.

Если кривая определена над полем $\mathcal{R}$ действительных чисел, множество $\mathcal{L}$ есть в самом деле прямая, проходящая через точки P и Q кривой и пересекающая ее в третьей точке R. Суммой является противоположная к R точка $-R$ (Рис. 3.4).

Эта точка R может оказаться точкой инфлекции и совпасть с одной из точек P или Q (Рис. 3.5). Прямая $\mathcal{L}'$ есть касательная к кривой в точке $P = Q$. Тогда R есть точка пересечения касательной с кривой, $2P$ есть точка, противоположная к R точка $-R$ (Рис. 3.6).

Пример 1.3.3 На кривой $y^2 = x^3 - 36x$ возьмем точки $P = (-3, 9), Q = (-2, 8)$. Тогда при вычислении (используя формулы для кривых характеристики, не равной 2 или 3, из пято главы) $P + Q$ находим $x_3 = 6, y_3 = 0$, а при вычислении $2P$ находим $x_3 = 25/4, y_3 = -35/8$.

Упражнение 1.3.8 Если $P = (x, 0)$, то $2P = 0, 3P = P, 4P = 0$, и т.д.

Заметим, что описанная операция коммутативна и в рассмотренных случаях (3.13) и (3.14), поскольку $\lambda(x_1, x_2, y_1, y_2) = \lambda(y_1, y_2, x_1, x_2)$. и $\lambda'(x, y) = \lambda'(y, x)$.

Справедлива следующая теорема Анри Пуанкаре

Теорема 1.3.4 *Множество $\mathcal{E}(\mathcal{F})$ (множество точек эллиптической кривой вместе с точкой бесконечности O) с операцией сложения, описанной выше, является абелевой группой.*

Доказать ассоциативность операции в этой группе можно, используя явные формулы для вычисления координат точки (x_3, y_3), рассматриваемые далее.

Без громоздких вычислений можно вывести ассоциативность из следующей теоремы

Пусть три прямые $\mathcal{L}_1, \mathcal{L}_2, \mathcal{L}_3$ пересекают кубическую кривую в девяти точках $P_1, P_2, \ldots, P_9$ с возможными совпадениями и пусть $\mathcal{L}'_1, \mathcal{L}'_2, \mathcal{L}'_3$ – три прямые, пересекающие кривую в точках $Q_1, Q_2, \ldots, Q_9$. Если $P_i = Q_i$ для $i = 1, \ldots 8$, то также $P_9 = Q_9$, которую мы оставляем без доказательства, но далее доказываем нужный нам ее частный случай.

Заметим, что для доказательства тождества ассоциативности

$$P_1 + (P_2 + P_3) = (P_1 + P_2) + P_3$$

можно предполагать, что все точки конечные, т.е. $P_i \neq 0$, в противном случае оно очевидно.

Упражнение 1.3.9 Проверьте все же его и в этом случае.

Для доказательства в общем случае заметим, что точки $P_3, -P_3, O$ лежат на некоторой *вертикальной* прямой l_1 (т.е. прямой, определяемой уравнением вида $x = a$), так как эта прямая проходит через бесконечно удаленную точку O, точки $-P_1, -P_2, P_1 + P_2$ лежат на некоторой прямой l_2, точки $P_1, P_2 + P_3, T_1 = -(P_1 + (P_2 + P_3))$ лежат на некоторой прямой l_3, точки $P_1, -P_1, O$ лежат на некоторой вертикальной прямой m_1, точки $-P_2, -P_3, (P_2 + P_3)$ лежат на некоторой прямой m_2, и точки $P_3, P_1 + P_2, T_2 = -((P_1 + P_2) + P_3)$ лежат

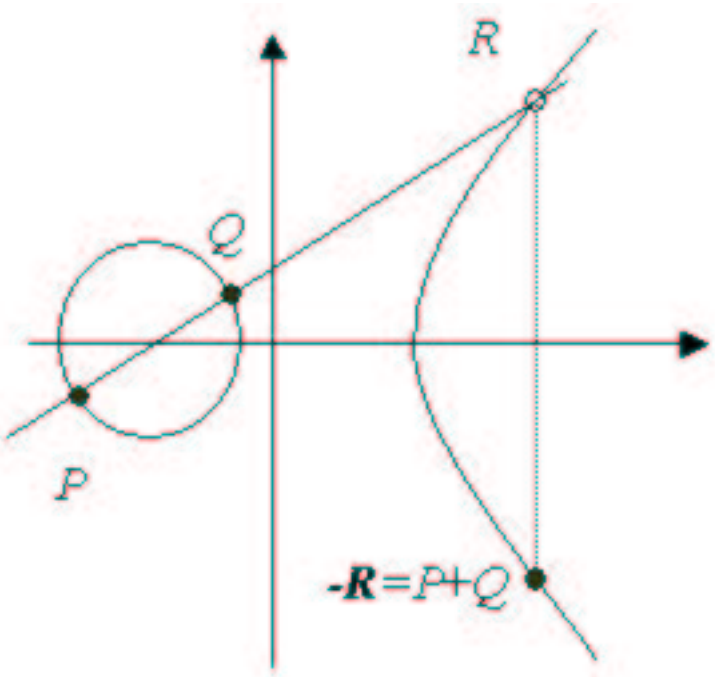

Рис. 1.12: Сложение точек (общий случай).

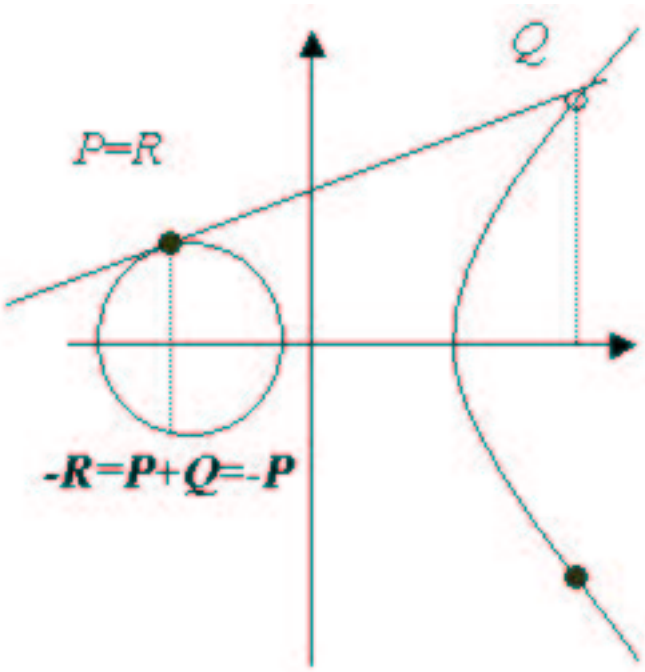

Рис. 1.13: Сложение точек с инфлекцией.

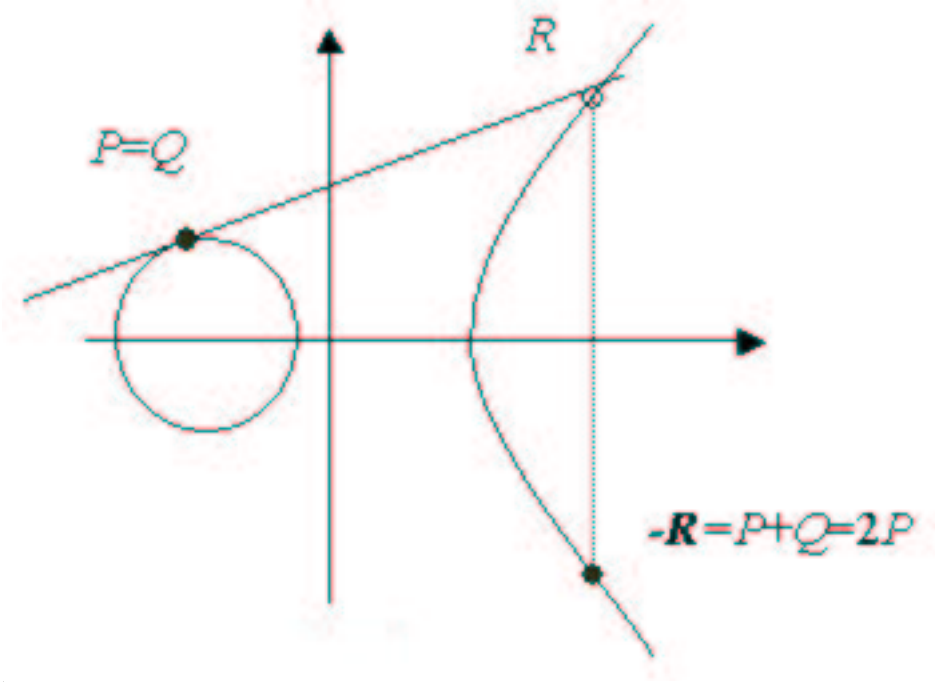

Рис. 1.14: Удвоение точки P

на некоторой прямой m_3. Применяя сформулированную выше теорему, получаем, что $T_1 = T_2$, что и требовалось доказать. Заметим однако, что указанную теорему в полной общности доказать не просто. Во первых, это теорема проективная, и среди упомянутых в ней точек P_i может встречаться бесконечно удаленная (а в нашем примере ее применения она действительно встечается). Во вторых, среди точек P_i могут быть совпадающие (и в нашем примере ее применения они тоже могут быть). Если же, например, совпадают точки Q_1 и Q_2 среди точек Q_1, Q_2, Q_3 лежащих одновременно и на одной прямой, и на данной кривой, то это означает, что эта прямая является касательной в точке $Q_1 = Q_2$ к данной кривой. Случаев касания среди данных шести прямых может быть довольно много, и каждый из них с точки зрения обычной (аффинной) геометрии определяет свою конфигурацию, не похожую на другие. С точки зрения же проективной геометрии все эти случаи одинаковы, однако провести доказательство, годящееся в общем случае, непросто. Можно вначале рассмотреть случай общего положения, когда все девять точек P_i различны, тогда остальные случаи можно вывести из общего случая аккуратно обоснованным предельным переходом. Однако эту идею реализовать не просто, тем более, что теорему надо доказать для произвольных полей, в том числе конечных.

Мы ограничимся разбором общего случая, но сделаем для любого поля и без использования приведенной выше теоремы. Воспользуемся следующими несложными леммами, которые оставляем читателю в виде упражнений.

Упражнение 1.3.10 Если $l(x,y) = ax + by + c, b \neq 0$, то любой многочлен $f(x,y)$ степени 3 представим в виде $l(x,y)q(x,y) + r(x)$, $\deg r(x) \leq 3$. Если $l(x) = ax + c, a \neq 0$, то любой многочлен $f(x,y)$ степени 2 по переменной y представим в виде $l(x)q(x,y)+r(y)$, $\deg r(y) \leq 2$.

Указание: разложить по степеням одной переменной и применить школьную теорему Безу.

Упражнение 1.3.11 Если кривая третьего порядка не содержит целиком данную невертикальную прямую, то она пересекается с ней не более чем в трех точках с учетом кратности.

Указание. Пусть кривая имеет вид $f(x,y) = 0$, а прямая $l(x,y) = 0$. Тогда $f(x,y) = l(x,y)q(x,y) + r(x), r(x) \neq 0$, $\deg r(x) \leq 3$, и если (x_i, y_i) — точки пересечения $l = 0$ с $f = 0$, то $r(x_i) = 0$.

Упражнение 1.3.12 Если кривая, в уравнение которой y входит не выше чем во второй степени, не содержит целиком данную вертикальную прямую, то она пересекается с ней не более чем в двух точках с учетом кратности.

Указание. Пусть кривая имеет вид $f(x,y) = 0$, а прямая $l(x) = 0$. Тогда $f(x,y) = l(x)q(x,y) + r(y), r(y) \neq 0$, $\deg r(y) \leq 2$, и если (x_i, y_i) — точки пересечения $l = 0$ с $f = 0$, то $r(y_i) = 0$.

Упражнение 1.3.13 Система уравнений

$$x_1a_1 + x_2a_2 + x_3a_3 = 0, x_1b_1 + x_2b_2 + x_3b_3 = 0$$

всегда имеет ненулевое решение.

Указание. Три двумерных вектора всегда линейно зависимы.

Упражнение 1.3.14 Если кривая третьего порядка содержит две разных прямых, то она является объединением трех прямых.

Указание. Если она кривая $f = 0$ содержит прямые $l_i = 0$, то $f = l_1q_1$, кривая $q_1 = 0$ содержит все точки прямой l_2, кроме м.б. одной, значит $q_1 = l_2l_3$.

Рассмотрим, кроме данной эллиптической кривой $F(x,y) = 0$, еще две кривые третьего порядка

$$L(x,y) = L_1(x)L_2(x,y)L_3(x,y) = 0, M(x,y) = M_1(x)M_2(x,y)M_3(x,y) = 0,$$

где L_i, M_i — уравнения прямых l_i, m_i соответственно. Заметим, что первая из них *приводимая* кривая, ее множество точек совпадает с объединением трех прямых l_i, причем степень многочлена $L(x,y)$ по переменной y не выше двух. Аналогичные утверждения справедливы и для кривой $M(x,y) = 0$. Допустим, что в приведенной выше конфигурации все точки различны (кроме T_i, равенство которых мы сейчас докажем). Значит, и все прямые различны, иначе бы некоторые из них пересекались бы с данной кривой $F(x,y) = 0$ в четырех точках, что невозможно согласно доказанному выше. Докажем, что $\alpha F + \beta L + \gamma M = 0$ для некоторого ненулевого вектора (α, β, γ). Для этого выберем на прямых m_1, l_1 по точке A_1, A_2 не совпадающей с точками конфигурации, лежащими на этих прямых. С помощью упражнения 1.3.13 найдем ненулевой

вектор (α, β, γ), такой, что кривая $\alpha F + \beta L + \gamma M$ проходит через A_i. Докажем, что на самом деле $\alpha F + \beta L + \gamma M = 0$. Допустим противное. Очевидно, что тогда кривая $\alpha F + \beta L + \gamma M$ является кривой не выше третьего порядка и переменная y входит в ее уравнение не выше чем во второй степени. Так как эта кривая проходит и через точки конфигурации $P_1, P_3, -P_i, P_1 + P_2, P_2 + P_3$, то она содержит прямые m_1, l_1, так как иначе с ними в трех точках. Так как $m_1 \neq l_1$, то $\alpha F + \beta L + \gamma M = M_1 L_1 S$, где уравнение $S = 0$ определяет некоторую прямую. Так как точки $P_1 + P_2, -P_2, P_2 + P_3, T_1 = -(P_1 + (P_2 + P_3))$ не лежат на прямых m_1, l_1, то они лежат на прямой s, значит она совпадает с прямой l_2, проходящей через $P_1 + P_2, -P_2$, с прямой m_2, проходящей через $-P_2, P_2 + P_3$, и с прямой l_3, проходящей через $P_2 + P_3, T_1$, а это невозможно. Поэтому $\alpha F + \beta L + \gamma M = 0$. Очевидно, что $\gamma \neq 0$, так как в противном случае кривые $F = 0, L = 0$ совпадают, что противоречит выбору кривой $F = 0$. Сокращая на γ, получаем, что $\alpha F + \beta L = M$ для некоторых α, β. Так как точка T_1 лежит на кривых $F = 0, L = 0$, то из этого равенства следует, что она лежит и на $M = 0$. Если бы она не совпадала с T_2, то на кривой $M = 0$ лежало бы 9 конечных точек кривой $F = 0$, что невозможно, так как на каждой из прямых $m_i, i > 1$ таких точек не более трех, а на прямой m_1 не более двух таких точек.

Заметим, что в приведенном доказательстве мы предполагали, что на каждой прямой можно всегда найти четвертую точку. Это верно только для полей, содержащих не менеее четырех точек. Но если поле содержит мало точек, мы можем рассмотреть ту же кривую и ту же конфигурацию, в которой мы доказывали, что $T_1 = T_2$, над полем, которое является расширением данного поля. Конфигурация при этом не изменится, но на каждой прямой в плоскости над этим полем будет уже достаточно точек, чтобы сохранить без измененния проведенное выше рассуждение.

Все же приведенное доказательство неполное, так в нем предполагалось, что точки $O, -P_3, P_3$ прямой l_1, точки $-P_1, -P_2, P_1 + P_2$ прямой l_2, точки $P_1, P_2 + P_3, T_1 = -(P_1 + (P_2 + P_3))$ прямой l_3, точки $O, -P_1, P_1$ прямой m_1, точки $-P_3, -P_2, (P_2 + P_3)$ прямой m_2, и точки $P_3, P_1 + P_2$ прямой m_3 попарно не совпадают друг с другом. Доказательство, справедливое и в случае кратных точек, строится по той же схеме, но более сложно в деталях. Если Вы хотите разобраться в нем — читайте дальше. Для краткости будем говорить про каждую из перечисленных точек, что она принадлежит прямым l_i, m_j — тем самым, на которых она лежит по определению, невзирая на то, что эта точка может принадлежать одновременно и еще другим прямым, если она кратная. Далее можно считать, что совпадать (склеиваться) могут только точки лежащие на одной прямой l_i или m_j и тогда эта прямая является не секущей, а касательной к кривой F в этой точке. Действительно, если например совпадают точки прямой l_1 и прямой m_2, не являющиеся их пересечением, то это значит, что, например или $P_3 = -P2$, или $P_3 = P_2 + P_3$, или $O = P_2$, или $O = P_2 + P_3$. Два из этих случаев невозможны, так $P_i \neq O$.

Упражнение 1.3.15 Докажите ассоциативность в случаях $P_2 = \pm P_3, \pm P_1$.

Теперь далее можно предполагать, что ни одна из прямых l_i не совпадает ни с одной из прямых m_j, потому что в противном случае точка прямой l_i, не

лежащая формально на прямой m_j будет совпадать с некоторой точкой прямой m_j, формально не лежащей на l_i (а такой случай мы уже исключили из рассмотрения). Действительно, в противном случае на прямой m_j находилось бы больше точек кривой $F = 0$ с учетом кратности, чем ей положено (две, в случае m_1 и три в остальных случаях.).

Далее понадобятся еще несколько простых утверждений о многочленах от двух переменных.

Упражнение 1.3.16 Проверьте, что если $f_1 = lq_1 + r_1$, $f_2 = lq_2 + r_2$, l —линейный многочлен, $r_i = r_i(x)$, если $l = l(x, y)$, и $r_i = r_i(y)$, если $l = l(x)$, то $f_1 + f_2 = l(q_1 + q_2) + r_1 + r_2$, то $f_1 f_2 = lq_3 + r_1 r_2$, где q_3 — некоторый многочлен.

Указание: примените школьную теорему Безу.

Будем говорить, что точка $P = (x_0, y_0)$ прямой $L(x, y) = 0$ является *точкой кратности* k кривой $F = 0$, если $F(x, y) = L(x, y)Q + R(x)$, и x_0 является корнем той же краткости многочлена $R(x)$, т.е. $R(x) = (x - x_0)^k H(x)$, $H(x_0) \neq 0$. Аналогично определяется кратность точки пересечения кривой $F = 0$ с вертикальной прямой $L(x) = 0$. Для точек, не лежащих на кривой, кратность определяем равной нулю. Двукратные точки называем двойными, а тройных у нас не будет.

Упражнение 1.3.17 Если $F = F_1 F_2$, и P — точка пересечения прямой $L = 0$ с кривыми $F_i = 0$ кратности k_i, то та же точка является точкой пересечения той же прямой с кривой $F = 0$ кратности $k_1 + k - 2$. В частности, если $F = lQ$, l —линейный многочлен, то для прямой $L = 0$ не равной прямой $l = 0$, кратность любой точки P пересечения прямой $L = 0$ с кривой $Q = 0$ равна кратности той же точки пересечения прямой $L = 0$ с кривой $F = 0$, если P не лежит на этой прямой, и на единицу меньше ее, если P лежит и на прямой $l = 0$. Если же прямые $L = 0$ и $l = 0$ совпадают, то упомянутые кратности совпадают всегда.

Если $F = F_1 + F_2$, и P — точка пересечения прямой $L = 0$ с кривыми $F_i = 0$ кратности $k_i, k_2 \geq k_1$ то та же точка является точкой пересечения той же прямой с кривой $F = 0$ кратности k_1.

Указание: примените предыдущее упражнение.

Как и в приведенном выше доказательстве в случае отсутствия кратных точек, выберем кривую $F_0 = \alpha F + \beta L + \gamma M$ проходящую через точки A_i, но точку A_2 выберем так, чтобы она не лежала на прямых m_i (ее можно выбрать, например, на прямой l_1, если на ней есть хотя бы четыре точки). Далее мы докажем, что если кривая F_0 ненулевая, то она совпадает с M, а это невозможно так как M не содержит A_2 по построению. Значит она нулевая, поэтому M можно представить в виде линейной комбинации $\alpha F + \beta L$, как и в том доказательстве. Также, как и в том доказательстве замечаем, что кривая F_0 содержит прямую m_1, откуда $F_0 = m_1 Q$. Надо только уточнить, что прямая m_1 пересекает кривую F не обязательно в двух разных конечных точках, а может быть и в одной, но двойной точке (т.е. касается ее в этой точке). Тогда и кривая F_0 пересекает эту прямую в точках с суммарной кратностью не меньше 3, так она содержит точку A_1, и остальные две (возможно, совпадающие) точки прямой m_1 имеют те же кратности согласно упражнению ведь $L = L_1 L_2 L_3$ имеет для тех же точек те же кратности, так как для двойной точки в ней будут пересекаться две из трех прямых l_i, а кривая $M = M_1 M_2 M_3$ имеет во всех точках прямой m_1 нулевую

кратность относительно ее. Поэтому, согласно упражнению кривая F_0 должна содержать m_1, значит $F_0 = M_1Q$. Докажем, что $Q = M_2R$. Для этого заметим, что сумма кратностей точек пересечения прямой m_2 с кривой второго порядка $Q = 0$ не меньше трех, откуда, аналогично упражнению будет следовать, что Q содержит m_2. Предыдущее утверждение о сумме кратностей очевидно, если все точки m_2 не лежат на m_1, так как согласно упражнению они имеют для кривой Q такую же кратность, как и для кривой F_0, т.е. равную трем. Если же, например точка $P_2 + P_3 = P_1$ (т.е. прямая l_3 в этой точке будет касательной), то эта точка будет двойной точкой пересечения кривой F_0 с прямой l_1 (это проверяется также, как и было сделано выше в случае прямой m_1), значит согласно упражнению она же будет точкой пересечения кривой Q с прямой l_1, т.е. будет лежать на Q, и ее кратность как точки пересечения Q с прямой m_2 будет не меньше единицы, т.е. кратности той же точки пересечения кривой F_0 с прямой m_2 (потому что эта точка не может совпадать с другой точкой прямой m_2, так как она уже совпадает с точкой прямой m_1). Таким образом, и для таких точек кратности относительно кривых Q и F_0 всегда совпадают, и сумма их равна 3, поэтому, как и выше, можно заключить, что $Q = M_1R$. Многочлен R уже линейный, осталось доказать, что он пропорционален M_3. Рассмотрим две точки $P_3, P_1 + P_2$ прямой m_3. Пусть они различны, и например P_3 встречается k раз на прямой l_1. Значит, кратность ее как точки пересечения F_0R с l_1 не меньше k (подобные утверждения уже не раз доказывались выше). Значит, среди прямых m_1, m_2 только $k-1$ проходят через точку P_3, поэтому кратность этой точки пересечения кривой второго порядка $M_1M_2 = 0$ (на самом деле это просто пара прямых) с прямой l_1 равна $k-1$. Применяя упражнение получаем, что кратность той же точки пересечения прямой $R = 0$ с прямой l_1 не меньше 1, значит прямая $R = 0$ содержит P_3. Аналогично, она содержит $P_1 + P_2$ и совпадает с прямой m_3. Рассмотрим последний оставшийся случай, а именно совпадение точек $P_3, P_1 + P_2$ (тогда m_3 является касательной к F в этой точке). Из наложенного выше ограничения следует, что эта точка не лежит на прямых $m_i, i < 3$. Опять проверяем, что кратность этой точки пересечения кривой $F_0 = 0$ с прямой m_3 не меньше 2. Так как $F_0 = M_1M_2R$, а эта точка не лежит на прямых $m_i, i < 3$, то согласно упражнению кратность этой точки как точки пересечения R с m_3 не меньше 2. Применяя аналог упражнения получаем, что и в этом случае прямая $R = 0$ совпадает с прямой m_3. Значит, доказано, что $M = \alpha F + \beta L$. Заметим, что при этом $\alpha\beta \neq 0$, так как если, например, $\beta = 0$, то кривые $M = 0$ и $F = 0$ совпадают, что невозможно по определению кривой F, а если $\alpha = 0$, то кривые $M = 0$ и $L = 0$ совпадают, что невозможно в силу различия прямых m_i и l_j в предположении, что данное поле, а значит и любая прямая содержит более трех точек. Выше мы уже объясняли, как можно избавиться от этого предположения. Поэтому $\alpha\beta \neq 0$, и следовательно F линейно выражается через M, L и L линейно выражается через F, L.

Допустим, что $T_1 \neq T_2$ и получим противоречие, тогда $T_1 = T_2$ и ассоциативность будет доказана. Обозначим T точку пересечения прямых l_3, m_3. Так как

T принадлежит L, M то она принадлежит и F, а значит T совпадает с одной из точек $P_1, P_2 + P_3, T_1$ пересечения $F = 0$ с l_3 и совпадает с одной из точек $P_3, P_1 + P_2, T_2$ пересечения $F = 0$ с m_3. Так как T не может совпадать одновременно с $T_1 \neq T_2$, то в силу симметрии без ограничения общности считаем, что $T \neq T_1$, значит $T = P_1$ или $T = P_2 + P_3$. Опять же в силу симметрии без ограничения общности можно считать, что $T = P_1$. Тогда $T \neq P_3, P_1 + P_2$ в силу действующего во время доказательства предположения о неравенстве точек, лежащих на прямых l_i, m_j, но не на их пересечении. Так как T совпадает с одной из точек $P_3, P_1 + P_2, T_2$ то остается только возможность $T = T_2$. Так как $T = P_1$, то T лежит на пересечении прямых m_1, m_3, поэтому согласно предыдущим упражнениям T не менее чем двукратная точка пересечения прямой l_3 с кривой $F = 0$, так как $F = (1/\alpha)M - (\beta/\alpha)L$. Если бы еще $T = P_2 + P_3$, то T лежала бы еще на m_2, тогда она была бы трехкратной точкой пересечения F и l_3. Но в этом случае точка $T_1 \neq 0, T_1 \neq T$ была бы четвертой точкой кривой $F = 0$ на прямой l_3 с учетом кратности, что невозможно согласно предыдущим упражнениям (если бы $T_1 = O$, то для прямой l_3 сумма кратностей точек пересечения с F была бы не больше двух также согласно предыдущим упражнениям). Если же $T = P_1 \neq P_2 + P_3$, то сумма кратностей точек $T_1, P_2 + P_3 \neq P_1 = T$ не меньше двух, независимо от того совпадают они, или нет, поэтому сумма кратностей всех перечисленных точек пересечения кривой F и прямой l_3 не меньше четырех, что невозможно согласно упомянутым упражнениям (или в случае $T_1 = O$ не меньше трех, что тоже невозможно по аналогичной причине). Во всех случаях получено противоречие, и теорема доказана.

1.3.3 Эллиптические кривые над полями действительных и рациональных чисел

Для кривых над полем действительных чисел, разумеется, справедливо все сказанное выше, и, как показано выше, все использованные наглядные представления и понятия имеют для них (и только для них) самый прямой смысл.

Уравнение, определяющее действительную алгебраическую кривую, в каком то смысле более естественно рассматривать над алгебраически замкнутым расширением поля действительных чисел — полем комплексных чисел. Можно и коэффициенты этого уравнения выбрать из поля комплексных чисел. Тогда получится комплексная эллиптическая кривая, теория которых, тесно связанная с теорией функций комплексного переменного, в особенности теорией эллиптических функций, была предметом изучения Абеля, Эйзенштейна, Якоби, Вейерштрасса и других выдающихся математиков 19 века. В частности, в этой теории дается свое доказательство ассоциативности операции сложения точек на эллиптической кривой. Непосредственных приложений в криптографии эта теория не имеет, и мы отсылаем заинтересовавшегося ей читателя к книгам [42], [56], [44].

Если коэффициенты кривой рациональны, то естественно рассматривать ее

над полем рациональных чисел. Изучение групп рациональных точек таких кривых привело к созданию очень сложной и обширной теории, являющейся важным разделом теории чисел. Хотя в ее создание внесли вклад многие выдающиеся математики, не все вопросы в ней еще получили решение. Представление об этой теории читатель может получить по книгам [54], [42], [56], [40]. В оставшейся части этого раздела мы кратко сформулируем несколько наиболее известных ее результатов, хотя в криптографии они приложений не имеют. О приложениях эллиптических кривых в алгоритмах факторизации натуральных чисел можно прочесть в [17], [60]

Порядком точки P кривой E называется минимальное натуральное число n, такое что $nP = 0$. Если такого числа не существует, то точка имеет бесконечный порядок. Понятие порядка точки на кривой является, конечно, частным случаем понятия порядка элемента в любой группе.

Точки конечного порядка в группе кривой E называются *точками кручения* и образуют подгруппу, называемую *подгруппой кручения.*

Упражнение 1.3.18 Проверьте, что точек второго порядка всегда не более трех. Для кривой

$$Y^2 + Y = X^3 - X^2$$

точки порядка два — это

$$(0, -1), (1, 0), (1, -1).$$

Уже отмечалось, что уравнение Ферма $x^3 + y^3 = 1$ можно свести к виду $y^2 = x^3 - 27/4$.

Упражнение 1.3.19 Воспользовавшись тем, что уравнение Ферма не имеет нетривиальных рациональных решений, докажите что кривая Ферма $y^2 = x^3 - 27/4$ имеет только три рациональные точки $(3, \pm 9/2)$ и O. Докажите, что $(3, \pm 9/2)$ имеют порядок 3.

Упражнение 1.3.20 Пусть $P = (0, 0)$ точка кривой $y^2 + y = x^3 - x$. Проверьте, что если $nP = (x, y)$, то $(n+1)P = \left((\frac{y}{x})^2 - x, -(\frac{y}{x})^3 + y - 1\right)$ и пользуясь этим, подсчитайте, что

$$2P = (1, 0), 3P = (-1, -1), 4P = (2, -3), 5P = (\frac{1}{4}, -\frac{5}{8}), 6P = (6, 14),$$

$$7P = (-\frac{5}{9}, \frac{8}{27}), 8P = (\frac{21}{25}, -\frac{69}{125}).$$

Можно доказать, что группа рациональных точек этой кривой бесконечна и порождается точкой $(0, 0)$ (эта точка — отнюдь не равна точке O — нулевому элементу указанной группы).

Известна теорема, доказанная независимо Элизабет Лутц и Торальфом Нагелем, характеризующая рациональные точки порядка большего 2 на рациональной эллиптической кривой $y^2 = x^3 + Ax + B$ следующим образом. Если точка (x, y) имеет конечный порядок, больший 2, то x, y — целые, и y^2 делит $4A^3 + 27B^2$. На самом деле в теореме доказывается больше, например следующее утверждение. Если для рациональной эллиптической кривой общего вида простое число p не делит ее дискриминат Δ, а в случае $p = 2$ еще и $a_1 = 0$ (кривая суперсингулярна), то при отображении редукции по модулю p (когда коэффициенты и точки кривой заменяются на элементы поля $GF(p)$ по правилу $(a/b)_p = a_p/z_p$, где для любого целого a a_p есть элемент поля $GF(p)$, соответствующий остатку от деления a на p) все точки кручения данной рациональной кривой переходят в разные точки кривой над полем $GF(p)$, являющейся редукцией данной кривой по модулю p.

Пример 1.3.4 При редукции по модулю два рациональной эллиптической кривой

$$Y^2 + Y = X^3 - X^2$$

получается кривая

$$Y^2 + Y = X^3 + X^2$$

над полем $GF(2)$. Эта кривая состоит из пяти точек (с учетом бесконечно удаленной точки O.) Значит группа этой кривой изоморфна Z_5 — циклической группе пятого порядка. Так как группа кручения этой же кривой над полем Q содержит точки

$$(0,0),(0,-1),(1,0),(1,-1),$$

то согласно теореме Лутц-Нагеля эта группа тоже изоморфна Z_5.

Упражнение 1.3.21 Рациональная эллиптическая кривая E

$$Y^2 + Y = X^3 - X$$

содержит точки

$$(\pm 1,0),(\pm 1,-1),(2,2),(2,-3),(6,-15)$$

и имеет дискриминант 37.

Упражнение 1.3.22 Проверьте, что над полем $GF(2)$ она имеет пять, а над полем $GF(3)$ — семь точек. Поэтому согласно теореме Лутц-Нагеля группа кручения изоморфна подгруппе Z_5 и подгрупе Z_7, значит она тривиальна. Поэтому все перечисленные выше точки имеют бесконечный порядок.

Полностью возможный вид группы кручения произвольной рациональной кривой дается следующей очень трудной теоремой, доказанной в 1976 г. американским математиком Б.Мазуром. Эта группа изоморфна одной из 15 групп: циклическим группам порядка от 1 до 12 и группам вида $Z_2 \oplus Z_n$, n =2,3,4, где Z_n — циклическая группа порядка n, а знак $\oplus$ обозначает операцию *прямой суммы* двух групп.

Группа всех рациональных точек может быть бесконечной, но в этом случае, согласно гипотезе Пуанкаре, высказанной в 1901 г. и доказанной спустя 20 лет Морделлом, она является *конечно-порожденной*, т.е. существует конечное множество точек такое, что любая рациональная точка кривой может быть получена из данного множества с помощью операции сложения. На языке теории абелевых групп это означает, что группа рациональных точек любой рациональной кривой изоморфна группе $Z^r \oplus T$, где T — ее подгруппа кручения. Число r называется *рангом* кривой. До сих пор неизвестно, может ли он быть сколь угодно большим.

Упражнение 1.3.23 Рациональная кривая содержит бесконечное число рациональных точек если и только если ее ранг больше нуля.

Упражнение 1.3.24 Докажите, что координаты любой рациональной точки (x,y) на эллиптической кривой $y^2 = x^3 + ax + b$ имеют вид несократимых рациональных дробей $x = m/e^2, y = n/e^2$.

Интересно, что известная со времен древних греков задача о конгруэнтных числах сводится к до сих пор не получившим полного решения задачам о рациональных кривых. Рациональное число s называется *конгруэнтным*, если существует прямоугольный треугольник с рациональными сторонами и площадью s.

Упражнение 1.3.25 Проверьте, что для описания множества всех конгруэнтных чисел достаточно описать все конгруэнтные *натуральные числа, свободные от квадратов* (т.е. не делящиеся на квадрат натурального числа, большего 1.)

Поэтому далее рассматриваем только свободные от квадратов числа.

Еще древним египтянам фактически было известно, что 6 — конгруэнтное число.

Упражнение 1.3.26 Докажите, что Вы не глупее египтян.

В тринадцатом веке Фибоначчи, решая задачу на турнире, доказал что 5 — конгруэнтное число.

Упражнение 1.3.27 Докажите, что Вы не глупее средневековых итальянцев.

Эйлер доказал, что 7 тоже конгруэнтное число.

Упражнение 1.3.28 Попробуйте-ка посоревноваться с Эйлером.

Но Ферма доказал, что 1 и 2 — не конгруэнтные числа.

Упражнение 1.3.29 Докажите это.

Указание. Ферма свел задачу к вопросу о разрешимости в натуральных числах уравнения $x^4 = z^2 + y^4$.

В общем случае справедлив следующий неэффективный критерий конгруэнтности. Число n конгруэнтно если и только если найдется такое x, что $x, x+n, x-n$ являются квадратами рациональных чисел.

Упражнение 1.3.30 Докажите это.

Указание. Если X, Y, Z — стороны пифагорова треугольника площади n, то при $x = (Z/2)^2$ числа $x, x \pm n$ — квадраты рациональных чисел. Обратно, если числа $x, x \pm n$ — квадраты рациональных чисел, то $X = \sqrt{x+n} - \sqrt{x-n}, Y = \sqrt{x+n} + \sqrt{x-n}, Z = 2\sqrt{x}$ — стороны пифагорова треугольника с площадью n.

Предыдущее упражнение подсказывает рассмотреть кривую $y^2 = x^3 - n^2 x$.

Упражнение 1.3.31 Проверьте, что если n конгруэнтно, то на кривой $y^2 = x^3 - n^2 x$ есть хоть одна точка, порядка большего 2. Точки порядка 2 на этой кривой есть $(\pm n, 0)$ и $(0, 0)$.

Довольно трудно доказывается следующая теорема: кроме трех точек порядка два и точки O на кривой $y^2 = x^3 - n^2 x$ больше нет точек конечного порядка. Из этой теоремы можно вывести, что

Упражнение 1.3.32 Число n конгруэнтно если и только если ранг кривой $y^2 = x^3 - n^2 x$ положителен.

Используя этот критерий и сложные методы теории модулярных форм, Таннел в 1983 г. доказал, что если n конгруэнтно, то в случае нечетного n количество целочисленных решений уравнения $n = 2x^2 + y^2 + 32z^2$ равно половине количества целочисленных решений уравнения $n = 2x^2+y^2+8z^2$, а для четного n в аналогичном утверждении уравнения заменяются на $n/2 = 4x^2 + y^2 + 32z^2$ и $n/2 = 4x^2 + y^2 + 8z^2$. Он также доказал, что верно и обратное утверждение, но для этого ему пришлось воспользоваться пока не доказанной гипотезой Берча и Свиннертон-Дайера. Подробнее об этом можно прочитать в книге [42] и частично в [40].

Использование теории эллиптических кривых помогло американским математикам Уайлсу и Тейлору доказать большую теорему Ферма. Представление о некоторых применяемых в этом доказательсте методах можно получить по [40], [56]. Но полностью доказательство ввиду его сложности в доступной литературе пока не опубликовано.

Однако некоторые частные случаи теоремы Ферма могут быть сравнительно несложно сведены к задачам о конкретных рациональных эллиптических кривых. Как это сделать в случае $n = 3$, кратко указывалось выше (подробности см. в [40]). Но можно это сделать и в случае $n = 7$, как показал N.D. Elkies в работе «The Klein Quartic in Number Theory» из сборника « The Eightfold Way», MSRI Publications, V.35,1998. У нас нет места для изложения его рассуждений, но желающим мы можем предложить попробовать свои силы в решении следующих трудных упражнений.

Упражнение 1.3.33 Неразрешимость уравнения Ферма $x^7 + y^7 = z^7$ в ненулевых целых числах можно свести к неразрешимости в ненулевых целых числах уравнения Клейна $X^3Y + Y^3Z + Z^3X = 0$, сделав подстановку $X = x^3z, Y = y^3x, Z = z^3y$. Разрешимость уравнения Клейна в ненулевых целых числах равносильна разрешимости в ненулевых рациональных числах уравнения $x^3y + y^3 + x = 0$.

Упражнение 1.3.34 Выразите симметрический многочлен

$$(X^3Y + Y^3Z + Z^3X)(X^3Z + Y^3X + Z^3Y)$$

через

$$s_1 = X + Y + Z, s_2 = XY + XZ + YZ, s_3 = XYZ,$$

и пользуясь полученной формулой докажите, что если уравнение Клейна имеет решение в ненулевых целых числах, то уравнение

$$s_2^4 + s_3(s_1^5 - 5s_1^3s_2 + s_1s_2^2 + 7s_1^2s_3)) = 0$$

тоже имеет решение в ненулевых целых числах.

Упражнение 1.3.35 Если уравнение

$$s_2^4 + s_3(s_1^5 - 5s_1^3s_2 + s_1s_2^2 + 7s_1^2s_3)) = 0$$

имеет решение в ненулевых целых числах, то уравнение

$$s_2^4 + s_3(1 - 5s_2 + s_2^2) + 7s_3^2 = 0$$

имеет решение в ненулевых рациональных числах.

Упражнение 1.3.36 Если уравнение

$$s_2^4 + s_3(1 - 5s_2 + s_2^2) + 7s_3^2 = 0$$

имеет решение в ненулевых рациональных числах, то подобное же решение имеет уравнение

$$1 + u(1 - 5v^{-1} + v^{-2}) + 7u^2 = 0.$$

Упражнение 1.3.37 Если уравнение

$$1 + u(1 - 5v^{-1} + v^{-2}) + 7u^2 = 0.$$

имеет решение в ненулевых рациональных числах, то подобное же решение имеет уравнение

$$y^2 = -28u^3 + 21u^2 - 4u.$$

Отстутсвие ненулевых рациональных решений у последнего уравнения N.D. Elkies доказал совершенно элементарно (хотя и не просто).

1.3.4 Эллиптические кривые над конечными полями

Эллиптические кривые над конечными полями имеют, естественно, конечные группы точек. Порядок этой группы будем называть *порядком эллиптической кривой*. Напомним, что *порядком точки* P эллиптической кривой называется наименьшее число k такое, что $kP = O$. По теореме Лагранжа порядок точки делит порядок эллиптической кривой.

При определении порядка кривой ее можно заменить на удобную изоморфную ей кривую, так как у изоморфных кривых порядки очевидно одинаковы. Менее очевидно, что и их группы изоморфны, но это верно, поэтому далее всегда можно ограничится рассмотрением кривых с уравнениями специального вида, указанного в предыдущих разделах.

Для небольших полей вычисление группы точек данной кривой и ее порядка не составляет труда.

Пример 1.3.5 Например, кривая $y^2 = x^3 + x$ над полем $GF(23)$ состоит из 23 точек

$$(0,0), (1,5), (1,18), (9,5), (9,18), (11,10), (11,13), (13,5), (13,18),$$

$$(15,3), (15,20), (16,8), (16,15), (17,10), (17,13), (18,10),$$

$$(18,13), (19,1), (19,22), (20,4), (20,19), (21,6), (21,17).$$

Учитывая бесконечно удаленную точку, получаем что порядок этой кривой равен 24. Для составления списка точек достаточно перебирать элементы $a = 0, 1, \ldots, 22$ поля и для каждого из них для нахождения y нужно решать в этом поле уравнение $y^2 = a^3 + a$, т.е. извлекать квадратный корень. Достаточно найти только один корень y, второй корень вычисляется по формуле $-y \bmod 23 = 23 - y$. Квадратные корни существуют однако не всегда.

Квадратичные вычеты и извлечение квадратных корней в конечных полях. Число a, $a \not\equiv 0 \pmod p$, называется *квадратичным вычетом по модулю* p, если существует такое b, что $b^2 \equiv a \pmod p$ и *квадратичным невычетом по модулю* p, если такого b не существует. Таким образом, квадратные корни извлекаются только из квадратичных вычетов и из нуля.

Таблицу квадратных корней можно составить заранее. Ее легко составить, если есть таблицы логарифмов и антилогарифмов по произвольному примитивному элементу этого поля.

Упражнение 1.3.38 Докажите, что для одного из квадратных корней в конечном поле нечетного порядка справедливо равенство

$$\log_\alpha \sqrt{x} = \frac{1}{2} \log_\alpha x \bmod (p-1),$$

где p — порядок поля, а деление на два выполняется по модулю $p-1$, .

Однако для больших полей такие таблицы громоздки, да и их вычисление затруднительно, так как проблема дискретного логарифмирования в больших конечных полях труднорешаема (на этом факте и основана почти вся криптография с открытым ключом). Гораздо проще непосредственно вычислять квадратные корни.

Известен алгоритм извлечения квадратных корней в конечном поле Тонелли, позднее переоткрытый Д.Шенксом. Его мы изложим в виде задач, решения которых можно найти в учебнике И.М.Виноградова по теории чисел (они там содержались задолго до появления работы Д.Шенкса).

Упражнение 1.3.39 Докажите, что при $p = 4k+3$, если a является квадратичным вычетом по модулю p, то квадратным корнем из a по модулю p является $\pm a^{k+1}$.

Упражнение 1.3.40 Докажите, что при $p = 8k+5$, если a является квадратичным вычетом, по модулю p, то квадратным корнем из a по модулю p является $\pm a^{k+1}$. $\pm a^{k+1} 2^{(2k+1)s}$ где $s = 0$ или 1.

???????Пропадает изображение за пределами полей страницы@@@@@@@@@@@@@@@@

Упражнение 1.3.41 Пусть $p = 8k+1$ и N — некоторый квадратичный невычет по модулю p и a — произвольный квадратичный вычет. Обоснуйте следующий быстрый метод извлечения квадратного корня из a по модулю p. Представим p в виде $2^k h + 1$, $k \geq 3$, $h = 2m+1$. Тогда $a^{2^{k-1}h} \bmod p = 1$, $N^{2^{k-1}h} \bmod p = -1$, $a^{2^{k-2}h} \bmod p = \pm 1$, поэтому при некотором $s_2 = 0$ или 1 $a^{2^{k-2}h} N^{s_2 2^{k-1}h} \bmod p = 1$, $a^{2^{k-3}h} N^{s_2 2^{k-2}h} \bmod p = \pm 1$, поэтому при некотором $s_3 = 0, 1$ или 3 $a^{2^{k-3}h} N^{s_3 2^{k-2}h} \bmod p = 1$, $a^{2^{k-4}h} N^{s_3 2^{k-3}h} \bmod p = \pm 1$, и т.д., пока не получим при некотором $0 \leq s_k \leq 2^{k-1} - 1$ равенство $a^h N^{s_k 2h} \bmod p = 1$, значит искомым корнем будет $x = \pm a^{(h+1)/2} N^{s_k h}$.

Указанный алгоритм имеет сложность $O(\log_2 p)^4$, если нам заранее известен невычет N. Если он неизвестен, его можно найти перебором, если научится быстро определять по числу, является ли оно вычетом или невычетом по данному модулю. Как это делать, указано ниже. Американский математик Бах [82] доказал в предположении справедливости расширенной гипотезы Римана,

что наименьший квадратичный невычет по модулю p не превосходит $2\ln^2 p$ при $p > 1000$. Поэтому, вероятно, его тоже можно найти со сложностью $O(\log_2 p)^4$, если применить теорию квадратичных вычетов Эйлера-Лежандра-Гаусса, которую далее мы кратко излагаем в виде задач (в основном, очень трудных). За более подробным изложением отсылаем читателя к учебнику И.М.Виноградова или к более сложным книгам [4], [64].

Упражнение 1.3.42 В множестве Z_p^*, $p > 2$, поровну квадратичных вычетов и невычетов. Произведение вычетов есть вычет, произведение двух невычетов — опять вычет, а произведение вычета на невычет — невычет. Число $p-1$ является вычетом тогда и только тогда, когда $p = 4k+1$.

По определению, *символ Лежандра* $\left(\frac{a}{p}\right)$ равен 1, если a — вычет, и равен минус 1, если a — невычет, $\left(\frac{a}{p}\right) = 0$, если a кратно p.

Упражнение 1.3.43 Докажите, что $\left(\frac{a}{p}\right)$ и $a^{\frac{p-1}{2}}$ имеют равные остатки при делении на p и $\left(\frac{a}{p}\right)\left(\frac{b}{p}\right) = \left(\frac{ab}{p}\right)$.

Упражнение 1.3.44 Докажите, что:

а) для любых a и x, не кратных p, остаток от деления ax на p равен остатку от деления $\epsilon_x r_x$ на p, где

$$1 \le r_x \le (p-1)/2, \varepsilon_x = (-1)^{\lfloor 2ax/p \rfloor},$$

б) для любого a, не кратного p, справедливо равенство

$$\left(\frac{a}{p}\right) = (-1)^s,$$

где

$$s = \sum_{x=1}^{(p-1)/2} \lfloor 2ax/p \rfloor.$$

Упражнение 1.3.45 Докажите, что

$$\left(\frac{1}{p}\right) = (-1)^{\frac{p-1}{2}} = \sin\frac{\pi p}{2}.$$

$$\left(\frac{2}{p}\right) = (-1)^{\frac{p^2-1}{8}} = \sqrt{2}\sin\frac{\pi p}{4}.$$

Упражнение 1.3.46 Докажите, что для любых простых нечетных p и q

$$\left(\frac{q}{p}\right)\left(\frac{p}{q}\right) = (-1)^{\frac{p-1}{2}\frac{q-1}{2}}.$$

Для более изящной формы записи предыдущей теоремы Гаусса ?????????????????????? понадобится следующее определение. Для любого нечетного числа a выберем a^* так, чтобы $a = \pm a^*$ и a^* имело бы вид $4k+1$, где k— целое.

Упражнение 1.3.47 Докажите, что
а) для любых нечетных a, b

$$ab^* = a^*b^*,\ a = (-1)^{\frac{a-1}{2}}a^*$$

б) для любого простого $p > 2$

$$\left(\frac{2}{p}\right) = (-1)^{\frac{p^*-1}{4}}, \left(\frac{-2}{p}\right) = (-1)^{\frac{p-1}{2}+\frac{p^*-1}{4}},$$

в) для любых простых $p, q > 2$

$$\left(\frac{q}{p}\right) = \left(\frac{p^*}{q}\right).$$

Упражнение 1.3.48 Докажите, что

$$\left(\frac{-3}{p}\right) = \frac{2}{\sqrt{3}}\sin\frac{2\pi p}{3}.$$

Упражнение 1.3.49 Докажите, что при $p = 4k+1$ квадратным корнем по модулю p из -1 является $\pm 1 \cdot 2 \cdots 2k$.

Квадратичный закон взаимности позволяет быстро выяснять, является ли данное число a квадратичным вычетом по простому модулю p. Для этого нужно вычислить символ Лежандра $\left(\frac{a}{p}\right)$. Если мы умеем разложить число a на простые множители $a = p_1^{\alpha_1} \ldots p_m^{\alpha_m}$, то это можно сделать с помощью многократного применения следующей задачи и закона взаимности.

Упражнение 1.3.50 Докажите, что

$$\left(\frac{a}{p}\right) = \left(\frac{p_1}{p}\right)\cdots\left(\frac{p_m}{p}\right).$$

Применение закона взаимности сводит вычисление $\left(\frac{p_i}{p}\right)$ к вычислению $\left(\frac{p}{p_1}\right) = \left(\frac{r}{p_1}\right)$, где $r = p \bmod p_1$, и указанная выше процедура опять применяется, но к уже меньшим числам.

Поскольку неизвестен быстрый алгоритм разложения на простые множители, указанный способ, также как и аналогичный способ нахождения наибольшего общего делителя двух чисел с известными разложениями на простые множители, применяется только при устном счете.

Символ Якоби и алгоритм Кронекера. К счастью, есть быстрый алгоритм вычисления символов Лежандра, основанный на применении его обобщения — символа Якоби, и очень напоминающего алгоритм Евклида.

Введем символ Кронекера-Якоби $\left(\frac{a}{b}\right)$ следующим образом:

1) Если $b = 0$, то при $a = \pm 1$ символ $\left(\frac{a}{0}\right) = 1$, и равен 0 в противном случае.

2) Если $b \neq 0$, то запишем его в виде произведения $b = \prod p_i$ не обязательно различных простых чисел p_i и, может быть, одной минус единицы для представления отрицательных чисел и определим символ Кронекера-Якоби $\left(\frac{a}{b}\right)$ как произведение $\prod\left(\frac{a}{p_i}\right)$ символов Лежандра, а также символов

$$\left(\frac{a}{2}\right) = \begin{cases} 0, & \text{если } a \text{ четно} \\ (-1)^{(a^2-1)/8}, & \text{если } a \text{ нечетно} \end{cases}$$

и

$$\left(\frac{a}{-1}\right) = \begin{cases} 1, & \text{если } a \geq 0 \\ -1, & \text{если } a < 0. \end{cases}$$

Упражнение 1.3.51 Докажите, что

а) $\left(\frac{a}{b}\right) = 0$ если и только если $(a, b) > 1$.

б) Для любых целых a, b, c справедливы мультипликативные свойства символа Кронекера-Якоби

$$\left(\frac{ab}{c}\right) = \left(\frac{a}{c}\right)\left(\frac{b}{c}\right), \ \left(\frac{a}{bc}\right) = \left(\frac{a}{b}\right)\left(\frac{a}{c}\right), \text{если } bc \neq 0.$$

Упражнение 1.3.52 Докажите, что

а) При фиксированном целом $b > 0$ символ Кронекера-Якоби как функция «числителя» a является периодической с периодом b в случае $b \bmod 4 \neq 2$ и периодом $4b$ в противном случае.

б) При фиксированном целом $a \neq 0$ символ Кронекера-Якоби как функция «знаменателя» b является периодической с периодом $|a|$ в случае $a \bmod 4 = 0$ или 1, и с периодом $4|a|$ в противном случае.

Упражнение 1.3.53 Докажите, что следующие равенства справедливы для любых нечетных натуральных чисел p, q

$$\left(\frac{-1}{p}\right) = (-1)^{\frac{p-1}{2}},$$

$$\left(\frac{2}{p}\right) = (-1)^{\frac{p^2-1}{8}},$$

$$\left(\frac{q}{p}\right)\left(\frac{p}{q}\right) = (-1)^{\frac{p-1}{2}\frac{q-1}{2}}.$$

Упражнение 1.3.54 Следующие равенства справедливы для любых нечетных целых чисел p, q при $q > 0$

$$\left(\frac{q}{p}\right)\left(\frac{p}{|q|}\right) = (-1)^{\frac{p-1}{2}\frac{q-1}{2}}.$$

Упражнение 1.3.55 Алгоритм Кронекера.

а) *(Проверка b на равенство нулю)* Если $b = 0$ тогда $\left(\frac{a}{b}\right) = 1$ если $|a| = 1$ и $\left(\frac{a}{b}\right) = 0$ в противном случае.

б) *(Удаление двоек из b)* Если a, b оба четные, то $\left(\frac{a}{b}\right) = 0$. В противном случае полагаем v равным наибольшей степени двойки, делящей b и заменяем b на $2^{-v}b$. Если v четно, полагаем $k = 1$, иначе берем $k = (-1)^{(a^2-1)/8}$. Если $b < 0$, то меняем перед ним знак, а если к тому же $a < 0$ меняем знак у k.

в) *(Проверка a на равенство нулю)* Если $a = 0$, то при $b > 1$ выход алгоритма равен 0, а при $b = 1$ выход равен k и работа закончена.

г) *(Удаление двоек из a)* В случае $a \neq 0$ полагаем v равным наибольшей степени двойки, делящей a и заменяем a на $2^{-v}a$. Если v четно, умножаем k на $(-1)^{(b^2-1)/8}$.

д) *(Применение закона двойственности и шаг Евклида)* Умножаем k на $(-1)^{(a-1)(b-1)/4}$, полагаем $r = |a|$, заменяем a на $b \bmod r$ и b на r и возвращаемся к пункту в).

Докажите, что этот алгоритм действительно вычисляет символ Кронекера-Якоби.

Упражнение 1.3.56 Покажите, что для вычисления $(-1)^{(a^2-1)/8}$ достаточно найти $a \bmod 8$ и воспользоваться таблицей $\{0, 1, 0, -1, 0, -1, 0, 1\}$. Для вычисления $(-1)^{\frac{(a-1)(b-1)}{4}}$ достаточно найти $a \bmod 4$, $b \bmod 4$ и воспользоваться заранее вычисленной таблицей.

Применяют алгоритм Кронекера в основном для вычисления символов Лежандра, но в промежуточных вычислениях возникают символы Кронекера-Якоби, благодаря использованию которых фактически удается применить даже некоторый ускоренный вариант алгоритма Евклида. При использовании только лишь символов Лежандра применить алгоритм Евклида до конца редко удается и приходится использовать разложение на множители промежуточных результатов, что легко выполнить при работе с маленькими числами и практически невозможно с большими.

Упражнение 1.3.57 Докажите, что битовая сложность алгоритма Кронекера, применяемого к n-разрядным числам, равна $O(n^2)$.

Упражнение 1.3.58 Докажите, что на последнем шаге алгоритма Кронекера вместо $a = b \bmod r$ в случае $a > r/2$ можно взять $a - r$. Тогда всегда будет выполняться неравенство $0 < a \leq b/2$ и алгоритм будет сходится несколько быстрее.

Упражнение 1.3.59 Покажите по аналогии с бинарным алгоритмом Евклида как модифицировать алгоритм Кронекера так, чтобы в нем не выполнялись операции деления с остатком, а только операции вычитания.

Теорема Хассе и порядок группы точек эллиптической кривой над конечным полем. Пользуясь символом Лежандра, легко указать формулу для числа точек на кривой $y^2 = f(x)$. Действительно, число решений сравнения $y^2 \equiv f(x) \pmod{p}$ относительно y при фиксированном x равно $1 + \left(\frac{f(x)}{p}\right)$ (это верно и при $f(x) = 0$). Учитывая бесконечно удаленную точку, получаем формулу для порядка кривой в виде

$$p + 1 + \sum_{x=0}^{p-1} \left(\frac{f(x)}{p}\right).$$

При малых простых p, пользуясь этой формулой и теорией квадратичных вычетов порядок кривой над полем $GF(p)$ находить довольно легко.

Упражнение 1.3.60 Найдите порядок кривой $y^2 = x^3 + x + 4 \bmod 23$.

Но вычисление порядка эллиптической кривой не всегда просто или даже возможно. Общая формула для порядка кривой неизвестна. Неизвестно даже, можно ли за полиномиальное время найти кривую данного порядка. Некоторые результаты по этому поводу можно найти в [171]. В [118] показано, что задача вычисления порядка эллиптической кривой над кольцом вычетов по модулю n полиномиально эквивалентна задаче разложения числа n на множители, но эта эквивалентность доказана в классе вероятностных алгоритмов.

Тем не менее известны способы выбора эллиптических кривых над конечными полями, допускающих простое определение порядка. Эти способы важны, потому что в криптографическом отношении полезными являются эллиптические кривые, порядок которых содержит большие простые множители. Для кривых, у которых порядок является «гладким» числом (т.е. разлагающимся только на малые простые) проблема дискретного логарифмирования может быть решена сравнительно быстро алгоритмом Полига-Хеллмана-Зильбера, еще раньше найденном, но не опубликованном в открытой печати В.И.Нечаевым (см., например [52], [68], [17]).

Известна асимптотически точная формула для порядка эллиптической кривой над конечным полем. Она была найдена в тридцатые годы немецким математиком Гельмутом Хассе. По теореме Хассе порядок N эллиптической кривой над полем $GF(q)$ удовлетворяет неравенству

$$|N - q - 1| \leq 2\sqrt{q}.$$

Это эквивалентно системе неравенств

$$q + 1 - 2\sqrt{q} \leq N \leq q + 1 + 2\sqrt{q}.$$

Теорема Хассе в случае простого конечного поля кажется интуитивно очевидной, так как вычеты и невычеты по простому модулю распределены в определенном смысле равномерно и в сумме

$$\sum_{x=0}^{p-1} \left(\frac{f(x)}{p}\right)$$

слагаемые ± 1 ведут себя подобно случайному блужданию по прямой. Однако все известные ее доказательства очень сложны, даже элементарное доказательство Ю.И.Манина в случае простых q, которое можно найти в [40] и в [26].

Справедлива и более общая теорема Хассе-Вейля

Пусть $\mathcal{E}$ – эллиптическая кривая над полем $GF(q)$ и N – порядок ее группы. Тогда для порядка $N(n)$ группы эллиптической кривой $\mathcal{E}(\mathcal{GF}(Ш^{\backslash}))$ над полем $GF(q^n)$ справедлива формула

$$N(n) = q^n + 1 - \alpha^n - \beta^n,$$

где α и β – корни квадратного уравнения $x^2 - tx + q = 0$, в котором коэффициент $t = q + 1 - N$. Всегда выполняется неравенство $t^2 \leq 4q$ и в случае строгого неравенства корни квадратного уравнения α и β будут комплексно сопряженными.

Эта теорема была после войны обобщена на широкий класс алгебраических кривых А.Вейлем, а в семидесятые годы ее обобщение на произвольные алгебраические многообразия было получено П.Делинем. Эти исключительно трудные и неэлементарные результаты мы можем здесь только упомянуть. Заметим однако, что сравнительно элементарное (но тоже весьма сложное) доказательство результатов Вейля получил С.А.Степанов [61].

В случае полей малой характеристики порядок группы эллиптической кривой легко найти с использованием теоремы Хассе.

Упражнение 1.3.61 Проверьте, что для кривой $y^2 = x^3 + 2x^2 + 1$ над полем $GF(3)$

$$N = N(1) = 5, t = 4 - N = -1,$$

$$N(n) = 3^n + 1 - \alpha^n - \beta^n = 3^n + 1 - ((1 + \sqrt{11}i)/2)^n - ((1 - \sqrt{11}i)/2)^n.$$

Непосредственно убедитесь, что эта формула верна для $n = 2$.

Иногда можно точно вычислить порядок группы эллиптической кривой и для полей большой характеристики. Например, при $q = p^d$, p не кратном $2n$, и $q \equiv 3 (\mathrm{mod}\ 4)$ порядок кривой $y^2 = x^3 - n^2 x$ равен $q + 1$.

Упражнение 1.3.62 Докажите это.

Указание. Точки порядка 2 — это $(0,0), (\pm n \bmod p, 0)$ и O. Разобьем $x \neq 0, \pm n \bmod p$ на пары $\{x, -x\}$. Так как $f(x) = x^3 - n^2 x$ нечетная функция и (-1) не является квадратом в поле $GF(q)$ (так как иначе $(-1)^{(q-1)/2} = 1$ согласно теореме Ферма, что невозможно при $q \equiv 3 (\mathrm{mod} 4)$, то только один из двух элементов $f(x), f(-x) = -f(x)$ является квадратом (так как в поле $GF(q)$ произведение квадратов — квадрат, и произведение неквадратов — тоже квадрат), поэтому каждая пара дает пару точек кривой.

Пусть u — произвольный квадратичный невычет в поле $GF(q)$, q нечетно. Тогда кривая $E' : y^2 = x^3 + u^2 ax + u^3 b$ называется *скручиванием* кривой $E : y^2 = x^3 + ax + b$.

Упражнение 1.3.63 Докажите, что сумма порядков кривой и ее скручивания равна $2q + 2$.

Указание. Пусть $f(x) = x^3 + ax + b$. Когда x пробегает элементы поля $GF(q)$, то x/u тоже пробегает все это поле и каждому корню многочлена $g(x)$ соответствует одна и та же точка точка на обоих кривых. Каждому значению $f(x)$, которое есть квадратичный вычет, соответствует две точки на E и ни одной на E' так как u^3 — квадратичный невычет. Аналогично, каждому значению $f(x)$, которое есть квадратичный невычет, соответствует две точки на E' и ни одной на E. Так имеется q значений $f(x)$ с учетом кратности, то общее число конечных точек на обоих кривых равно $2q$.

Благодаря этому факту после того, как найден порядок кривой, порядок ее скручивания находится без вычислений.

Приведем еще несколько примеров кривых, для которых легко вычислить порядок.

Упражнение 1.3.64 Порядок кривой $y^2 = x^3 + b \bmod p$, $p \equiv 2 \bmod 3$, равен $p+1$.

Указание. Так как кубический корень в поле $GF(p)$ всегда существует и однозначно определен, то для каждого y на кривой лежит ровно одна точка $((y^2-b)^{1/3}, y)$.

Упражнение 1.3.65 Порядок кривой $y^2 = x^3 + ax \bmod p$, где $p \equiv 3 \bmod 4$), $\frac{a}{p} = 1$, равен $p+1$.

Указание. Так как -1 есть квадратичный невычет в $GF(p)$, то каждая пара $\{x, -x\}, x \neq 0$, дает два решения:

$$(x, (x^3+ax)^{1/2}), (x, -(x^3+ax)^{1/2})$$

или

$$(x, (-x^3-ax)^{1/2}), (x, -(-x^3-ax)^{1/2})$$

в зависимости от того, будет ли квадратичным вычетом x^3+ax или нет. Уравнение $x^3+ax = 0 \bmod p$ имеет только одно решение.

Для определения порядка группы эллиптической кривой над полем $GF(q)$ известен алгоритм R.Schoof'a [148], [149] и его варианты, известные как SEA (Schoof-Elkies-Atkin) алгоритм, указанные в работах Elkies, Atkin, Mueller, Charlap, Coley, Robbins, Dewaghe, Morain, Lercier и др. (см. например, [94], [122], [120]), в которых оценка его сложности уменьшилась с $O(\log_2 q)^8$ до $O(\log_2 q)^6$ и он доведен до состояния, позволившего в работе [136] вычислить порядок кривой над полем $GF(p)$ для 500-значного простого p. На русском языке изложение алгоритма Schoof'a для простых конечных полей недавно появилось в [17] в связи с его применением в алгоритмах разложения целых чисел на множители. Алгоритм Schoof'a был расширен в [116] и на случай четного q, и в ряде работ существенно усовершенствован (см., например [132], [121]). Более быстрый алгоритм недавно появился в [145], [96].

Известен также другой метод определения порядка кривых— метод комплексного умножения [117],[137],[123]. Однако подробное изложение этих методов выходит за рамки книги.

Группы точек эллиптических кривых довольно похожи на мультипликативные группы конечных полей. Они тоже коммутативные и имеют асимптотически такой же порядок. Но операции в них более сложные, и естественно ожидать, что и проблема дискретного логарифмирования в них решается более сложно. Эти ожидания оправдываются, и далее мы об этом скажем подробнее. Но с абстрактной алгебраической точки зрения группы точек кривых над конечными полями устроены не так сложно, как у рациональных кривых.

В общем случае, эти *группы или циклические (изоморфные Z_m при некотором m), или являются прямыми суммами двух циклических групп $Z_{m_1} \oplus Z_{m_2}$, где m_2 делит m_1 и m_2 делит $q-1$.*

Справедливы и более точные утверждения из [?]

Теорема 1.3.5 *Для порядка эллиптических кривых над полем $GF(q), q = p^n$, справедливы следующие утверждения: порядок имеет вид $q+1-t$, где для t удовлетворяет одному из условий*

1) $t \not\equiv 0 (\bmod\ p)$ и $t^2 \leq 4q$

2) n нечетно и либо $t = 0$, либо $t^2 = pq$ при $p = 2, 3$

3) n четно и либо $t = 4q$, либо $t^2 = q$ при $p \not\equiv 1 (\bmod\ 3)$, либо $t = 0$ при $p \not\equiv 1 (\bmod 4)$. Если p делит t, кривая является суперсингулярной.

Из этой теоремы следует, что для суперсингулярных кривых $t^2 = 0, q, 2q, 3q, 4q$. Структура групп суперсингулярных кривых описывается следующей теоремой [?].

Теорема 1.3.6 *Если $t^2 = q, 2q, 3q$, то группа циклическая. Если $t^2 = 4q$, то группа изоморфна $Z_{\sqrt{q}-1} \oplus Z_{\sqrt{q}-1}$ в случае $t = 2\sqrt{q}$, или изоморфна $Z_{\sqrt{q}+1} \oplus Z_{\sqrt{q}+1}$ в случае $t = -2\sqrt{q}$. Если $t = 0, q \not\equiv 3 (\bmod 4)$. то группа циклическая. Если $t = 0, q \equiv 3 (\bmod\ 4)$, то группа или циклическая или изоморфна $Z_{(q+1)/2} \oplus Z_2$*

В [113] доказано, что среди эллиптических кривых по модулю p доля *циклических* (т.е. имеющих циклическую группу) не менее $0.7785 - o(1)$.

Доказательство этих теорем сложное, и поэтому здесь не приводится. Разлагая каждую из циклических групп в произведение групп порядков p^n для разных простых, можно представить группу произвольной кривой единственным образом в виде произведения групп вида

$$Z_{p^\alpha} \oplus Z_{p^\beta},\ \alpha \geq \beta, \alpha > 0.$$

Соответствующий вектор с компонентами

$$(p^\alpha, p^\beta)_{p|N},$$

где N — порядок группы, называется ее *типом*. Он однозначно определяет группу с точностью до изоморфизма.

Упражнение 1.3.66 Докажите, что группы кривых из двух предыдущих упражнений являются циклическими.

Указание. Допустим, что группы нециклические. Тогда для некоторого p в их типах встречаются компоненты $(l^\alpha, l^\beta), \alpha > \beta > 0$, где l — делитель $p-1$. Но так как порядок группы точек равен $p+1$, то l — делитель $p+1$ по теореме Лагранжа. Значит $l = 2$. Но сравнение $x^3 + b = 0 \bmod p$ имеет одно решение. Значит $\beta = 0$.

Пример 1.3.6 Группа кривой $y^2 = x^3 - n^2 x$ над полем $GF(q)$, $q = p^d$, при p, не кратном $2n$, и $q \equiv 7(\ rmmod 8)$ — не циклическая.

Упражнение 1.3.67 Докажите это.

Указание. Согласно одному из предыдущих упражнений, ее порядок равен $q+1$ и кратен 8, а точек порядка 2 в ней ровно 3.

Упражнение 1.3.68 Найдите тип кривой $y^2 = x^3 - x$ над полем $GF(71)$.

Указание. Согласно предыдущему упражнению в ее тип входят числа $4, 2$. Остается проверить, будет ли тип иметь вид $(4, 2, 9)$ или $(4, 2, 3, 3)$. Для различения этих случаев достаточно проверить, что точек порядка 3 меньше 9, поэтому тип будет $(4, 2, 9)$. Для поиска таких точек P заметьте, что $2P = -P$, значит у этих точек равны x-координаты. Получите для них уравнение $3x^4 - 6x^2 - 1 = 0$. Его решать не надо, если заметить, что корни у него распадаются на пары $x, -x$, и так как $x^3 - x$ нечетна, то только один из элементов пары дает две точки кривой, а другой не дает таких точек.

Преимущество эллиптических кривых над конечными полями заключается в том, что имеется большое многообразие групп с разными порядками для одного и того же поля $GF(q)$. Даже доказано в [119], что для любого простого p порядки групп кривых над полем $GF(p)$ почти равномерно распределены на отрезке $[p+1-2\sqrt{p}, p+1+2\sqrt{p}.]$ Это часто дает возможность подобрать кривую, порядок которой имеет только один большой простой делитель.

Глава 2

Неприводимые многочлены

2.1 Тесты и поиск неприводимых многочленов

2.1.1 Зачем нужно искать неприводимые многочлены над конечными полями

Знание неприводимых многочленов нужно уже для того, чтобы строить сами эти поля. Поэтому в любой книге по кодированию есть таблицы таких многочленов невысоких степеней. Среди неприводимых многочленов особый интерес представляют примитивные многочлены, т.е. такие, корни которых являются примитивными (или порождающими) элементами поля разложения этого многочлена. Примитивные элементы являются основаниями дискретных логарифмов, таблицы которых (и примитивных элементов, и логарифмов) также есть в любой книге по кодированию, так как сильно облегчают умножение в конечных полях.

Но представление элементов поля в виде степеней примитивного элемента хотя и облегчает умножение, но сильно затрудняет сложение, поэтому чаще элементы поля представляют в виде векторов, разложенных по стандартному полиномиальному базису, тогда сложение сводится к сложению этих векторов (особенно просто оно выполняется в полях порядка, равного степени двойки, так как сводится к покомпонентной логической операции XOR), а умножение представляет из себя умножение многочленов над исходным конечным полем коэффициентов, выполняемое по модулю неприводимого многочлена, определяющего рассматриваемое представление поля.

Однако во многих случаях важно иметь подобные многочлены с минимальным по возможности числом одночленов, например трехчлены или пятичлены (в полях характеристики два неприводимых четночленов, очевидно, не бывает.) Теоретически такие многочлены предъявить удается не всегда и их приходится строить с помощью компьютера.

Впрочем, для небольших полей компьютер особенно не нужен. Удивительно, но выдающийся специалист по кодированию и конечным полям Элвин Берлекемп (автор известного алгоритма факторизации многочленов над конечными полями) в своей книге [8] написал в шестидесятые годы, что большие конечные поля представляют только академический интерес.

Но сейчас интерес к большим конечным полям, в частности с малой характеристикой, уже не академический. Без этих полей немыслима современная криптография с открытым ключом, также как и построение классических криптосистем.

Примитивные многочлены также находят существенные применения в криптографии, главным образом для построения линейных рекуррентных последовательностей максимально возможной длины, которые применяются, например, при построении датчиков случайных

чисел.

Ввиду важности для приложений, неприводимые и примитивные многочлены уже давно табулированы. Таблицы для небольших степеней имеются почти в любом учебнике теории кодирования. Однако большие таблицы мало доступны для нашего читателя, о чем сетовал В.Жельников в [31]. Они последовательно появлялись в статьях [165, 166, 167, 168, 169]. Фрагменты этих таблиц имеются в доступной через интернет книге [133]. Однако детали алгоритмов составления этих таблиц там не описываются.

Прочитав эту главу, читатель может узнать, как написать программу, быстро генерирующую и тестирующую неприводимые многочлены, и проверяющие их на примитивность.

2.1.2 Тест на неприводимость. Алгоритм Берлекемпа

Алгоритм Берлекемпа ([8]) определяет, является ли данный многочлен произвольного вида над полем Галуа $GF(q)$ неприводимым.

Пусть нужно проверить многочлен $P(x)$ на неприводимость. Сначала вычислим производную $P'(x)$ многочлена $P(x)$.

Примечание. Производная $P'(x)$ образуется по правилу формального дифференцирования многочлена $P(x)$: если $P(x) = \Sigma_{i=0}^{n-1} a_i x^i$, то $P'(x) = \Sigma_{i=1}^{n-1} i * a_i x^{i-1}$.

Если наибольший общий делитель многочленов $P(x)$ и $P'(x)$ не равен 1, то многочлен $P(x)$ не является неприводимым, т.к. его разложение на множители содержит квадрат многочлена (степени большей 0). В противном случае, проверка неприводимости продолжается. При этом вычисляются остатки $P_i(x)$ от деления многочленов x^{2i} на $P(x)$, $i = 1, 2, \ldots, \deg(P(x)) - 1$. Для сокращения времени вычисления эти остатки определяются последовательно. Так, если вычислен $P_i(x)$, то $P_{i+1}(x)$ равен остатку от деления $x^2 P_i(x)$ на $P(x)$. Поэтому сначала вычисляем многочлен $R(x)$, равный остатку от деления $xP_i(x)$ на $P(x)$, а уже затем $P_{i+1}(x)$. Нетрудно видеть, что что $R(x) = xP_i(x)$, если $\deg(P_i(x)) + 1 < \deg(P(x))$ и $R(x) = xP_i(x) + P(x)$, в противном случае. Многочлен $P_{i+1}(x)$ равен остатку от деления $xR(x)$ на $P(x)$. Положим $R_i(x) = P_i(x) + x^i$, $i = 1, 2, \ldots, \deg(P(x)) - 1$. Далее вычислим ранг матрицы $A = (R_1(x), R_2(x), \ldots, R_{\deg(P(x))-1}(x))$. Если он оказался равным $\deg(P(x)) - 1$, то многочлен $P(x)$ неприводим, в противном случае этот многочлен приводим.

Пример 2.1.1 Пусть нужно проверить на неприводимость многочлен $P(x) = x^4 + x + 1$. Имеем $P'(x) = 1$ и наибольший общий делитель многочленов $P(x)$ и $P'(x)$ равен 1. Поэтому, если $P(x)$ приводим, то его разложение $P(x)$ в произведение неприводимых многочленов содержит только сомножители в первой степени (то есть не являющиеся кратными). Далее вычислим остатки $P_1(x), P_2(x), P_3(x)$ от деления многочленов x^2, x^4, x^6, соответственно на $P(x)$. Нетрудно видеть, что $P_1(x) = x^2$, $P_2(x) = x + 1$, $P_3(x) = x^2 P_2(x) (\mathrm{mod} P(x)) = x^3 + x^2$. Поэтому $R_1(x) = x^2 + x$, $R_2(x) = x^2 + 1 + x$, $R_3(x) = x^3 + x^2 + x^3 = x^2$. Таким образом, необходимо вычислить ранг матрицы $A = (R_1(x), R_2(x), R_3(x)) = (x^2 + x, x^2 + x + 1, x^2)$. Третья компонента вектора A имеет наибольшую степень (она была выбрана потому, что имеет наименьшее число слагаемых среди многочленов с наибольшей степенью). Поэтому после первой итерации алгоритма приведения матрицы A к треугольному виду получаем $A = (x, 1+x)$, $B = (x^2)$. Теперь в матрице A выберем первую строку и проделаем вторую итерацию этого алгоритма. Тогда получим $A = (1)$, $B = (x^2, x)$. После третьей итерации алгоритма приведения к треугольному виду матрица A не содержит элементов, а для матрицы B имеем: $B = (x^2, x, 1)$. Таким образом, после окончания работы рассматриваемого алгоритма вектор B содержит три компоненты, т.е. исходный многочлен неприводим.

Приведенный выше алгоритм Берлекемпа замечателен тем, что его можно расширить до алгоритма факторизации (разложения на множители) многочленов над конечными полями. Однако он не является самым быстрым алгоритмом для тестирования неприводимости. Далее мы опишем более быстрые алгоритмы. Для оценки их сложности понадобится оценить сложность алгоритма Евклида.

2.1.3 Оценка сложности алгоритма Евклида

Согласно школьному алгоритму деления операция приведения многочлена степени m по модулю k-члена степени n имеет сложность $O(mk)$. Используя этот факт, легко получить оценку сложности алгоритма Евклида вычисления НОД многочленов, степени которых не превосходят n. Несколько лучшая оценка получается при применении версии алгоритма Евклида, в которой вместо полного деления с остатком применяется вычитание делимого, умноженного на соответствующий одночлен. Эта оценка в наихудшем случае равна $O(n^2)$. В среднем этот алгоритм работает существенно быстрее.

Можно использовать теоретически более быстрый вариант Шенхаге-Моенка алгоритма Евклида ([146], [139]), который дает в наихудшем случае оценку $O(M(n)\log n)$, где $M(n)$ — сложность умножения многочленов степени n в рассматриваемом поле. Изложение этого алгоритма в [5] некорректно; корректное изложение, восходящее к Штрассену, имеется в [101]. Аккуратная оценка, полученная в [101], имеет вид $24M(n)\log_2 n + O(n)$. Для чисел n специального вида она понижается до $12M(n)\log_2 n + O(n)$.

Отметим (см., например,[101] или [25]), что сложность вычисления остатка от деления многочлена степени $2n$ на многочлен степени n равна $5M(n) + O(n)$, а Шенхаге показал [147] (см. также [101]), что для любого поля $GF(q)$ справедливо равенство $M(n) = 64n\log n\log\log n + O(n)$. Все три мультипликативные константы в указанных оценках довольно велики, и в итоговой оценке они перемножаются, поэтому реально указанная оценка сложности быстрого алгоритма Евклида $O(n\log^2 n\log\log n)$ становится меньше квадратичной оценки стандартного алгоритма Евклида, вероятно, начиная с многих тысяч. Возможно, использование вместо алгоритма Шенхаге метода Карацубы для умножения сделает быстрый алгоритм Евклида более практичным.

2.1.4 Тестирование на неприводимость многочленов над конечными полями

Для проверки неприводимости полинома q заданной степени n по заданному простому модулю используется следующий известный алгоритм ([45]).

Проверяем многочлен на наличие кратных корней. Для этого вычисляем его производную и находим наибольший общий делитель многочлена и его производной. Если он не равен 1, то очевидно, что q приводим.

Впрочем, видимо эффективность этого шага для быстрой отбраковки приводимых многочленов невелика, а алгоритм Евклида работает достаточно медленно, поэтому этот шаг алгоритма проще всего опустить.

Далее выполняем следующий тест, пригодный не только для поля $GF(p)$, но и для произвольного конечного поля, но который быстрее всего работает при $p = 2$. Строим последовательность полиномов $q_{k+1} = q_k^p \bmod q$ начиная с полинома $q_0 = x$. Очевидно, что $q_k = x^{p^k} \bmod q$. Полином q будет неприводимым тогда и только тогда, когда $q_n = x \bmod q$ и для любого простого делителя s числа n наибольший общий делитель многочленов $q_{n/s} - x$ и q будет равен 1. Очевидно, что если для некоторого s $q_{n/s} = x$, то многочлен q приводим, и вычислять последовательность q_k далее n/s не нужно.

Для оценки сложности алгоритма заметим, что возведение в степень p проводится по формуле $f(x)^p = f(x^p) \bmod q$ и имеет сложность $O(pkn)$, где k — число одночленов в многочлене q, то есть при тестировании «малочленов» эта сложность линейна, а в общем случае не более чем квадратична. Поэтому сложность вычисления последовательности $q_k, k = 0, \ldots, n$ в случае «малочленов» квадратична, а в общем случае не более чем кубична.

Более точная оценка сложности тестирования неприводимости для произвольного многочлена получена в [101]. Она имеет вид

$$O(M(n)\log_2 p + (n^{(\omega+1)/2} + n^{1/2}M(n))d(n)\log_2 n),$$

где $d(n)$ есть число различных простых делителей числа n, $M(n)$ обозначает сложность умножения многочленов степени n, а ω есть так называемая экспонента матричного умножения, т.е. наименьшее число, для которого существует алгоритм умножения двух матриц размера $n \times n$ сложности $O(n^\omega)$. Далее будет доказана несколько более точная оценка. Как известно (см., например [55]), $d(n) = O((\ln n)/\ln\ln n)$, а почти всегда $d(n)$ асимптотически равно $\ln\ln n$.

2.1.5 О быстрой линейной алгебре

Наилучшая известная оценка $\omega < 2.376$ принадлежит Винограду и Куппершмиту (см., напр.[101]), откуда следует при малых p оценка сложности тестирования неприводимости $O\ (n^{1.688})$, где символ $O\ (n)$ означает $O(n\log^{O(1)} n)$. В [101] отмечается, что используя результат В.Я.Пана и Хуана о быстром умножении прямоугольных матриц, показатель степени в предыдущей оценке можно уменьшить до 1.667.

Заметим однако, что мультипликативная константа в оценке сложности умножения матриц (и, как следствие, в приведенной выше оценке), видимо, очень велика и о практическом применении их алгоритма (а также предшествующих подобных алгоритмов) пока ничего не слышно. Реальную возможность применения имеет только исторически первый такой алгоритм, открытый в 1967 году Ф.Штрассеном [73]. Сложность этого алгоритма оценивается как $42n^{\log_2 7}$ операций в поле $GF(p)$. Однако при небольших значениях n (порядка тысячи при $p = 2$) видимо более быстро будет работать так называемый алгоритм «четырех русских»— так называют на Западе алгоритм для умножения булевских матриц, предложенный в [3] для решения задачи о вычислении транзитивного замыкания графа (см. [5]). Модификация этого алгоритма позволяет перемножать $n \times n$ матрицы над полем $GF(p)$ со сложностью $n^3/\log_p n$ как в программной имплементации, так и при реализации схемами над данным полем. Преимущество этого алгоритма также в том, что он удобен для использования векторных операций (когда вместо битов и битовых операций используются машинные слова в 32 бита, то время его работы уменьшается примерно в 32 раза), а также в относительной простоте программирования, так как он не содержит рекурсии. При больших значениях n (порядка нескольких десятков тысяч) возможно выгодным будет комбинирование этого алгоритма с несколькими шагами штрассеновской рекурсии (а при использовании двух-трех шагов рекурсии можно в программировании вообще избежать рекурсивных вызовов программы).

2.1.6 Алгоритм «четырех русских» для умножения матриц над конечным полем

Далее опишем схемную реализацию, которая основана на идее О.Б.Лупанова из статьи о вентильных схемах [46], опубликованной намного раньше работы [3]. Задача умножения двух $n \times n$ матриц сводится к n-кратному выполнению умножения $n \times n$ матрицы A на n-мерный вектор X. Компонента произведения с номером i равна

$$\sum_{j=1}^{n} a_{i,j}x_j.$$

Пусть $r < n$ некоторый параметр, который выберем ниже. Разобьем переменные x_i на $\lceil n/r\rceil$ групп по r переменных $x_{kr+1},\ldots,x_{(k+1)r}$ (в последней группе их может быть меньше). Реализуем заранее все линейные функции от каждой из этих групп переменных. Все функции от переменных $x_{kr+1},\ldots,x_{(k+1)r}$ имеют вид $c_1x_{kr+1}+\ldots+c_kx_{(k+1)r}$ и их количество равно p^r. Общее число всех функций оценивается как $(1+n/r)p^r$. Сложность реализации каждой функции оценивается как $2r-1$ (r умножений в поле $GF(p)$ и $r-1$ сложений). Сложность реализации системы всех функций оценивается поэтому как $(2r-1)(1+n/r)p^r = (2n+2r-1-n/r)p^r$, а мультипликативная сложность — как $r(1+n/r)p^r = (n+r)p^r$.

В случае схемной реализации конечно надо предполагать, что одна из матриц известна заранее. В случае программной имплементации это предположение излишне. В этом случае для вычисления значений этих функций по их коэффициентам и переменным заранее вычисляются и загружаются в память $\lceil n/r \rceil$ таблиц — двумерных массивов размера $p^r \times r$, из которых значение функции извлекается за одну операцию. После этого для вычисления каждой из n компонент произведения достаточно сделать по формуле

$$\sum_{j=1}^{n} a_{i,j}x_j = \sum_{i=1}^{r} a_{i,j}x_j + \ldots + \sum_{j=r(\lceil n/r \rceil -1)+1}^{n} a_{i,j}x_j$$

$\lceil n/r \rceil - 1 \le n/r$ сложений в поле $GF(p)$. Общая оценка сложности поэтому имеет вид

$$n^2/r + (2n + 2r - 1 - n/r)p^r,$$

а мультипликативная сложность оценивается как $(n + r)p^r$. Учитывая, что сложение в поле $GF(p)$ требует $O(\log_2 p)$ операций, а умножение — $M(GF(p)) = O(\log_2 p)^2$ операций, получаем оценку битовой сложности (и сложности реализации логическими схемами)

$$M(GF(p))(n + r)p^r + O(n^2/r + (n + r - 1 - n/r)p^r)(\log_2 p) =$$

$$O(\log_2 p)^2(n + r)p^r + O(n^2/r + (n + r - 1 - n/r)p^r)(\log_2 p).$$

Выбираем теперь параметр r так, чтобы главным членом был n^2/r. Для этого достаточно, чтобы $p^r \log_2^2 p = o(n/r)$. Можно выбрать $r = \lfloor \log_p n - 3 \log_p \log_p n \rfloor$, тогда

$$p^r \log_2^2 p \le (1 + o(1)) \log_2^2 p n/r^3 = O(n/r^2)$$

при $\log_p n \ge \log_2^2 p$. Значит оценка битовой сложности умножения матрицы на вектор имеет вид $O(n^2/r) = O((\log_2 p)^2 n^2/\log_2 n)$, а оценка сложности реализации схемами над полем $GF(p)$ асимптотически равна $n^2/\log_p n$. Для умножения матриц соответствующие оценки умножаются на n и имеют вид $O(\log_2 p)^2 n^3/\log_2 n)$ и $n^3/\log_p n$. В частности, при $p = 2$ получается оценка сложности

$$n^3/(2\log_2 n) + O(n^3 \log_2 \log_2 n/\log_2^2 n)$$

и глубины $\log_2 r + \log_2 n/r + O(1) = \log_2 n + O(1)$, так как глубина каждой из заранее вычисленных линейных функций равна $\log_2 r + O(1)$, а глубина каждой из внешних сумм равна $\log_2 n/r + O(1)$.

В случае программной имплементации эта оценка заменяется соответствующей оценкой по порядку, так как там надо учитывать еще сложность операций извлечения из памяти. Заметим, что объем памяти, необходимый для умножения матриц, асимптотически равен $np^r(\log_2 p)^2 = O((\log_2 p)^2 n^2/\log_p^C n)$ бит при выборе $r = \lfloor \log_p n - C \log_p \log_p n \rfloor$ при сохранении указанного выше порядка сложности, так как при умножении матрицы на новый вектор можно использовать память, освободившуюся после выполнения предыдущего умножения.

В реальной компьютерной имплементации объем памяти конечно ограничен, и поэтому для сложности умножения асимптотически получается лишь оценка $O(n^3)$, но с маленьким мультипликативным множителем (для его минимизации при выборе r мы ограничены лишь объемом используемой памяти, поэтому при малых n можно выбирать r и больше $\log_p n$.). Скорость извлечения значения функции из загруженных в память таблиц зависит от их размера (все же лучше, чтобы он был поменьше) и от числа p. Если оно не степень двойки, то преобразование p-ичных векторов в двоичные адреса элементов массива делается не так быстро и может сильно замедлить работу алгоритма. Нас однако в основном интересует случай $p = 2$, тогда это преобразование делается быстрее. В случае программной имплементации можно добиться дополнительного ускорения, представляя умножение матриц как

умножение матрицы на вектор, элементами которого тоже являются векторы. Тогда вместо предварительного вычисления системы линейных функций $c_1x_{kr+1}+\ldots+c_rx_{(k+1)r}$ от групп переменных из поля $GF(2)$ $x_{kr+1},\ldots,x_{(k+1)r}$ в количестве не более $(1+n/r)2^r$ штук, можно вычислять соответствующие суммы двоичных векторов длины n. Если длина машинного слова равна w (например $w=32$), то каждый из этих векторов можно представить в виде вектора из $\lceil n/w\rceil$ машинных слов, и заранее вычислять и запоминать $\lceil n/w\rceil$ раз все суммы вида $c_1x_{kr+1}+\ldots+c_rx_{(k+1)r}$, где роль переменных играют машинные слова. Так как умножение машинного слова на бит и побитовый XOR для машинных слов выполняются отдельными машинными операциями (и более быстрыми, чем сложение чисел), то сложность вычисления системы всех этих сумм оценивается величиной $\lceil n/w\rceil(1+n/r)2^r$, а объем памяти для их хранения оценивается в битах как $O(w(n/r)2^r)$, потому что вычисления столбцов произведения матриц делаются последовательно и память после каждого из них освобождается. После этого вычисление произведения матриц требует не более $\lceil n/w\rceil n^2/r$ операций побитового XOR машинных слов и $\lceil n/w\rceil n\lceil n/r\rceil$ операций извлечения машинных слов из памяти. При выборе $r=\lceil\log_2 n-\log_2\log_2 n\rceil$ оценка сложности умножения матриц асимптотически имеет вид $2n^3/(w\log_2 n)$. Если же выбрать $r=\lceil\log_2 n-C\log_2\log_2 n\rceil$, то при той же асимптотике сложности памяти будет достаточно аимптотически $wn^2/\log_2^{C+1} n$.

2.1.7 Быстрый алгоритм для решения систем линейных уравнений над конечным полем

Решение системы линейных уравнений, как известно, сводится к обращению матрицы и умножению полученной обратной матрицы на вектор. Как показано в [5], обращение матрицы можно выполнить по порядку с той же сложностью, что и умножение матриц. Поэтому алгоритм предыдущего раздела порождает алгоритм решения системы линейных уравнений над полем $GF(p)$ сложности $O(n^3/\log_p n)$. Однако метод [5] довольно сложный и мультипликативная константа в оценке будет достаточно велика. Но еще до появления работ [5], [3], [146] в работе [43] был получен алгоритм сложности асимптотически $n^3/\log_p n$ для решения систем линейных уравнений. Он представляет из себя модификацию алгоритма Гаусса и осуществляет приведение расширенной матрицы системы к верхнетреугольному виду с ненулевыми диагональными элементами. Кратко опишем его.

Выберем $r=\lfloor\log_p n-3\log_p\log_p n\rfloor$ и назовем первые r элементов любой строки матрицы ее *кортежем*; очевидно, что число различных кортежей не больше $p^r=O(n/r^3)=O(n/\log_p^3 n)$. Вычитая одну строку с данным кортежем из всех остальных строк с таким же кортежем, получаем матрицу, у которой будет не более p^r-1 строк с ненулевыми кортежами. Переставляя строки, можно переместить все строки с ненулевыми кортежами в верхнюю часть матрицы. Сложность проведенных вычислений не больше $n^2+O(nr)+O(n^2/\log_p^2 n)$, если учесть, что сложность перестановки двух строк матрицы равна $O(n)$, а сложность сравнения кортежей равна $O(r)$. Применим теперь к верхней части матрицы алгоритм Гаусса и преобразуем ее в матрицу, у которой только r строк имеют ненулевые кортежи, образующие верхнетреугольную $r\times r$-матрицу, расположенную в левом верхнем углу исходной $n\times n$ матрицы. Для этого придется выполнить $O(p^r r)$ раз элементарные преобразования над строками матрицы и общая их сложность оценивается как $O(p^r rn)=O(n^2/\log_p^2 n)$. Таким образом, со сложностью $n^2+O(n^2/\log_p^2 n)$ можно получить матрицу, у которой в левом верхнем углу расположена верхнетреугольная $r\times r$-матрица, а в остальной части первых r столбцов стоят нули. После этого остается привести к верхнетреугольному виду $(n-r)\times(n-r)$ подматрицу, лежащую в пересечении остальных столбцов и строк с нулевыми кортежами. Для этого рекурсивно применяем описанный выше алгоритм еще $\lceil(n-r)/r\rceil$ раз (в последний раз значение r может уменьшится). Окончательная оценка сложности имеет вид

$$(n^2+O(n^2/\log_p^2 n))(n/r+1)=n^3/\log_p n+O(n^3/\log_p^3 n).$$

Если учесть, что при выполнении элементарных преобразований битовая сложность операции умножения в поле $GF(p)$ равна $O(\log_2^2 p)$, а операции сложения равна $O(\log_2 p)$, то оценка битовой сложности имеет вид

$$(\log_2 p)n^3/\log_p n + O(n^3/\log_p^2 n)$$

при $\log_p n \geq \log_2^2 p$.

В отличие от предыдущего, указанный алгоритм пригоден только для программной имплементации и не требует дополнительной памяти. Для него не существен вид числа p. При $p = 2$ его можно ускорить точно также, как описано в предыдущем разделе.

Далее нам понадобится оценить сложности модулярной экспоненциации многочленов над конечными полями.

2.1.8 Оценка сложности модулярной экспоненциации многочленов над конечными полями

Заметим, что сложность возведения произвольного $p-$ичного многочлена степени n в степень $m < p^n$ по модулю неприводимого k-члена можно оценить, используя p-ичный вариант метода А.Брауэра для аддитивных цепочек , как

$$O(kpn\log m) + O(\frac{M(n)\log m}{\log\log m})$$

операций в поле $GF(p)$.

Подробнее об этом см. в разделе ??????? добавить ссылку!!!! При этом для модулярного возведения в степень p использовалась оценка сложности $O(kpn)$.

Для произвольного неприводимого модуля f первое слагаемое можно заменить или на $O(M(n)\log_2 p)$, вычисляя p-ю степень методом аддитивных цепочек, или на $O(pM(n))$, быстро возводя в p-ю степень, а потом выполняя приведение по модулю данного f. Мультипликативные константы можно уменьшить, если по модулю f заранее вычислить «обратный» многочлен R, тогда приведение по модулю можно свести к двум обычным умножениям многочленов (см., например, [25].)

Асимптотически почти все операции модулярного умножения являются на самом деле операциями возведения в квадрат и их надо реализовывать, учитывая это обстоятельство, так как и в школьном алгоритме, и в алгоритме Карацубы, и в алгоритмах, основанных на использовании быстрого преобразования Фурье возведение в квадрат имеет меньшую константу в оценке сложности, чем общее умножение (в последнем случае, например, потому что вместо трех преобразований Фурье можно делать два).

В случае малых p второй способ может оказаться экономнее, так как в нем не выполняются операции умножения, а только операции приведения по модулю. При $p = 2$ его оценка понижается до $2M(n) + O(n)$.

При небольших n (порядка нескольких сотен) некоторого ускорения операции приведения по данному модулю можно достичь, рассматривая ее как линейное преобразование, т.е. как умножение $np \times n$-матрицы на n-мерный вектор и реализуя его со сложностью $O(n^2 p/\log_p n$ «алгоритмом четырех русских». Если модуль представляет из себя k-член, то школьный алгоритм дает еще лучшую оценку $O(kpn)$.

Теперь можно перейти к вероятностным алгоритмам.

2.1.9 Вероятностные алгоритмы тестирования на неприводимость многочленов над конечными полями

Такие алгоритмы были предложены Рабиным и Бен-Ором (см., напр. [101]). Бен-Ор заметил, что в среднем у любого многочлена q над полем $GF(p)$ имеется неприводимый делитель f

степени не выше $r = O(log_2 n)$. Так как f делит $x^{p^r} - x$, то $НОД(q, x^{p^r} - x) \neq 1$, значит для рассмотренной выше последовательности $q_m = x^{p^m} \bmod q$ имеем, что $НОД(q_r - x, q) \neq 1$. Поэтому для проверки того, что многочлен q не имеет делителей степени не выше r, достаточно проверить, что $НОД(q_m - x, q) = 1, m \leq r$, в противном случае он такие делители имеет и поэтому приводим. Сложность вычисления последовательности $НОД(q_m - x, q), m \leq r$ оценивается как $r(6 \log_2 pM(n) + E(n))$ в общем случае и как $r(O(pkn) + E(n))$ в случае, если тестируемый многочлен q содержит k одночленов, где $E(n)$ есть сложность вычисления НОД многочленов степени n. В случае $p = 2$ первая оценка понижается до $r(2M(n) + E(n))$.

При $r = \lfloor n/2 \rfloor$ указанный алгоритм является не вероятностным, а детерминированным, и всегда дает правильный ответ. Однако его сложность оказывается выше квадратичной. Если же выбрать $r = C \log_2 n$, то он становится вероятностным алгоритмом и будет давать правильный ответ только с некоторой положительной вероятностью. Однако сомнительными будут только его утверждения о неприводимости данного многочлена. Утверждения о приводимости будут, очевидно, всегда правильными.

Если для вычисления НОД в вероятностном алгоритме Бен-Ора использовать обычный алгоритм Евклида со сложностью $E(n) = O(n^2)$, то его сложность будет все равно выше квадратичной, но если использовать быструю версию алгоритма Евклида с оценкой $E(n) = O(M(n) \log n)$, то сложность указанного вероятностного алгоритма будет равна $O(M(n) \log^2 n + M(n) \log n log_2 p)$ и при малых p она будет меньше сложности детерминированного алгоритма даже в его самом быстром варианте. Причина этого в том, что в алгоритме Бен-Ора вычисляется только небольшое число членов последовательности q_i, и поэтому количество операций модулярного умножения многочленов в нем меньше, чем в детерминированном алгоритме, хотя количество операций вычисления НОД может быть и больше.

Поэтому алгоритм Бен-Ора можно использовать для ускорения работы любого детерминированного алгоритма. Для этого сначала запускаем алгоритм Бен-Ора (может быть с небольшим значением r), который быстро будет отсеивать приводимые многочлены, а после того как он закончил работу, объявив что многочлен вероятно неприводим, запускаем детерминированный алгоритм.

Для ускорения работы в этом предварительном применении алгоритма Бен-Ора можно ограничится $k = \lfloor \log_2 n \rfloor$, т.е. вычислять НОД данного многочлена q с многочленами $f_i = x^{2^i} + x, i \leq \log_2 n$. Тогда на первом шаге алгоритма Евклида выполняется деление на f_i с остатком, а потом алгоритм Евклида работает с многочленами степени не больше 2^i, поэтому оценку сложности этого варианта алгоритма Бен-Ора можно улучшить до

$$\sum_{i=1}^{\lfloor \log_2 n \rfloor} \frac{O(nM(2^i))}{2^i} + E(2^i) =$$

$$= O(n \log^2 n \log\log n + E(n)) = O(n \log^2 n \log\log n) + O(E(n))$$

в случае использования быстрого алгоритма Евклида и быстрого деления и до

$$\sum_{i=1}^{\lfloor \log_2 n \rfloor} \frac{O(nM(2^i))}{2^i} + E(2^i) = O(n^2 + E(n)) = O(n^2) + O(E(n))$$

в случае использования обычного алгоритма Евклида и школьного деления. При малых i вычисление НОД можно заменить на деление на заранее вычисленные неприводимые делители многочленов $x^{2^i - 2} + \ldots + 1$.

2.1.10 Еще один вероятностный алгоритм

был предложен в [17] для случая $p > 2$. В этом алгоритме сначала вычисляется разложение $p^n - 1 = 2^k t, t = 2s + 1$ и для данного многочлена $q(x)$ выбирается случайно многочлен a

меньшей степени и для него вычисляются по модулю q сначала многочлен a^t, и если он не равен 1, то последовательным возведением в квадрат вычисляются многочлены $a^{2^i t}, i < k$, пока не получится минус единица (к сожалению, операция возведения в квадрат в поле характеристики $p > 2$ выполняется не так быстро, как хотелось бы). Если этого не случилось, то многочлен приводим, так как для неприводимого многочлена все выполняемые операции фактически выполняются в определяемом этим многочленом поле порядка p^n, а в этом поле для любого ненулевого a очевидно $a^{p^n-1} = 1$, откуда, извлекая поочередно квадратные корни, всегда находим такое r, что $a^{2^r t} = -1, r < k$, или выясняем, что $a^t = 1$. Если же процедура завершилась удачно, выбирается новый случайный многочлен a, и она повторяется, но не более чем, скажем, 20 раз. В [17] показано, что для приводимого многочлена вероятность удачи в этой процедуре не больше половины. Значит, если процедура удалась 20 раз, то этот многочлен приводим с вероятностью меньше одной миллионной, и можно для точной проверки запускать детерминированный тест. Если число n имеет достаточно много различных простых делителей, то время работы детерминированного больше, чем указанного вероятностного, хотя бы потому, что в нем не применятся алгоритм Евклида. Однако если n простое, то в детерминированном алгоритме алгоритм Евклида тоже не применяется ни разу, а только вычисляется $x^{p^n} \bmod q$, и в этом случае рассматриваемый вероятностный алгоритм бессмысленно применять. Но в сравнении с алгоритмом Бен-Ора алгоритм [17] работает асимптотически медленнее, если в первом из них применять быстрый алгоритм Евклида. В случае же применения его стандартной версии, возможно, в некоторых случаях алгоритм Бен-Ора теряет свое преимущество особенно, если для вычисления $a^t \bmod q$ применять не метод аддитивных цепочек, а упоминавшийся выше алгоритм [101]. Кроме того, алгоритм [17] легче программировать. Возможно, определенный эффект может дать совместное применение небольшого числа шагов этого алгоритма вместе с небольшим числом шагов алгоритма Бен-Ора.

2.2 Поиск неприводимых и примитивных многочленов

2.2.1 Генерация неприводимых многочленов над конечными полями

До сих пор неизвестен детерминированный алгоритм, который за полиномиальное от n время генерирует какой-нибудь неприводимый многочлен степени n над полем $GF(p)$. Очень мало известно результатов о существовании неприводимых многочленов с какими-либо ограничениями на коэффициенты (впрочем, известна формула для числа возвратных неприводимых многочленов, см. например [?]).

Однако полиномиальный алгоритм, генерирующий неприводимые многочлены с вероятностью, близкой к единице, указать теперь не трудно. Вероятность выбранного наугад многочлена степени n оказаться неприводимым равна $O(1/n)$. Действительно, как известно, число неприводимых многочленов степени n над полем $GF(p)$ равно

$$I(n,p) = \sum_{d|n} \frac{1}{n} \mu(n/d) p^d,$$

где $\mu(n)$ есть функция Мебиуса, откуда следует, что $1/2n \le I(n,p) \le 1/n$ при $p^n \ge 16$.

Поэтому для генерации неприводимого многочлена степени n достаточно сделать $O(n)$ случайных выборок многочленов степени n и после каждой выборки тестировать результат на неприводимость каким-нибудь из описанных выше полиномиальных по сложности алгоритмов.

Для ускорения составления таблицы неприводимых многочленов можно заметить, что взаимно возвратные друг к другу многочлены, т.е. получающиеся друг из друга преобразованием $p(x) \to p^*(x) = x^n p(x^{-1})$, будут приводимыми или неприводимыми одновременно, значит достаточно перебрать только половину всех многочленов. Как известно, порядки многочленов $p(x)$ и $p^*(x)$ совпадают, поэтому взаимно возвратные многочлены будут примитивными или непримитивными одновременно, поэтому аналогичным образом вдвое сокращается перебор при составлении таблицы примитивных многочленов.

2.2.2 Тестирование примитивности неприводимого многочлена

В некоторых вопросах теории кодирования и криптографии важную роль играют так называемые примитивные многочлены. Напомним одно из эквивалентных друг другу определений примитивного многочлена. Неприводимый над полем $GF(p)$ многочлен степени n называется примитивным, если хотя бы один из его корней в поле $GF(p^n)$ (на самом деле все его корни) является примитивным элементом в этом поле, т.е. степени этого элемента пробегают все ненулевые элементы поля.

Известно, что примитивных элементов в любом поле $GF(q)$ имеется ровно $\phi(q-1)$, где ϕ – функция Эйлера. Деля $\phi(p^n-1)$ на число корней примитивного многочлена (все они – примитивные элементы поля $GF(p^n)$), получаем что число примитивных многочленов степени n равно $\phi(p^n-1)/n$ (то, что это число – целое, вовсе не очевидно, но проведенное рассуждение фактически это доказывает).

Несмотря на наличие формулы для их числа, никаких явных общих конструкций примитивных многочленов неизвестно. Мало известно и о существовании примитивных многочленов с какими-либо ограничениями на коэффициенты. Впрочем, недавно Вен Бао Хан [162] доказал при нечетном p существование при $n \geq 7$ примитивного многочлена с заранее заданными первыми двумя старшими коэффициентами.

Для проверки построенного неприводимого многочлена на примитивность надо проверить, является ли элемент $x \bmod q$ образующим элементом построенного поля $GF(p^n)$. Для этого известен со времен Гаусса следующий алгоритм, который заключается в проверке того, что для любого простого делителя s числа p^n-1 многочлен $x^{(p^n-1)/s} \bmod q$ не равен 1. Этот алгоритм применим конечно и для тестирования на примитивность произвольного элемента конечного поля.

Заметим, что количество простых делителей числа n не превосходит, как отмечалось выше, по порядку $(\log_2 p)n/\log n$, а в среднем, вероятно, гораздо меньше, скорее всего, $O(\ln n + \ln\ln p)$. Во всяком случае, при $n=1$ в среднем согласно [55] число простых делителей у $p-1$ асимптотически равно $\ln\ln p$.

2.2.3 Факторизация чисел вида $p^n - 1$

В определенном смысле самая трудная часть тестирования это разложение на простые множители числа p^n-1. Однако ее достаточно сделать для данных n и p только один раз. Обычно указанные разложения берут из таблиц «Каннингемовского проекта» [89, 160] или используют программы типа «Математика». Однако эти таблицы для нашего читателя (как, возможно, и программы) малодоступны, поэтому в общих чертах опишем, как можно построить простейший алгоритм самостоятельно.

Сначала надо написать простейший алгоритм разложения на множители небольших чисел. Простой тест на простоту можно найти в [17] в параграфе 1.2. Он удобен для программирования тем, что использует только модулярное умножение и бинарный алгоритм возведения в степень. Если он не проходит, то можно применить алгоритм пробного деления для неболь-

ших чисел, а для чуть больших алгоритм Ферма, Шермана-Лемана или один из алгоритмов Полларда, описанных там же или в [39].

Если n составное, то имеет смысл разложить $p^n - 1$ на множители, пользуясь каким-нибудь алгоритмом разложения многочлена $x^n - 1$ на круговые многочлены

$$x^n - 1 = \prod_{d|n} \Phi_d(x),$$

о которых можно прочитать в любом хорошем учебнике алгебры. Самый быстрый алгоритм имеется в [101]. Его сложность $O(M(n)\log_2 n)$ операций над целыми числами, где $M(n)$ есть сложность умножения целокоэффициентных многочленов. При небольших n реальная оценка здесь $M(n) = O(n^{\log_2 3})$ получается методом Карацубы. Сам алгоритм рекурсивен и выглядит следующим образом. Сначала разлагаем на множители число n и находим все его простые делители $p_i, i = 1, \dots s$. Полагаем $f_0 = x - 1$ и в цикле по i от 1 до s выполняем процедуру $f_i = \frac{f_{i-1}(x^{p_i})}{f_{i-1}}$. Тогда $f_i = \Phi_{p_1 \dots p_i}$. Если положить $m = p_1 \dots p_s$, то $\Phi_n = \Phi_m(x^{n/m}) = f_r(x^{n/m})$.

Далее приходится разлагать каждое из чисел $\Phi_d(p)$ на простые множители, например, одним из алгоритмов Полларда. Отметим еще, что для тестирования простоты чисел Мерсенна $2^p - 1$ имеется быстрый полиномиальный алгоритм Лемера-Люка, также изложенный в [17], [39].

Предположим теперь, что факторизация проведена или взята из таблицы и оценим сложность остальной процедуры. Для этого понадобится

2.2.4 Оценка сложности совместного вычисления системы степеней в конечном поле

Для ее получения воспользуемся оценкой сложности совместного вычисления системы из m натуральных чисел величины не большей N аддитивными цепочками ([24],[39] или следующий далее раздел книги). Эта оценка имеет вид

$$\log_2 N + \frac{(1 + o(1))m \log_2 N}{\log_2 log_2 N}.$$

Из нее немедленно следует, что совместное вычисление произвольной системы степеней

$$x^{n_1}, \dots, x^{n_m}, n_i \le N, i = 1, \dots, m$$

в поле $GF(p^n)$ можно выполнить за

$$(3M(n) + O(n))(\log_2 N + \frac{(1 + o(1))m \log_2 N}{\log_2 log_2 N})$$

операций в поле $GF(p^n)$. Доказательство оценки для совместного вычисления системы чисел сводится методом перехода к двойственной цепочке к оценке сложности вычисления вектора векторной аддитивной цепочкой. Так как при построении двойственной цепочки элементы удвоения переходят опять в элементы удвоения, а возведение в квадрат в данном поле имеет сложность $K(n) + 2M(n) + O(n)$, где $K(n)$ есть сложность возведения многочлена в квадрат, то указанную оценку можно уточнить как

$$(K(n) + 2M(n) + O(n)) \log_2 N + (3M(n) + O(n)) \frac{(1 + o(1))m \log_2 N}{\log_2 log_2 N},$$

а в случае $p = 2$ как

$$(2M(n) + O(n)) \log_2 N + (3M(n) + O(n)) \frac{(1 + o(1))m \log_2 N}{\log_2 log_2 N},$$

Если представление поля задает неприводимый k-член, то предыдущая оценка усиливается до

$$O(kn)\log_2 N + (3M(n) + O(n))\frac{(1+o(1))m\log_2 N}{\log_2 log_2 N}.$$

2.2.5 Оценка сложности теста на примитивность в конечном поле малой характеристики

Применяя предыдущую оценку к возникающей при тестировании элемента f поля $GF(p^n)$ на примитивность задаче вычисления системы степеней $f^{(p^n-1)/p_i}$, где $p_i, i = 1, \ldots d$ есть система всех различных простых делителей числа $p^n - 1$, получаем следующую оценку сложности вычисления упомянутой системы

$$(\log_2 p)(K(n) + 2M(n) + O(n))n + O(\frac{dM(n)n\log_2 p}{\log n}).$$

Величину d в худшем случае можно оценить как $n/\log n$, в среднем она, вероятно, $O(logn)$. Значит, рассматриваемая часть алгоритма имеет в худшем случае сложность $O(n^3)\log_2 p$, а в среднем, вероятно, $O\ (n^2)\log_2 p$, причем при малом числе k членов в данном неприводимом полиноме в среднем, вероятно, $O(n^2k)\log_2 p$.

2.2.6 Генерация примитивных многочленов над конечными полями

Для этой цели до сих пор неизвестен полиномиальный детерминированный алгоритм. Изложенный выше алгоритм тестирования примитивности является полиномиальным в предположении что $p^n - 1$ уже факторизовано. Но вероятностный полиномиальный алгоритм генерации примитивных многочленов в этом предположении указать теперь не трудно. Вероятность выбранного наугад многочлена степени n оказаться примитивным равна $O(1/n\log(n\log p)) = O\ (1/n)$. Действительно, число примитивных многочленов степени n над полем $GF(p)$ равно $\phi(p^n - 1)/n$, где $\phi(n)$ есть функция Эйлера, откуда доля примитивных среди всех p^n многочленов степени n равна

$$\frac{\phi(p^n-1)}{p^n n} = \frac{O(1)}{n\log_2(n\log_2 p)},$$

так как согласно [55] $n/\phi(n) = O(\log_2\log_2 n)$.

Поэтому для генерации с вероятностью $1 - \epsilon$ примитивного многочлена степени n достаточно сделать

$$O(n\log(n\log p))\log 1/epsilon = O\ (n)\log 1/epsilon$$

случайных выборок многочленов степени n и после каждой выборки протестировать результат сначала на неприводимость, а потом на примитивность. В худшем случае сложность этой генерации равна

$$O(n^4)\log(n\log p)\log_2 p\log 1\epsilon.$$

При построении таблиц примитивных многочленов [168, 169] случайные выборки не используют, а просто перебирают многочлены в порядке увеличения числа коэффициентов, благодаря чему в таблицу попадают многочлены с наименьшим возможным числом ненулевых коэффициентов; из таблиц [168, 169] видно, что в случае $p = 2$ всегда существуют примитивные пятичлены и семичлены, но, кажется, теоретически это никто еще не доказал.

Сказанное выше о многочленах справедливо, конечно, и для генерации примитивных элементов в данном поле, за исключением того, что в этом случае вместо простейшего в каком-то смысле примитивного элемента обычно достаточно найти хоть какой-нибудь такой

элемент. Для этой задачи со времен Гаусса известен еще один алгоритм, могущий составить конкуренцию изложенному выше алгоритму.

2.2.7 Алгоритм генерации примитивных элементов в поле $GF(p^n)$

Этот алгоритм основан на следующей идее, лежащей в основе одного из доказательств существования примитивного элемента.

Допустим, что известно разложение на простые множители

$$p^n - 1 = \prod_{i=1}^{d} p_i^{\alpha_i}.$$

Для каждого i найдем элемент g_i порядка $q_i = p_i^{\alpha_i}$, т.е. такой, что $g_i^{q_i} = 1$, $g_i^{q_i/p_1} \neq 1$, тогда, согласно известной лемме, порядок элемента $g_1 \dots g_d$ будет равен $q_1 \dots q_d = p^n - 1$, т.е. этот элемент и будет примитивным. Для нахождения, скажем, элемента g_i берем случайный элемент h и вычисляем $H = h^{(p^n-1)/q_i}$. Если $H^{q_i/p_i} = h^{(p^n-1)/p_i} \neq 1$, то порядок элемента H равен q_i (ведь $H^{q_i} = h^{p^n-1} = 1$) и можно положить $g_i = H$. Так как уравнение $h^{(p^n-1)/p_i} = 1$ имеет не более $(p^n - 1)/p_i$ корней в поле $GF(p^n)$ (на самом деле ровно столько), то вероятность выбрать h так, чтобы $h^{(p^n-1)/p_i} \neq 1$ равна $1 - p_i^{-1}$. Для того, чтобы выбрать g_i с вероятностью $1 - \epsilon/d$, достаточно сделать $\frac{O(\log_2 d/\epsilon)}{\log_2 p_i}$ попыток, после чего примитивный элемент можно будет сгенерировать с вероятностью $1-\epsilon$ по формуле $g_1 \cdots g_d$. Так как сложность возведения в степень m элемента поля $GF(p^n)$ оценивается как $O(kpn \log m) + O(\frac{M(n) \log m}{\log \log m})$ операций в поле $GF(p)$, где k– число членов в неприводимом многочлене, определяющем используемое представление поля, то, учитывая, что на практике всегда можно выбрать $k \leq 5$, эту оценку можно переписать при малых k в виде $\frac{O(M(n)) \log m}{\log \log m}$, значит сложность генерации примитивного элемента с вероятностью $1 - \epsilon$ оценивается при малых k как

$$\frac{O(M(n)n \log_2(d/\epsilon))}{\log n} \left(\sum_{i=1}^{d} log_2^{-1} p_i \right) =$$

$$\frac{O(M(n)nd \log_2(d/\epsilon))}{\log d \log n} = \frac{O(M(n)n^2 \log_2(n/\epsilon))}{\log^3 n}.$$

Работу описанного алгоритма можно с некоторой вероятностью ускорить следующим образом. После того, как выбрали h такой, что $h^{(p^n-1)/p_1} \neq 1$, можно вместо вычисления g_1 по формуле $g_1 = h^{(p^n-1)/q_1}$ продолжить вычисления степеней $h^{(p^n-1)/p_i}, i = 2, 3, \dots$, пока впервые не получим $h^{(p^n-1)/p_j} = 1$. Если этого не произойдет, h и есть искомый примитивный элемент и работа алгоритма на этом заканчивается. Если же найдется r_1 такое, что

$$h^{(p^n-1)/p_{r_1+1}} = 1, h^{(p^n-1)/p_i} \neq 1, i \leq r_1,$$

то полагая $Q_1 = q_1 \dots q_{r_1}, G_1 = h^{(p^n-1)/Q_1}$, имеем

$$G_1^{Q_1} = 1, G_1^{Q_1/p_i} = h^{(p^n-1)/p_i} \neq 1, i \leq r_1,$$

т.е. порядок элемента G_1 равен Q_1. Аналогичным образом, выбирая случайный элемент h и возводя его в степени $(p^n-1)/p_i, i = r_1+1, r_1+2, \dots$, до получения первой единицы (конечно, если она получилась уже на первом шаге, то элемент h заменяется на новый случайный элемент), находим элемент G_2 порядка $Q_2 = q_{r_1+1} \dots q_{r_2}$ и т.д. Найдя последний элемент G_s, имеющий порядок $Q_s = q_{r_{s-1}+1} \dots q_d$ вычисляем примитивный элемент по формуле $G_1 \dots G_s$.

Доказательство примитивности не отличается от доказательства примитивности элемента $g_1 \dots g_d$, так как порядки Q_i попарно взаимно просты.

Если применить для поиска примитивного элемента алгоритм, описанный в предыдущих секциях, то после случайного выбора элемента h тоже нужно вычислять степени $h^{(p^n-1)/p_i}, i = 1, 2, \dots$ и бросать это дело после появления первой единицы, выбирая новый случайный элемент, пока не будет найден элемент, для которого все эти степени отличны от единицы. В новом же варианте алгоритма после выбора очередного случайного элемента вычисления начинаются с той степени, на которой случился провал в предыдущей попытке, и количество операций экспоненцирования может при удаче существенно уменьшится.

Отметим еще, что в худшем случае сложность описанного в предыдущих секциях алгоритма при малых p оценивается как

$$\frac{O(M(n))nd \log d \log 1/\epsilon}{\log n},$$

так как вероятность удачи в каждом испытании равна в худшем случае

$$\phi(p^n - 1)/(p^n - 1) = \prod_{i=1}^{d}(1 - p_i^{-1}) = O(1/\log d).$$

Полученная оценка сложности чуть выше, чем сложность алгоритма этой секции.

После нахождения одного примитивного элемента g можно найти примитивные элементы $g^{p^i}, i = 1, \dots, n-1$ последовательным возведением в степень p со сложностью $O(pkn^2)$. При малых p, k это квадратичная сложность. В случае большого p или k близкого к n этот алгоритм в [101] заменен на более быстрый, однако на практике этот случай почти не встречается. Перемножая над полем $GF(p^n)$ линейные двучлены $x - g^{p^k}, k = 0, \dots, n-1$ методом «деления пополам», получаем примитивный многочлен $f(x)$ над полем $GF(p)$, выполнив $O(M(n)) \log n$ операций умножения в поле $GF(p^n)$. Для умножения в этом поле в свою очередь нужно $O(M(n))$ операций в поле $GF(p)$. Окончательно, для генерации примитивного многочлена над полем $GF(p)$ дополнительно нужно $O(M(n)^2) \log n$ операций. Используя оценку Шенхаге для сложности умножения многочленов, получаем окончательную оценку $O\ (n^2)$, а используя при небольших n оценку Карацубы, имеем окончательную оценку в виде $O(n^{3.2})$. В любом из случаев последний алгоритм генерации примитивного многочлена кажется более быстрым, чем приведенный в предыдущих секциях, но он будет порождать не простейший, а какой-то случайный примитивный многочлен.

2.2.8 Об ускорении тестирования неприводимости малочленов над конечными полями

Для простоты рассмотрим наиболее практически важный случай поля $GF(2)$.

Как уже отмечалось, для ускорения поиска неприводимых трехчленов $x^n + x^k + 1$ можно заметить, что трехчлены $x^n + x^k + 1$ и $x^n + x^{n-k} + 1$ взаимно возвратны друг к другу, значит достаточно перебрать только трехчлены $x^n + x^k + 1, k < n/2$, трехчлен $x^n + x^{n/2} + 1$ очевидно является квадратом, то есть приводим. Аналогичным образом можно сократить вдвое перебор и в случае пятичленов.

Далее, с помощью теоремы Штикельбергера-Суона (см., например [8]), проверяем, не является ли выбранный трехчлен приводимым. Она утверждает, что трехчлен $x^n + x^k + 1$ над полем $GF(2)$ заведомо будет приводимым (и более того, будет иметь четное число неприводимых делителей)

1) при четном n, нечетном @?????????$k, k \neq n/2,$, и $nk/2 = 0, 1 \bmod 4$;
2) при четном k и $n = \pm 1 \bmod 8$ в случае, когда k делит $2n$;
3) при четном k и $n = \pm 3 \bmod 8$ если k не делит $2n$.

Кроме того, очевидно, что при четных n и k трехчлен является квадратом, и поэтому приводим.

Указанная проверка результативна с вероятностью асимптотически $1/2$

Уже отмечалось, что для ускорения работы алгоритма генерации неприводимых многочленов можно для просеивания приводимых многочленов применять алгоритм Бен-Ора, а точнее, вычислять НОД данного многочлена q с многочленами $x^{2^i}+x, i=1,2,\ldots$. При $i<\log_2 n$ фактически надо выполнять пробное деление на неприводимые делители многочленов $x^{2^i-2}+\ldots+1$.

Например, вначале выполняем деление на трехчлен $1+x+x^2$. Быстрее всего для этого поделить с остатком на $1+x^3$, для чего достаточно просто заменить все показатели степени в тестируемом многочлене на остатки по модулю 3, и выполнить приведение подобных членов по модулю два, в результате чего и получается искомый остаток по модулю $1+x^3$. Если он будет равен трехчлену $1+x+x^2$, то тестируемый многочлен делится на него, и, значит, приводим. Фактически никакого деления производить не надо, а достаточно проверить, что делится ли $n+k$ Вероятность этого в случае трехчленов равна асимптотически $2/9$, а в случае пятичленов — $20/81$. Подобный же прием в случае пятичленов и деления на $1+x+x^2+x^3+x^4$ возможен, но не эффективен.

Эффективно, однако, пробное деление на $1+x^7$, остаток в котором находится аналогичным способом (с заменой 3 на 7). После этого полученный остаток проверяется на делимость на $1+x+x^3$ и $1+x^2+x^3$ (это нетривиальные неприводимые делители $1+x^7$), для чего достаточно его сравнить со всеми кратными им трехчленами не выше 6-й степени, которые суть

$$1+x^2+x^6, 1+x^4+x^6, 1+x^2+x^3, 1+x+x^3, 1+x+x^5, 1+x^4+x^5,$$

и в случае совпадения с одним из них тестируемый многочлен является приводимым. По существу здесь вычислялся НОД тестируемого многочлена и многочлена $1+x^7$. Вероятность того, что указанный НОД будет отличен от 1, в случае трехчленов равна $6/49$, а полная вероятность результативности одного из двух описанных пробных делений и проверки по теореме Суона равна согласно формуле включения и исключения и китайской теореме об остатках $1/2+2/9+6/49-12/441-1/9-3/49+2/147=83/126=0,658...$ В случае пятичленов указанная вероятность меньше и равна приблизительно 0.347...

В [166] предлагается продолжить эту процедуру и фактически вычислять НОД тестируемого многочлена с двучленами вида $1+x^{2^m-1}, m=4,5,\ldots,10$. На самом деле, конечно, вначале надо выполнить деление с остатком на $1+x^{2^m-1}$ аналогично тому, как это делалось выше, а потом полученный остаток (трехчлен или пятичлен степени не выше 2^m-1 искать в заранее заготовленном и помещенном в память машины списке трехчленов и пятичленов указанной степени, кратных одному из неприводимых многочленов степени, делящей m (произведение всех таких многочленов как известно и равно $1+x^{2^m-1}$).

Логарифмический поиск в этом списке требует времени $O(m)$ и памяти $O(2^{4m})$ в случае пятичленов и $O(2^{2m})$ в случае трехчленов (что ограничивает m числом 5 в первом и числом 10 во втором случае). Использование алгоритма Евклида или пробного деления на заранее вычисленные неприводимые многочлены степеней, делящих m, ликвидирует проблемы с памятью, но требует в обоих случаях времени в худшем случае $O(2^{2m})$ на один многочлен.

Для составления указанного списка можно использовать любой из этих подходов, но оценка времени будет умножаться на число проверяемых трехчленов $O(2^{2m})$ или на число проверяемых пятичленов $O(2^{4m})$, что приводит к границе $m\le 8$ в первом и $m\le 6$ во втором случаях. Граница может быть повышена до 10 если тестируются только многочлены специального вида, например $1+x+x^n$, о чем пойдет речь далее.

Отметим, что указанные приемы сокращают лишь время составления списка неприводимых многочленов с заданными границами степеней, а для тестирования индивидуальных многочленов ускорение достигается лишь с некоторой вероятностью.

Теоретически оценить вероятность взаимной простоты трехчлена степени n с многочленом $x^{2^k}+x$ в общем случае затруднительно. В недавней работе [87] на основе вычислительного эксперимента выдвинута гипотеза, что эта вероятность близка к $1/k$. Также там была выдвинута подтвержденная численными экспериментами гипотеза, что время $T(k)$ проверки условия $(f, x^{2^k}+x)=1$ удовлетворяет равенству

$$T(k)=\sum_{i=1}^{k-1} T(i).$$

Если обозначить $t(n)$ оценку времени работы детерминированного теста для случайного трехчлена степени n, то в [87] стратегия выбора параметра k в процедуре предварительного просеивания основывалась на проверке неравенства $kT(k) < t(n)$, которое должно обеспечить малое время предварительного просеивания по сравнению с временем выполнения детерминированного теста в случае неудачи предварительного просеивания.

В [87] велся поиск неприводимых трехчленов простых степеней n, для которых 2^n-1 тоже простое (называемое числом Мерсенна). Интерес к ним вызван тем, что такие трехчлены автоматически оказываются примитивными. Существуют они не для всех экспонент Мерсенна. В [87] установлен мировой рекорд: найдены два примитивных многочлена (естественно, трехчлены) степени 3021377. Значения k у них равны 361604 и 1010202.

Интересно, что в [87] не использовался и даже не упоминался упоминавшийся выше теоретически самый быстрый известный алгоритм сложности $O\ (n^{1.688})$. Вместо него использовался стандартный описанный выше квадратичный алгоритм с некоторыми программными ухищрениями.

Так, кроме стандартного использования заранее вычисленной таблицы для быстрого возведения в квадрат, и ускорения деления на трехчлен за счет работы не с битами, а с 32-битными машинными словами, в [87] рассматривались только трехчлены с четным $n-k$ (в противном случае можно перейти к взаимному трехчлену, так как n нечетно) и ловко использовалось то обстоятельство, что в квадрате данного многочлена все коэффициенты при нечетных степенях нулевые. Благодаря этому, удалось уменьшить в четыре раза число битовых операций XOR, на четверть уменьшилась используемая память и почти в два раза уменьшилось число обращений к ней.

В [87] интересовались только многочленами с простыми степенями. Для многочленов с составными степенями можно добиться некоторого ускорения следующим способом.

2.2.9 Еще об ускорении тестирования неприводимости <малочленов> над полем из двух элементов

Для трехчленов x^n+x^k+1 над полем $GF(2)$ в случае $(n,k)=t>1$, где t нечетно, проверку на неприводимость можно ускорить следующим образом (если t четно, то все такие многочлены очевидно являются квадратами). Заметим, что тогда $p(x)=f(x^t)$, где $f(x)$ - трехчлен степени $m=n/t$, и если он приводим, то и $p(x)$ тоже приводим, поэтому достаточно провести проверку на неприводимость у всех трехчленов степени m, что делается существенно быстрее, чем при прямой проверке.

Если выяснится, что многочлен $f(x)$ неприводим, то неочевидно, что тогда многочлен $p(x)=f(x^t)$ тоже неприводим.

Имеется, однако, теорема(теорема 3.35 [45]) о том, что если порядок e многочлена $f(x)$ таков, что все простые делители числа t делят e, но не делят $\frac{2^m-1}{e}$, то многочлен $p(x)=f(x^t)$ будет неприводимым порядка et (порядок многочлена $g(x)$ равен r — это, по определению означает, что r есть наименьший показатель степени такой, что x^r-1 делится на $g(x)$). Для проверки указанного условия для e и t разложим t на простые множители,

$$t=p_1^{\alpha_1}\dots p_s^{\alpha_s},$$

что делается со сложностью @????????$o(t)$, определим для каждого p_i максимальный показатель степени β_i, такой что $p_i^{\beta_i}$ делит 2^m-1, что делается со сложностью $O(m^2)$, тогда указанное выше условие равносильно тому, что e должно делится на $p_1^{\beta_1}\dots p_s^{\beta_s}$, а так как e это порядок многочлена $f(x)$, и он не должен быть делителем чисел

$$a_i=\frac{2^m-1}{p_i},$$

значит $x^{a_i} \bmod f(x)$ должно быть отлично от единицы при всех $i=1,\dots,s$, и это условие равносильно указанному выше, а его проверка, как вытекает из оценки соответствующей секции, выполняется со сложностью $O(sm^2\log\log m)$, после чего остается (в случае необходимости) со сложностью $O(m^2)$ убедится в неприводимости многочлена $f(x)$. Заметим, что s не превосходит числа простых делителей у t, то есть не больше $O(\log t/\log\log t)$ (см. [55]), а в среднем $O(\log\log t)$ как отмечалось выше. Поэтому $O(sm^2\log\log m)=$ @???????$o(n^2)$ при $t^2\log\log t/\log t >> \log\log m$.

Указанная оценка сложности проверки упомянутого условия слишком груба, так как в ней оценивается число необходимых умножений в поле $GF(2^m)$ как $O(sm/\log m)$ при m возведениях в квадрат, а сложность каждого умножения согласно оценке Шенхаге как

$$O(m\log m\log\log m).$$

В реальности число умножений может быть не столь большим, так как показатели степени не произвольные s чисел, а числа вида

$$\frac{2^m-1}{p_i}.$$

Если, например, m_i — такое наименьшее число, что $2^{m_i}-1$ кратно p_i, и m_i много меньше m, то, как известно, m кратно m_i и

$$\frac{2^m-1}{p_i}=b_i\frac{2^m-1}{2^{m_i}-1},$$

где

$$b_i=\frac{2^{m_i}-1}{p_i}.$$

Представляя число

$$\frac{2^m-1}{2^{m_i}-1}$$

в виде

$$\frac{a^{m/m_i}-1}{a-1},$$

где $a=2^{m_i}$, можно, предварительно вычислив все степени с маленькими показателями b_i, найти все нужные нам степени, используя только

$$\sum_{i=1}^{s}O(\log m/m_i)$$

умножений, а последняя величина в худшем случае не больше $O(m)$, а в среднем значительно меньше, но число возведений в квадрат при таком методе будет грубо оцениваться как $O(sm)$.

Например, при $n=1000$ возможные значения t есть только 5 и 25, и для упомянутой проверки достаточно выполнить возведения в $(2^{200}-1)/5$ и $(2^{40}-1)/25$ степени. Для этого вычислим $(2^4-1)/5=3$ степень, и

$$(2^{20}-1)/25=3(1+3(1+2^4+1+2^8+2^4+1+(2^4+1)(2^8+1)))=$$

$$= 3(1 + 3(4 + 2^4 + 2^5 + 2^9 + 2^{12})),$$

что требует всего 8 умножений, а потом выполним возведения в $(2^{200} - 1)/(2^4 - 1)5$ и в $2^{20} + 1$ степени. Последняя операция требует лишь одного умножения и 20 кратного возведения в квадрат. Первая операция требует не более 200 возведений в квадрат и не более 12 умножений. Как видим, в реальности общее число операций может быть существенно меньше даже числа n (не считая сложность тестирования многочленов $f(x)$ степени n/t).

Покажем, что если упомянутое выше условие не выполняется, то тестирование проводить не нужно, так как тогда многочлен $f(x^t)$ будет всегда приводимым. Сначала рассмотрим случай, когда $(t, e) = 1$.

Известно [45],теоремы 2.47,3.5, что произведение всех неприводимых над полем $GF(2)$ многочленов степени m и порядка e и единичными старшими коэффициентами равно Q_e — круговому многочлену порядка e, если m — такое наименьшее число, что $2^m - 1$ кратно e (называемое порядком двойки по модулю e), $e > 1$. Круговой многочлен Q_e имеет степень $\phi(e)$, коэффициенты его принадлежат $GF(2)$ и все его корни некратны и лежат в поле $GF(2^m)$, являясь первообразными корнями e-й степени из единицы, то есть имеют вид $\alpha^s, (s, e) = 1, 1 \le s \le e$, где $\alpha \in GF(2^m), \alpha^e = 1, \alpha^r \ne 1, r = 1, \ldots, e - 1$. Отсюда видно, что если $f(x)$ — любой из упомянутых неприводимых многочленов (а их количество, кстати, равно $\phi(e)/m$,), и $\alpha^s, (s, e) = 1$ — любой из его корней, лежащих в поле $GF(2^m)$, то $\alpha^r, (r, e) = 1, rt = s \bmod e$, будет корнем многочлена $f(x^t)$ и одновременно корнем какого-то неприводимого над полем $GF(2)$ многочлена $g(x)$ степени m и порядка e (может быть и равного $f(x)$), значит многочлены $g(x)$ и $f(x^t)$ не взаимно просты, а так как $g(x)$ — неприводим, то он должен быть делителем $f(x^t)$, и последний поэтому не является неприводимым, так как его степень равна $mt > m$.

Рассмотрим теперь случай, когда t имеет простой делитель p, не делящий e. Тогда, согласно предыдущему, многочлен $h(x) = f(x^p)$ приводим над полем $GF(2)$, а значит и многочлен $f(x^t) = h(x^{t/p})$ тоже.

Осталось рассмотреть случай, когда все простые делители t делят e и $(t, (2^m - 1)/e) > 1$. Тогда $t' = t/(t, (2^m - 1)/e) < t$. Если $t' = 1$, то $et = e(t, (2^m - 1)/e)$ делит $2^m - 1$, так как $(t, (2^m - 1)/e)$ делит $(2^m - 1)/e$, значит порядок двойки по модулю et равен $m < mt$, также как и по модулю e. Если же $t' > 1$, пусть p — любой его простой делитель, и $t'' = (t, (2^m - 1)/e)p$. Тогда t'' делит $(t, (2^m - 1)/e)t' = t$. Проверим, что тогда порядок двойки по модулю et'' будет равен $mp < mt''$.

Действительно, указанный порядок s должен быть кратен m и иметь вид mk, так как иначе $2^s - 1$ не делилось бы на e, но

$$2^{mk} - 1 = \frac{2^{mk} - 1}{2^m - 1}(1 + 2^m + \ldots + 2^{(k-1)m})$$

будет делится на $et'' = e(t, (2^m - 1)/e)p$ тогда и только тогда, когда

$$1 + 2^m + \ldots + 2^{(k-1)m}$$

будет делится на p (ведь $e(t, (2^m - 1)/e)$ делит $e(2^m - 1)/e) = 2^m - 1$, но et'' не делит $2^m - 1$, так как тогда бы число $(t, (2^m - 1)/e)p$ делило бы $((2^m - 1)/e)$ и $t = t'(t, (2^m - 1)/e)$ одновременно, что противоречит тому, что $(t, (2^m - 1)/e)$ — наибольший общий делитель чисел t и $(2^m - 1)/e)$, а так как $2^m - 1$ делится на e, а значит и на p, то по модулю p указанная сумма будет равна k, и будет кратна p лишь при k кратном p, откуда и следует требуемое. Заметим, что в обоих случаях было доказано существование такого делителя t' у числа t (может быть $t' = t$), что порядок двойки по модулю et' будет меньше mt'.

Уже отмечалось выше, что произвольный неприводимый над полем $GF(2)$ многочлен $f(x)$ степени m и порядка e является делителем кругового многочлена Q_e, а, значит, многочлен $f(x^{t'})$ делит многочлен $Q_e(x^{t'})$, а так как все простые делители t делят e, то, как показано в доказательстве теоремы 3.35 [45], из свойств круговых многочленов вытекает, что

$Q_e(x^{t'}) = Q_{et'}(x)$, значит многочлен $f(x^{t'})$ делит круговой многочлен $Q_{et'}(x)$, но как показано там же, согласно теореме 2.47 степень каждого неприводимого делителя кругового многочлена $Q_{et'}(x)$ равна порядку двойки по модулю et', то есть в рассматриваемом случае меньше mt', значит, делитель $h(x) = f(x^{t'})$ будет приводимым, так как его степень равна mt', а поэтому и многочлен $f(x^t) = h(x^{t/t'})$ тоже будет приводимым.

Итак, во всех рассматриваемых случаях (кроме самого первого) многочлен $f(x_t)$ оказывается приводимым.

Из указанного выше вытекает также, что при поиске неприводимых многочленов достаточно ограничиться только многочленами с условием $(n, k) = 1$, которых, как известно, $\phi(n)$ штук. Отметим, что при n имеющем много малых простых делителей, доля таких многочленов может стремиться к нулю, но в среднем, как известно, вероятность того, что трехчлен будет удовлетворять указанному условию асимптотически равна $6/\pi^2$. Если целью является поиск одного неприводимого трехчлена, то согласно указанному выше, имеет смысл начать с многочленов с большим нечетным (n, k).

Для пятичленов справедливо почти все сказанное выше, за исключением оценки сложности тестирования всех пятичленов, которая становится трудно выводимой.

2.2.10 О нормальных базисах, соответствующих трехчленам и пятичленам

Пусть $\{\alpha, \alpha^p, \alpha^{p^2}, \ldots, \alpha^{p^{n-1}}\}$ — корни трехчлена $f(x) = x^n + ax^m + b$ в поле $GF(p^n)$. Очевидным необходимым условием базисности системы $\{\alpha, \alpha^p, \alpha^{p^2}, \ldots, \alpha^{p^{n-1}}\}$ является неравенство нулю суммы $\alpha + \alpha^p + \alpha^{p^2} + \ldots \alpha^{p^{n-1}}$, называемой следом элемента α. Так как, по теореме Виета, эта сумма противоположна по знаку коэффициенту при x^{n-1} многочлена $f(x)$, то этот коэффициент должен быть отличен от нуля, то есть $f(x)$ должен иметь вид $x^n + ax^{n-1} + b$. Однако эти многочлены надо проверять на нормальность с помощью указанного ниже алгоритма. Количество проверок равно $O(p^2)$ в случае трехчленов, и $O(n^2p^4)$ в случае пятичленов. Если $p = 2$, нужно проверить только один трехчлен, причем легко видеть, что он не может иметь кратных корней.

В случае поля $GF(2)$ поиск неприводимых трехчленов, порождающих нормальные базисы, нужно вести среди трехчленов вида $x^n + x^{n-1} + 1$. Такие трехчлены будут неприводимыми (или примитивными) одновременно с трехчленами вида $x^n + x + 1$.

Заметим, что приведение по модулю таких многочленов выполняется особенно просто, поэтому если есть такая возможность, именно их можно рекомендовать для имплементации операций в соответствующем поле. Встречаются они вначале довольно часто, а потом все реже и реже. В [166] имеется таблица таких трехчленов до степени 30000. Наша программа нашла еще 2 трехчлена: $1 + x^1 + x^{32767}$ и $1 + x^1 + x^{34353}$. Приведем таблицу таких трехчленов со степенями не превосходящими 1000. В ней также отмечено, какие из них являются примитивными, а также нормальными (т.е. порождающими нормальные базисы, в последнем случае указано число единиц в матрице разложения стандартного полиномиального базиса, соответствующего этому многочлену, по соответствующему нормальному базису).

```
1 + x^1 + x^2          3
1 + x^2 + x^3          5
1 + x^3 + x^4          9
1 + x^5 + x^6          15
1 + x^6 + x^7          23
1 + x^8 + x^9          33
1 + x^14 + x^15        107    primitiv
1 + x^21 + x^22        185    primitiv
1 + x^27 + x^28        331
```

```
1 + x^29 + x^30         361
1 + x^45 + x^46         915
1 + x^59 + x^60         1709   primitiv
1 + x^62 + x^63         1915   primitiv
1 + x^126 + x^127              primitiv
1 + x^152 + x^153              primitiv
1 + x^171 + x^172       13995
1 + x^302 + x^303       44823
1 + x^470 + x^471       109433
1 + x^531 + x^532       139431
1 + x^864 + x^865       370059
1 + x^899 + x^900       400731
```

2.2.11 Алгоритм проверки системы нормального вида на базисность

Пусть $\{\alpha, \alpha^p, \alpha^{p^2}, \ldots, \alpha^{p^{n-1}}\}$ — последовательность элементов поля $GF(p^n)$. Для проверки того, образует ли она базис (естественно, нормальный) в этом поле, достаточно выяснить, является ли она линейно независимой над полем $GF(p)$. Для этого можно вычислить коэффициенты $a_{i,j}$ разложения элементов α^{p^2} по тому базису поля $GF(p^n)$, который используется в рассматриваемом представлении этого поля, и проверить $n \times n$ матрицу $(a_{i,j})$ на вырожденность. Матрица будет вырожденной тогда и только тогда, когда соответствующая ей система линейных уравнений с нулевой правой частью имеет ненулевое решение. Базис будет нормальным, если и только если эта матрица невырождена. Конечно, матрица будет вырожденной, если элементы последовательности $\{\alpha, \alpha^p, \alpha^{p^2}, \ldots, \alpha^{p^{n-1}}\}$ циклически повторяются. В противном случае для решения системы можно применить метод Гаусса, имеющий очевидную оценку сложности $O(n^3)$. В случае $p = 2$ работу этого алгоритма можно ускорить в 32 раза, если при выполнении элементарных преобразований столбцов матрицы представлять их в виде векторов из $n/32$ длинных целых чисел и для сложения столбцов по модулю два применять операцию XOR с длинными целыми числами. Объем используемой памяти при этом тоже уменьшается в 32 раза.

Еще большего ускорения можно достичь, используя быстрые алгоритмы линейной алгебры. Оценка сложности при этом равна $O(n^\omega)$, где ω — экспонента матричного умножения. Однако даже простейший из этих алгоритмов (алгоритм Штрассена) становится эффективным только при очень больших значениях n. При малых n лучше использовать алгоритм Коновальцева [43], который при $p = 2$ также можно ускорить за счет применения операций над длинными целыми числами. Его оценка сложности равна $O(n^3/\log_p n)$.

Для вычисления матрицы $(a_{i,j})$ можно использовать описанный в предшествующих разделах алгоритм вычисления последовательности $x, x^p, \ldots, x^{p^{n-1}}$ по модулю неприводимого многочлена $f(x)$, соответствующего полиномиальному базису рассматриваемого представления поля. Как отмечалось выше, при использовании в качестве $f(x)$ малочлена сложность этого вычисления будет $O(n^2 \log_2^2 p)$. Если же удается пользоваться представлением поля $GF(p^n)$ с помощью нормального базиса, то указанная сложность равна нулю.

Заметим, что если один нормальный базис $\{\alpha_1, \ldots, \alpha_n\}$ в поле $GF(p^n)$ известен, то все остальные можно выразить через него в явном виде путем умножения на *невырожденную циркулянтную матрицу* $(c_{i,j})$, $c_{i,j} = c_{i+1 \bmod n, j+1 \bmod n}$, т.е. такую, у которой каждая следующая строка получается из предыдущей циклическим сдвигом вправо. Подсчитывая число таких матриц (а это не просто, см. [45],[?]), отсюда можно получить точную формулу для числа нормальных базисов

$$\frac{1}{n} p^n \prod_{d|m} \left(1 - p^{-o_d(p)}\right)^{\phi(d)/o_d(p},$$

где $n = p^a m$, m не кратно p, знак $|$ означает делимость, ϕ — функцию Эйлера, а $o_d(p)$ минимальное натуральное число o, такое, что $p^o - 1$ кратно d. Например, число нормальных базисов в поле $GF(2^6)$ равно

$$\frac{1}{6}2^6\left(1-2^{-1)}\right)\left(1-2^{-2)}\right)=4$$

так как $\phi(3) = 2 = o_3(2)$. Несмотря на наличие такой формулы и на довольно большую вероятность случайно выбранного элемента α порождать нормальный базис, явной конструкции нормальных базисов для любой размерности, видимо, не известно.

2.2.12 Алгоритмы вычисления минимального многочлена

Если элемент α порождает нормальный базис, то он порождает и стандартный полиномиальный базис $\{1, \alpha, \ldots, \alpha^{n-1}\}$, так если минимальный многочлен f, аннулирующий элемент α, имеет степень $m < n$, то система $\{1, \alpha, \ldots, \alpha^{m-1}\}$ линейно независима, а все следующие степени через нее линейно выражаются, значит порожденное всеми степенями пространство $m-$мерно и не может совпадать с полем $GF(p^n)$. Поэтому минимальный многочлен f с единичным старшим коэффициентом должен иметь в рассматриваемом случае степень n. Если же α не порождает базис в данном поле, то его степень $m < n$ такова, что $\alpha^{p^m} = \alpha$, причем m минимальное такое число. Действительно, все элементы $\alpha, \alpha^p, \ldots, \alpha^{p^{m-1}}$ являются корнями многочлена f, а так как коэффициенты многочлена

$$g(x) = (x-\alpha)\ldots(x-\alpha^{p^{m-1}})$$

выражаются через его корни в виде симметрических многочленов согласно теореме Виета, то они не меняются при возведении в степень p (слагаемые и сомножители в формулах Виета циклически переставляются, что не меняет коэффициентов), значит они принадлежат полю $GF(p)$, а так как $g(x)$ делит $f(x)$, то они совпадают в силу неприводимости $f(x)$, поэтому

$$f(x) = (x-\alpha)\ldots(x-\alpha^{p^{m-1}}).$$

Из периодичности последовательности α^{p^i} следует, что m делит n. Вычисляя степени α^{p^i} также как в предыдущем разделе, минимальный многочлен элемента α, и в частности, многочлен, определяющий полиномиальный базис, порожденный α, можно найти путем умножения m многочленов над полем $GF(p^n)$. Применяя для умножения метод деления пополам (называемый иногда «разделяй и властвуй»), можно получить оценку сложности вычисления $f(x)$ в виде $O(M(n)\log n)M(GF(p^n))$, где $M(GF(p^n))$— сложность умножения в поле $GF(p^n)$, а $M(n)$ — сложность умножения многочленов степени $n-1$. При использовании асимптотически быстрых методов умножения многочленов и соответствующих методов умножения в конечных полях, более подробно описанных далее, можно получить оценку

$$O(n^2\log^3 n(\log\log n)^2 n\log p\log\log p\log\log\log p.$$

Однако при n больших многих тысяч эта оценка остается еще неэффективной. Используя вместо метода Шенхаге метод Карацубы, получаем оценку $O(M(n)^2 M(\log_2 p))$, где $M(n) = n^{\log_2 3}$, которая, однако будет хуже, чем у следующего очевидного метода, основанного на линейной алгебре.

Вычисляем со сложностью $nM(n)M(\log_2 p)$ $n\times(n+1)$-матрицу $(a_{i,j})$ координат элементов $\alpha^i, i = 0, n$ в данном поле и для нахождения коэффициентов минимального многочлена решаем однородную систему уравнений с этой матрицей. В $r-$мерном пространстве ее решений выбираем вектор с $r-1$ нулями в старших координатах $c_m = \ldots = c_{m-r+2} = 0$, и $c_{m-r+1} = 1$. Но для программирования, вероятно, первый метод проще.

2.2.13 Быстрый алгоритм вычисления минимального многочлена

Минимальный многочлен $f(x) = x^m + f_{m-1}x^{m-1} + \ldots + f_0$ элемента α позволяет быстро вычислять последовательность степеней α^i с помощью только линейных операций в поле $GF(p^n)$ по рекуррентной формуле

$$\alpha^{m+i} = \alpha^m\alpha^i = \alpha^i(f_{m-1}\alpha^{m-1} + \ldots + f_0) = f_{m-1}\alpha^{i+m-1} + \ldots + f_0\alpha^i.$$

Если вместо последовательности α^i взять последовательность $a_i \in GF(p)$ ее первых координат, то эта последовательность будет вычисляться в силу линейности по тем же рекуррентным формулам, которые можно также переписать в виде

$$a_{i+m} + f_{m-1}a_{i+m-1} + \ldots + f_0a_i = 0.$$

Многочлен минимальной степени, коэффициенты которого позволяют рекуррентно вычислять данную последовательность, называется минимальным многочленом этой последовательности.

Упражнение 2.2.1 Докажите, что любой другой многочлен над полем $GF(p)$, аналогичным способом вычисляющий данную последовательность, делится на минимальный многочлен.

Так как многочлен $f(x)$ неприводим, следовательно он является также минимальным многочленом последовательности a_i. Очевидно, его коэффициенты могут быть найдены путем решения линейной системы уравнений

$$a_{i+m} + f_{m-1}a_{i+m-1} + \ldots + f_0a_i = 0, i = 0, \ldots, m-1.$$

Увеличивая m от 1 до n, можно найти минимальное m, при котором система имеет единственное решение. Так как ее матрица коэффициентов имеет специфический вид (похожа на циркулянтную матрицу), то неудивительно, что у задачи вычисления минимального многочлена данной рекуррентной последовательности есть более простое решение, и не одно. Один из алгоритмов для решения этой задачи называется алгоритм Берлекемпа-Месси, и имеет сложность $O(n^2)M(\log_2 p)$ (см. [10].) Известно также, что для ее решения подходит соответствующая модификация алгоритма Евклида. Как уже отмечалось, ее можно реализовать со сложностью $O(M(n)M(\log_2 p)\log n)$, где $M(n)$ — сложность умножения многочленов степени $n-1$. Сложность вычисления последовательности $a_0, \ldots, a_{2n-1}$ можно оценить также как и сложность вычисления последовательности $\alpha_0, \ldots, \alpha_{2n-1}$, а именно, как $(2n-2)M(GF(p^n)) = O(nM(n)M(\log_2 p)$. Степень m многочлена f находится очевидно со сложностью $O(n^2M(\log_2 p))$ одновременно с вычислением степеней $\alpha^{p^i}, i = 1, \ldots, n-1$. После этого достаточно будет вычислить только $\alpha^i, i = 2, \ldots, 2m-1$. Некоторую экономию сложности можно достичь, вычисляя $\alpha^{pi} = (\alpha^i)^p$ с помощью более быстрой, чем умножение операции возведения в степень p, которая делается с линейной сложностью при хорошем выборе базиса в поле $GF(p^n)$. Остальные степени можно вычислять по формуле $\alpha^{pi+s} = \alpha^{ip}\alpha^s, s < p$. Окончательная оценка сложности вычисления минимального многочлена данного элемента в поле $GF(p^n)$ равна

$$2n(1-1/p)M(GF(p^n)) + O(n^2)M(\log_2 p).$$

2.2.14 Еще об алгоритмах тестирования базисности и нормальности

Известны неэффективные на практике алгоритмы, которые однако удобны тем, что сводят проверку «базисности» данной системы к вычислению некоторых явных формул. Например,

для выяснения, образует ли в поле $GF(p^n)$ базис система элементов $\{\alpha_1, \ldots, \alpha_n\}$ достаточно проверить, не обращается ли в нуль определитель $n \times n$ матрицы, состоящей из элементов $Tr(\alpha_i \alpha_j)$.

Упражнение 2.2.2 Докажите, что эта система образует базис тогда и только тогда, когда указанный определитель отличен от нуля.

Непосредственное вычисление следа $Tr(\alpha)$ по формуле

$$Tr(\alpha) = \alpha + \alpha^p + \ldots + \alpha^{p^{n-1}}$$

имеет сложность $O(n^2)M(\log_2 p)$ в хорошем полиномиальном базисе и сложность $O(n^2 \log_2 p)$ в нормальном базисе. Заметив, что след является линейным отображением поля $GF(p^n)$ в подполе, его можно вычислить со сложностью $O(nM(\log_2 p))$. Поэтому сложность вычисления всех элементов матрицы равна $O(n^2 M(GF(p^n))$. Сложность вычисления определителя равна $O(n^2 I(\log_2 p) + n^3 M(\log_2 p)$, где $I(\log_2 p)$ — сложность инвертирования в поле $GF(p)$. При малых p окончательно имеем оценку $O(n^2 M(GF(p^n))$.

Замечая, что

$$Tr(\alpha_i \alpha_j) = \alpha_i \alpha_j + \alpha_i^p \alpha_j^p + \ldots + \alpha_i^{p^{n-1}} \alpha_j^{p^{n-1}},$$

и представляя матрицу из следов в виде произведения матрицы $\alpha_i^{p^j}$ на транспонированную к ней, получаем, что наша система будет базисом если и только если определитель новой матрицы отличен от нуля.

Упражнение 2.2.3 Докажите это утверждение.

Сложность вычисления новой матрицы уменьшается по сравнению со старой, а в случае представления поля $GF(p^n)$ нормальным базисом, и вовсе становится равной нулю. Но элементы матрицы теперь принадлежат полю $GF(p^n)$ и поэтому сложность вычисления определителя возрастает до $O(n^2 I(GF(p^n)) + n^3 M(GF(p^n))$.

Применяя последний алгоритм к тестированию на базисность системы $\{\alpha, \alpha^p, \alpha^{p^2}, \ldots, \alpha^{p^{n-1}}\}$ (тестированию нормальности), получаем для проверки на невырожденность в точности циркулянтную матрицу. Невырожденность таких матриц можно проверить быстрее. Заметим вначале, что любая такая матрица $C = (c_{i,j})$ представляется в виде

$$C = c_0 E + c_1 P + \ldots + c_{n-1} P^{n-1},$$

где $c_0, \ldots, c_{n-1}$ — верхняя строка матрицы C, а P — матрица перестановки, т.е. циркулянтная матрица с n единицами, стоящими на позициях $c_{0,1}, \ldots, c_{n-2,n-1}, c_{n-1,0}$, и нулями в остальных местах, а P^i — ее степени (тоже циркулянтными матрицами, но других перестановок.)

Упражнение 2.2.4 Докажите это утверждение.

Так как $P^n = E$ — единичной матрице, умножение циркулянтных матриц выполняется по формуле

$$AB = \sum_{k=0}^{n-1} \left(\sum_{i=0}^{n-1} a_i b_{k-i,} P^k \right),$$

где a_i элементы первой строки матрицы A, b_i элементы первой строки матрицы B, а вычитание в индексах делается по модулю n.

Упражнение 2.2.5 Докажите это утверждение. Докажите также, что множество всех циркулянтных матриц относительно матричного сложения и умножения образует кольцо.

Заметим теперь, что умножение многочленов степени меньше n по модулю x^n-1 выполняется по тем же формулам, что и умножение циркулянтных матриц.

Упражнение 2.2.6 Докажите, что кольцо циркулянтных матриц изоморфно кольцу многочленов с операциями по модулю x^n-1.

Теперь ясно, что циркулянтная матрица обратима (а это равносильно ее невырожденности) если и только если соответствующий ей при изоморфизме многочлен степени меньше n обратим по модулю x^n-1.

Упражнение 2.2.7 Докажите, что многочлен обратим по модулю x^n-1 если и только если он взаимно прост x^n-1 (не имеет с ним общих делителей, кроме констант).

Отсюда сразу вытекает следующий критерий нормальности [45]. Пусть $\{\alpha, \alpha^p, \alpha^{p^2}, \ldots, \alpha^{p^{n-1}}\}$ — последовательность элементов поля $GF(p^n)$. Чтобы она была базисом необходимо и достаточно, чтобы многочлены x^n-1 и

$$f(x) = \alpha x^{n-1} + \alpha^p x^{n-2} + \alpha^{p^2} x^{n-3} + \ldots + \alpha^{p^{n-1}}$$

были взаимно просты над полем $GF(p^n)$ (а значит, согласно свойствам алгоритма Евклида, и над любым расширением этого поля).

2.2.15 Быстрый алгоритм проверки системы нормального вида на базисность

Если применить для вычисления НОД этих многочленов быстрый алгоритм Евклида и алгоритм Шенхаге для умножения , то оценка сложности будет

$$O(M(n))M(GF(p^n))\log n = O(n^2\log^3 n(\log\log n)^2 n\log p\log\log p\log\log\log p.$$

Однако мультипликативная константа в этой оценке даже при $p=2$ будет не меньше $64^2\cdot 12$. Используя для умножения многочленов ДПФ, можно $M(n)$ заменить на $3F(n)+2n$, где $F(n)$ — сложность вычисления преобразования Фурье порядка n в минимальном расширении $GF(p^m)$ поля $GF(p^n)$, содержащем все n корней n-степени из единицы. Тогда $m=nk$, и n должно делить p^m-1, а оценку $M(GF(p^n)))$ нужно заменить на $M(GF(p^m))$.

Указанную оценку можно улучшить, отбросив множитель $O(\log n)$, если заметить, что взаимная простота рассматриваемых выше многочленов над полем $GF(p^m)$ равносильна тому, что $f(x)$ не имеет общих корней с x^n-1, значит все его значения в корнях n-й степени из единицы отличны от нуля. Но вычисление всех этих значений в поле $GF(p^m)$ есть не что иное, как вычисление преобразования Фурье F_n, поэтому мы получаем оценку $F(n)M(GF(p^m))$. Разлагая n на простые множители $n=p_1^{\beta_1}\ldots p_r^{\beta_r}$ и применяя алгоритмы Гуда-Томаса и Кули-Тьюки (см. [10]), получаем оценки

$$F(n) = n\left(\frac{F(p_1^{\beta_1})}{p_1^{\beta_1}} \ldots + \frac{F(p_r^{\beta_r})}{p_r^{\beta_r}}\right) \leq$$

$$\leq n\left(\beta_1\left(\frac{F(p_1)}{p_1}+1\right) + \ldots + \beta_r\left(\frac{F(p_r)}{p_r}+1\right)\right).$$

В случае гладкого числа n (то есть когда все p_i малы) получается оценка, близкая к $O(n\log n)$.

Если же какое-нибудь p_i велико, то для оценки $F(p_i)$ можно применить основанный на применении китайской теоремы об остатках алгоритм Винограда ([10]) вычисления циклической свертки, разложив $x^{p_i}-1$ на круговые множители

$$\prod_{d|p_i} Q_d,$$

сначала над кольцом целых чисел, потом над полем $GF(p)$, которые будут иметь коэффициенты в поле $GF(p)$ и сравнительно мало ненулевых коэффициентов, и поэтому на них проще делить обычным школьным алгоритмом, потом полученные множители разложить над более широким полем $GF(p^m)$ и продолжить применение китайской теоремы. При больших n имеется быстрый вариант как прямого, так и обратного китайского алгоритма, описанный например в [5] (с дефектом) и корректно в [101].

В связи с этим заметим, что круговой многочлен Q_n степени $\phi(n)$ над кольцом целых чисел в поле $GF(p^m)$ разлагается на неприводимые множители равной степени $d = \phi(n)/o_n(p^m)$, где $o_n(p^n)$ минимальное натуральное число o такое, что $p^{mo} - 1$ кратно n. Быстрый алгоритм разложения на неприводимые множители равной степени описан в [101] и [17].

Можно также методом Райдера ([10]) свести вычисление $F(p_i)$ к вычислению циклической свертки порядка $p_i - 1$, которая в свою очередь сводится к трехкратному применению преобразования Фурье порядка $p_i - 1$, которое придется вычислять над полем $GF(p^{m_i})$, где m_i/m — порядок числа p^m по модулю $p_i - 1$.

Отметим, что в некоторых случаях n может быть делителем $p^n - 1$, и значит в качестве $m = nk$ можно взять просто n. Например, возьмем $n = p^{\nu_1} - 1$, тогда порядок p по модулю n равен ν_1, значит $m = \frac{n\nu_1}{(n,\nu_1)}$ — наименьшее общее кратное чисел n и ν_1,

Покажем, что при $p > 2$ можно выбрать ν_1 так, чтобы n делилось на ν_1, тогда $(n, \nu_1) = \nu_1$ и m будет равно n. Но $n = p^{\nu_1} - 1$, и для того, чтобы оно делилось на ν_1 можно выбрать $\nu_1 = \nu_2 p^{\nu_2} - 1$, и так далее, и для некоторого s выберем $\nu_s = p - 1$, тогда $\nu_{s-1} = p^{\nu_s} - 1$ будет кратно ν_s и мы получаем бесконечную последовательность n_s, рекуррентно определяемую равенствами $n_s = n_{s-1}(p^{n_{s-1}} - 1)$, $n_1 = p - 1$, такую, что $p^{n_s} - 1$ делится на n_s. Действительно, $p^{n_1} - 1 = (n_1 + 1)^{n_1} - 1$ делится на n_1, и далее проверяется, что

$$p^{n_s} - 1 = p^{n_{s-1}(p^{n_{s-1}} - 1)} - 1$$

делится на $p^{n_{s-1}} - 1$, в частном получается $1 + a + a^2 + \ldots + a^{b-1}$, где $a = p^{n_{s-1}}, b = a - 1$, а это число по модулю b равно сумме b единиц то есть равно нулю по модулю b, значит указанное частное делится на n_{s-1}, так как $b = p^{n_{s-1}} - 1$ делится на n_{s-1}, согласно предположению индукции, поэтому $p^{n_s} - 1$ делится на $bn_{s-1} = p^{n_{s-1}} n_{s-1} = n_s$.

Если же $p = 2$, то n не может быть делителем $2^n - 1$. Действительно, пусть q — минимальный простой делитель n, тогда q делит одновременно $2^{q-1} - 1$ и $2^n - 1$, а значит, и их НОД, равный $2^{(q-1,n)} - 1 = 1$, так как $(q - 1, n) < q$, делит n и, согласно выбору p равно 1. Из доказанного следует, что НОД $(n, 2^n - 1)$ делит n/p. Пример, когда $(n, 2^n - 1) = n/q$ очевиден — просто берем $n = q$. Начиная с $n_1 = q$ определим рекуррентно последовательность равенствами $n_s = n_{s-1}(2^{n_{s-1}} - 1)$, тогда, рассуждая по индукции, замечаем, что $n_s/q = (2^{n_{s-1}} - 1)n_{s-1}/q$ делит $2^{n_s} - 1 = 2^{n_{s-1}(2^{n_{s-1}} - 1)} - 1$, потому, что при делении $2^{n_{s-1}(2^{n_{s-1}} - 1)} - 1$ на $(2^{n_{s-1}} - 1)$, как было проверено раньше, получается $1 + a + a^2 + \ldots + a^{b-1}$, где $a = 2^{n_{s-1}}, b = a - 1$, и это число делится на $b = 2^{n_{s-1}} - 1$, а значит, согласно предположению индукции, и на qn_{s-1}/q, поэтому $2^{n_s} - 1 = 2^{n_{s-1}(2^{n_{s-1}} - 1)} - 1$ делится на $(2^{n_{s-1}} - 1)n_{s-1}/q = n_s/q$. Поэтому для всех членов этой последовательности справедливо, что $(2^{n_s} - 1, n_s) = n_s/q$, значит, взяв $n = 2^{n_s} - 1$, получаем, что $m = \frac{nn_s}{(n,n_s)} = nq$.

2.2.16 Быстрый алгоритм тестирования на неприводимость

Здесь мы для алгоритма, изложенного в предыдущих разделах, докажем приведенную там же оценку оценку сложности, следуя [101].

2.2.17 Быстрая модулярная композиция многочленов

Модулярной композицией называется операция $g(h)$ композиции многочленов f, g вычисленная по модулю данного многочлена $f(x)$. Покажем, следуя [86], [101], что ее можно выполнить для многочленов степени меньшей $n = \deg f$ со сложностью

$$O(n^{(\omega+1)/2}) + 6n^{1/2}(M(n) + O(n)),$$

где ω — экспонента матричного умножения, $M(n)$— сложность умножения многочленов степени n. В частности, если для $M(n)$ использовать достаточно хорошую оценку Тоома [39], то для модулярной композиции будем иметь оценку $O(n^{(\omega+1)/2})$. На практике при n порядка тысячи эффективнее применять для умножения алгоритм «четырех русских» с оценкой сложности $O(\log_2 p)n^3/\log_p n$. Тогда применяя алгоритм умножения Тоома с оценкой $O(n^{1.4})$, будем иметь оценку для модулярной композиции

$$O(log_2 p)n^2/\log_p n + O(n^{1.9}).$$

В случае программной имплементации при $p = 2$ память может быть оценена как $n^{3/2}$, а время работы как

$$O(n^2)/(w\log_2 n) + O(n^{1.9}).$$

Использование алгоритма Карацубы еще не позволяет получить квадратичную оценку сложности.

Алгоритм модулярной композиции основан на следующей идее. Положим $m = \lceil n^{1/2} \rceil$ и представим $g(x)$ в виде

$$g = \sum_{i=0}^{m-1} g_i x^{mi},$$

где степени многочленов g_i меньше m. Очевидно, что это делается бесплатно. Вычислим $h^i \bmod f, i = 2, \ldots, m$. Если заранее вычислить многочлен $R(x)$ такой, что $R(x)f(x) = x^{2n} + r(x), \deg r(x) < n$, то модулярное вычисление всех этих степеней имеет сложность $3m(M(n) + O(n))$ и требует памяти $O(mn) = O(n^{3/2}$. Составим прямоугольную $m \times n$ матрицу A, строки которой состоят из коэффициентов многочленов

$$h^i \bmod f = \sum_{i=0}^{n-1} h_{i,j} x^j, i = 0, \ldots, m-1,$$

и квадратную $m \times m$ матрицу B, строки которой состоят из коэффициентов многочленов

$$g_i == \sum_{j=0}^{m-1} g_{i,j} x^j, i = 0, \ldots, m-1$$

и вычислим их произведение BA, разбивая матрицу A на m квадратных подматриц (одна из них может быть не квадратной, тогда превратим ее в квадратную, дополнив нулями), и выполняя m раз умножение квадратных $m \times m$ матриц со сложностью $mO(m^\omega) = O(n^{(\omega+1)/2})$. Заметим, что в полученном произведении BA в i строке будут стоять коэффициенты многочлена

$$r_i(x) = g_i(h) \bmod f,$$

так как при смене порядка суммирования имеем

$$r_i = \sum_{k=0}^{n-1} \sum_{j=0}^{m-1} g_{i,j} h_{j,k} x^k = \sum_{j=0}^{m-1} g_{i,j} \sum_{k=0}^{n-1} h_{j,k} x^k =$$

$$= \sum_{j=0}^{m-1} g_{i,j}(h^j \bmod f) = \sum_{j=0}^{m-1} g_{i,j} h^j \bmod f = g_i(h) \bmod f.$$

Окончательно, находим модулярную композицию по формуле

$$g(h) \bmod f = \sum_{i=0}^{m-1} g_i(h) h^{mi} \bmod f = \sum_{i=0}^{m-1} r_i h^{mi} \bmod f$$

применяя схему Горнера, делая m умножений и сложений многочленов степени $n-1$ по модулю f. Сложность этих вычислений равна $m(3M(n)+O(n))$ и памяти для них требуется $O(n)$. Алгоритм модулярной композиции требует еще памяти $O(n^{3/2})$ для запоминания матрицы B и произведения AB.

2.2.18 О схемной реализации модулярной композиции

Полученные выше оценки сложности справедливы и для схемной реализации, однако в случае использования «алгоритма четырех русских» это верно лишь когда один из многочленов фиксирован. Зато в этом случае можно, например, при $p = 2$ можно указать оценку с конкретной мультипликативной константой в главном члене

$$n^2/\log_2 n + O(n^2 \log_2 \log_2 n/\log_2^2 n) + 6n^{1/2}(M(n) + O(n)).$$

Кроме того, в этом случае можно получить оценку для глубины вида $O(\log_2 n)^2$ и даже вида $O(\log_2 n)$, если фиксирован внутренний многочлен композиции. Действительно, для одновременного вычисления системы степеней от первой до n-й существует схема глубины $\lambda(n)+1$ из $n + o(n)$ операций умножения.

Упражнение 2.2.8 Докажите это.

Известно, что схема для быстрого модулярного умножения многочленов имеет сложность $O(n\lambda(n)\lambda(\lambda(n))$ и глубину $D(n) = O(\lambda(n))$. Поэтому вычисление системы многочленов $h^i \bmod f, i = 2, \ldots, m$ методом Монтгомери выполняется с глубиной $(\lambda(m)+1)D(n)$ и сложностью $(3+o(1))m(M(n)+O(n))$. Если многочлен h фиксирован, то и сложность и глубина равны нулю. Глубина схемы Лупанова для умножения фиксированной $n \times n$- матрицы на произвольную оценивается как $\log_2 n + O(1)$. Поэтому глубина части схемы для модулярной композиции, осуществляющей перемножение матриц, не больше

$$(\lambda(m)+1)D(n) + \frac{1}{2}\log_2 n + O(1) = (\lambda(m)+1)(D(n)+1) + O(1).$$

Используя указанную выше схему для совместного вычисления степеней, схему Горнера для многочленов n-й степени можно модифицировать так, чтобы число умножений в ней увеличилось асимптотически в два раза, а глубина мультипликативной и аддитивной части схемы была бы $\lambda(n)+1$.

Упражнение 2.2.9 Докажите это.

На самом деле схему Горнера можно модифицировать так, что ее глубина будет асимптотически равна $\lambda(n)$, а сложность останется $O(n)$.

Поэтому, после того, как вычислен многочлен $h^m \bmod f$, вычисление окончательного результата с помощью модифицированной схемы Горнера можно выполнить с глубиной

$$(\lambda(m)+2)(D(n)+1) = (\frac{1}{2}\lambda(n)+2)(D(n)+1) = O(\lambda(n)^2)$$

и сложностью

$$n^2/\log_2 n + O(n^2 \log_2\log_2 n/\log_2^2 n) + O(n^{1/2})(M(n)+O(n)).$$

Если многочлен h фиксирован, то в случае схемной реализации вместо схемы Горнера можно выполнить параллельно m умножений по модулю f, а потом $m-1$ сложение многочленов. Сложность этой схемы равна $mM(n)+(m-1)n$, а глубина $\log_2 m + D(M(GF(2^n))) = O(\log_2 n)$, если используется умножение методом Шенхаге или методом Тоома. Поэтому в этом случае окончательно имеем оценку глубины $O(\log_2 n)$ при сложности

$$n^2/\log_2 n + O(n^2 \log_2\log_2 n/\log_2^2 n) + n^{1/2}(M(n)+n).$$

2.2.19 О возможности использования алгоритма Штрассена умножения матриц

Этот алгоритм сводит умножение $n \times n$-матриц к 7 умножениям $n/2 \times n/2$ матриц и 15 сложениям таких же матриц. Для битовой сложности (и для схемной сложности) он имеет рекуррентную оценку $T(n) \le 7T(n/2)+15n^2/4$, а для глубины — рекуррентную оценку $D(n) \le D(n/2)+4$, откуда имеем $D(n) \le 4\log_2 n + O(1)$, $T(n) \le 42n^{\log_2 7}$ и для сложности модулярной композиции

$$42n^{\log_2 7} + 6n^{1/2}(M(n)+O(n)).$$

В случае схемной реализации модулярной композиции при фиксированном внутреннем многочлене сложность оценивается как

$$42n^{\log_2 \sqrt{14}} + n^{1/2}(M(n)+n)$$

а глубина как $O(\log_2 n)$. Для уменьшения сложности умножения матриц можно матрицы до определенного размера умножать методом «четырех русских», а потом начать применять штрассеновские итерации. В случае схемной реализации это можно делать только когда произвольная матрица умножается на фиксированную. В этом случае 4 сложения подматриц этой матрицы делать не надо, и рекуррентная оценка заменяется на более точную $T(n) \le 7T(n/2) + 11n^2/4$. На самом деле для $n = 2^k$ в этом случае получается оценка $T(n) \le (14/3)n^{\log_2 7}$, а также оценка

$$T(2^k n) \le 7^k(T(n) + (11/3)n^2),$$

откуда следует, что

$$(84/3)n^{\log_2 \sqrt{14}} + n^{1/2}(M(n)+n).$$

Сравнивая с оценкой $n^3/\log_2 n$ алгоритма «четырех русских», получаем, что итерацию Штрассена выгодно делать, начиная с n, такого, что

$$8n^3/\log_2 2n > 7n^3/\log_2 n + 77n^2/3,$$

т.е. начиная с n порядка 200. Так как в модулярной композиции используется умножение матриц размера $n^{1/2}$, то эффект от использования штрассеновской итерации ожидается только начиная с $n = 40000$.

Еще один способ ускорения штрассеновского алгоритма умножения произвольной двоичной матрицы на фиксированную матрицу предлагается в следующем упражнении.

Упражнение 2.2.10 Постройте из элементов XOR схему для умножения матриц сложности асимптотически

$$\frac{77}{6}\frac{n^{\log_2 7}}{(\log_2 n)^{\log_2 7/4}}$$

при $n = 2^k k$.

Указание. Разбейте матрицу на подматрицы размера $\frac{1}{2}\log_2 n$ и вычислите заранее $n^{5/2}/\log_2 n$ линейных функций от $\frac{1}{2}\log_2 n$ переменных. Тогда умножение подматриц размера $\frac{1}{2}\log_2 n$ делается бесплатно.

Так как практически нас интересуют n порядка 200, то при схемной реализации вместо использования модулярного умножения для реализации любого линейного оператора Фробениуса проще непосредственно вычислить матрицы этих операторов и реализовать их схемами из элементов XOR, построенными с помощью какой-нибудь программной эвристики с глубиной не больше $\log_2 n$ и сложностью не больше $2n^2/\log_2 n$.

В случае программной имплементации строки матриц имеет смысл хранить в виде векторов длины n/w из машинных слов длины w. При сложении подматриц в алгоритме Штрассена и в алгоритме « четырех русских» используем операцию побитового XOR для машинных слов. Вместо $\log_2 n$ в алгоритме « четырех русских» используем подходящую степень двойки. Тогда блоки такой длины можно будет целиком выбирать из соответствующих машинных слов. В результате скорость работы программы увеличится в w раз по сравнении с побитовой организацией вычислений.

Однако на практике при имплементации модулярного произведения требуется умножать матрицы порядка 20 и вероятно достаточно будет использовать только алгоритм «четырех русских» при выборе длины блока 4.

2.2.20 Оценка сложности тестирования неприводимости многочленов

Здесь мы докажем несколько более точную, чем данная в [101], оценку

$$O\left(M(n)\log_2 p + \left(n^{(\omega+1)/2} + n^{1/2}M(n)\right)\left(log_2 n + d(n)\frac{\log_2 n}{\log_2\log_2 n}\right)\right),$$

где $d(n)$ —число различных простых делителей числа n, $M(n)$ — сложность умножения многочленов степени n, а ω —- экспонента матричного умножения.

Напомним указанный ранее алгоритм тестирования неприводимости многочлена q степени n. Построим последовательность полиномов $q_{k+1} = q_k^p \bmod q$ начиная с полинома $q_0 = x$. Очевидно, что $q_k = x^{p^k} \bmod q$ имеет степень меньше n. Полином q будет неприводимым тогда и только тогда, когда $q_n = x \bmod q$ и для любого простого делителя s числа n наибольший общий делитель многочленов $q_{n/s} - x$ и q будет равен 1. Напомним, что число различных простых делителей числа n равно $d(n) = O((\ln n)/\ln\ln n$, а почти всегда $d(n)$ асимптотически равно $\ln\ln n$. Так как сложность вычисления *НОД* многочленов степени n равна $O(M(n)\log n)$, то сложность вычисления всех $(q_{n/s} - x, q)$ оценивается как $O(d(n)M(n)\log n) = O(M(n)\log^2 n))$.

Оценим сложность вычисления всех $q_{n/s}$. Так как $q_1 = x^p \bmod q$, то сложность вычисления q_1 равна $((l(p)+1)(3M(n)+O(n))+6M(n) = O(\log_2 pM(n)$, где $l(p)$ — длина кратчайшей аддитивной цепочки для числа n. Указанная первой более точная оценка получается методом Монтгомери, а более грубая вторая очевидно получается с помощью бинарного метода вычисления аддитивных цепочек. Заметим, что q_{i+j} вычисляется с помощью модулярной композиции при известных q_i, q_j по формуле $q_{i+j} = q_i(q_j) \bmod q$. Действительно, применяя тождество Фробениуса, имеем

$$q_{i+j}(x) = x^{p^{i+j}} \bmod q = (x^{p^i} \bmod q)^{p^j} \bmod q = q_i^{p^j}(x) \bmod q =$$

$$= q_i\left(x^{p^j}\right) \bmod q = q_i\left(x^{p^j} \bmod q\right) \bmod q = q_i(q_j) \bmod q.$$

Хотя операция модулярной композиции вообще говоря некоммутативна, но в рассматриваемом тождестве многочлены q_i, q_j можно поменять местами. Благодаря полученному тождеству вычисление системы многочленов $\{q_{n/s}\} = \{q_{n_1}, \ldots, q_{n_d},\}, n_1 = n, d = d(n)$ может быть

выполнено со сложностью

$$((l(p)+1)(3M(n)+O(n))+6M(n)+l(n_1,\ldots,n_d)O(n^{(\omega+1)/2}+n^{1/2}M(n))$$

так как после вычисления q_1 требуется $l(n_1,\ldots,n_d)$ раз применить операцию модулярного умножения. Применяя теорему Яо, получаем, что

$$l(n_1,\ldots,n_d)=\log_2 n+O(d)\frac{\log_2 n}{\log_2\log_2 n}=O(\frac{\log_2^2 n}{(\log_2\log_2 n)^2}).$$

Так как для почти всех n как известно $d(n)=O(\log\log n)$, то $l(n_1,\ldots,n_d)=O(\log_2 n)$. Указанные выше оценки можно уточнить, используя упоминавшуюся ранее оценку [24]

$$l(n_1,\ldots,n_d)=\log_2 n++O(d)+\frac{R}{\log_2 R}\left(1+O\left(\left(\frac{\log_2\log_2 R}{\log_2 R}\right)^{1/2}\right)\right),$$

где $R=n^d/m$, m — произведение всех простых делителей n. Если число n свободно от квадратов, то $m=n$ и $R=n^{d-1}$.

Из полученных выше оценок очевидно имеем окончательную оценку

$$O\left(M(n)\log_2 p+\left(n^{(\omega+1)/2}+n^{1/2}M(n)\right)\left(log_2 n+d(n)\frac{\log_2 n}{\log_2\log_2 n}\right)\right),$$

так второе слагаемое в ней очевидно превосходит оценку $O(M(n)\log^2 n))$ сложности вычисления НОД.

Очевидно полученная оценка будет справедлива и для тестирования многочленов над произвольным конечным полем $GF(q)$, нужно в ней только заменить p на q. Первое слагаемое в полученной оценке сохранено только из-за возможного случая, когда p (или q) велико в сравнении с n. Тогда этот член в сумме может стать главным, и его имеет смысл заменить на указанную выше более точную оценку. Если же p мало, например $p=2$, то первый член в оценке можно отбросить, и тогда она принимает вид

$$O\left(\left(n^{(\omega+1)/2}+n^{1/2}M(n)\right)\left(log_2 n+d(n)\frac{\log_2 n}{\log_2\log_2 n}\right)\right).$$

Если для умножения многочленов используется достаточно быстрый алгоритм, например такой, что $M(n)=O(n^{\omega/2})$, то оценка принимает вид

$$O\left(n^{(\omega+1)/2}\left(log_2 n+d(n)\frac{\log_2 n}{\log_2\log_2 n}\right)\right).$$

Выше отмечалось, что с учетом новейших результатов о матричном умножении, окончательная асимптотическая оценка принимает вид $O(n^1.668)$.

При n порядка нескольких десятков тысяч более реалистичная оценка сложности модулярного умножения, как отмечалось, имеет в лучшем случае вид $(6+\epsilon_n)n^2/\log_p n$ (для этого нужно применять алгоритм умножения со сложностью $M(n)=O(n^{1.4})$). Используя стандартный алгоритм Евклида с оценкой сложности $GD(n)<5n^2/2+3n/2$ тогда имеем окончательную оценку

$$(6+\epsilon_n)\frac{n^2}{\log_p n}\left(log_2 n+(1+\epsilon_n)d(n)\frac{\log_2 n}{\log_2\log_2 n}\right)+$$

$$+d(n)(5n^2/2+3n/2)=(2.5+\epsilon_n)n^2 d(n)=O(n^2\log n/\log\log n),$$

в которой главным членом оказывается оценка сложности вычисления *НОД*. Асимптотически оценку сложности вычисления *НОД* многочленов степени n можно конечно оценить как $(24+\epsilon_n)M(n)\log_2 n$, где $M(n)$ можно оценить методом Карацубы как $(35/3)\,n^{\log_2 3}$. Однако эта оценка становится лучше оценки $5n^2/2$ только при n равной многим миллионам. Возможно при таких больших n использование методов умножения Шенхаге или Кантора ускорит выполнение алгоритма Евклида, но при n порядка десятков тысяч это сомнительно.

2.3 Генерация оптимальных нормальных базисов

2.3.1 Три типа оптимальных нормальных базисов

Оптимальные нормальные базисы были обнаружены Mullin, Onyszchuk, Vanstone, Wilson в работе [138]. Они удачно (как было описано выше) могут быть использованы в мультиплере, запатентованном в 1985 г. Massey и Omura. Впоследствии Gao и Lenstra ([97]) показали, что других оптимальных нормальных базисов, кроме найденных в [138], не существует.

Поскольку оптимальные нормальные базисы существуют не во всех полях, то представляют интерес базисы если и не оптимальные, но имеющие низкую сложность. В [80]. такие базисы строятся с помощью так называемых *гауссовых периодов*, изучению приложений которых посвящены также статьи [98],[99]. Об этих базисах речь пойдет позднее.

Все упомянутые базисы рекомендуются в стандарте NIST для имплементации криптосистем, основанных на использовании эллиптических кривых. Однако ни в этом стандарте, ни в доступной литературе не объясняется, как именно реализовывать арифметику в этих базисах. В [?] описаны, например, только автоматные схемы, а также схема Месси-Омура, которая, как будет объяснено ниже, все-таки имеет слишком высокую сложность по сравнению с быстрыми алгоритмами умножения в стандартных базисах.

Различают три типа оптимальных нормальных базисов в поле $GF(q^n)$ по типу их построения.

Первый тип оптимальных нормальных базисов можно построить, когда $n+1=p$ — простое, а q — примитивный корень по модулю p.

В этом случае генератором оптимального нормального базиса будет один из примитивных корней ζ p-й степени из единицы в поле $GF(q^n)$.

Второй тип оптимальных нормальных базисов возникает, когда $2n+1=p$ — простое, а q - как и в первом случае, – примитивный корень по модулю p.

Генератором этого базиса служит элемент $\alpha=\zeta+\zeta^{-1}$, где ζ – примитивный корень p-й степени из единицы в поле $GF(q^{2n})$.

Упражнение 2.3.1 Докажите, что если в поле $GF(q^n)$ существует базис второго типа, то в поле $GF(q^{2n})$ существует базис первого типа, который порождается элементом ζ – примитивным корнем p-й степени из единицы в поле $GF(q^{2n})$. Верно и обратное, если в поле $GF(q^{2n})$ существует базис первого типа, то в поле $GF(q^n)$ существует базис второго типа.

Третий тип оптимальных нормальных базисов порождается, когда $2n+1=p$ – простое, $p\equiv 3(\bmod 4)$, а q – квадратичный вычет по модулю p и любой квадратичный вычет представляется в виде степени q по модулю p (или, другими словами, порядок числа q по модулю p равен n).

Как и в случае базиса второго типа в качестве порождающего элемента базиса третьего типа берется $\alpha=\zeta+\zeta^{-1}$, где ζ – примитивный корень p-й степени из единицы в поле $GF(q^{2n})$.

Оптимальные базисы с ростом n встречаются все реже и реже. Согласно гипотезе Артина [81] для любого $a\neq 0,\pm 1$ и не равного полному квадрату число простых $n\leq x$, таких что a является примитивным корнем по модулю n асимптотически равно $c(a)x/\log_2 x$, где $c(a)$ — зависящая от a константа. В предположении справедливости ERH (расширенной гипотезы Римана) гипотезу Артина доказал Хули [108]. Из этих утверждений следует, например, что для конечной доли всех простых n существуют оптимальные нормальные базисы в полях $GF(2^n)$.

2.3.2 Таблица оптимальных нормальных базисов для $q=2$ и $10\leq n\leq 30$

В этой таблице, сгенерированной программой, указаны все оптимальные нормальные базисы для $10\leq n\leq 30$ и приведены соответствующие матрицы A и T, заданные перечислением

единичных элементов. Например, запись вида $A : (i, j) = (4, 3)(1, 0)$ означает, что элемент на пересечении строки 4 и столбца 3 единичный и элемент на пересечении строки 1 и столбца 0 тоже единичный (нумерация строк и столбцов начинается с нуля). Тоже самое относится и к заданию матрицы T, а запись $(i, j, j) = (2, 6, 8)$ означает, что в строке 2 единицы стоят на местах 6 и 8 и запись $T : (i) = (5)$ означает, что диагональный элемент с номером 5 равен единице. Кроме того, в таблице указываются многочлены, которые использовались для стандартного задания поля (в случае базисов второго и третьего типов речь идет о квадратичном расширении поля, в котором строится базис), и приведены порождающие элементы базисов в этом стандартном задании (хотя эти элементы реально нигде не используются).

```
n = 10

 1 тип:

Матрицы: T и A
    m=0  T : (i) = (5)     A : (i, j) =  (5, 0) (4, 9) (3, 8)
 (2, 7) (1, 6) (0, 5) (9, 4) (8, 3) (7, 2) (6, 1)
    m=1  T : (i, j) = (0, 1)    A : (i, j) = (9, 9)
    m=2  T : (i, j) = (1, 8)    A : (i, j) = (3, 2)
    m=3  T : (i, j) = (8, 2)    A : (i, j) = (6, 8)
    m=4  T : (i, j) = (2, 4)    A : (i, j) = (8, 6)
    m=5  T : (i, j) = (4, 9)    A : (i, j) = (5, 1)
    m=6  T : (i, j) = (9, 7)    A : (i, j) = (2, 3)
    m=7  T : (i, j) = (7, 3)    A : (i, j) = (4, 7)
    m=8  T : (i, j) = (3, 6)    A : (i, j) = (7, 4)
    m=9  T : (i, j) = (6, 5)    A : (i, j) = (1, 5)

 полином:  1 + x^3 + x^10

порождающий элемент(zeta):
0 + x^2 + x^3 + x^5 + x^7 + x^8
-----------------------------------------------------------------

n= 11

 3 тип:

Матрицы: T и A
    m=0 : (i, j)    = (0, 1)    A : (i, j) = (1,1)
    m=1 : (i, j, j) = (1, 8, 0)    A : (i, j) = (4, 3) (1, 0)
    m=2 : (i, j, j) = (2, 6, 8)    A : (i, j) = (7, 5) (5, 3)
    m=3 : (i, j, j) = (3, 5, 4)    A : (i, j) = (9, 6) (10, 7)
    m=4 : (i, j, j) = (4, 9, 3)    A : (i, j) = (6, 2) (1, 8)
    m=5 : (i, j, j) = (5, 7, 3)    A : (i, j) = (9, 4) (2, 8)
    m=6 : (i, j, j) = (6, 2, 9)    A : (i, j) = (4, 9) (8, 2)
    m=7 : (i, j, j) = (7, 5, 10)    A : (i, j) = (2, 6) (8, 1)
    m=8 : (i, j, j) = (8, 2, 1)    A : (i, j) = (6, 9) (7, 10)
    m=9 : (i, j, j) = (9, 4, 6)    A : (i, j) = (5, 7) (3, 5)
    m=10 : (i, j, j) = (10, 7, 10)    A : (i, j) = (3, 4) (0, 1)
 полином:  1 + x^1 + x^22
zeta =
```

```
1 + x^1 + x^2 + x^5 + x^7 + x^9 + x^15 + x^16 + x^19 + x^21
zeta{-1} =
1 + x^2 + x^5 + x^7 + x^9 + x^10 + x^11 + x^14 + x^17
порождающий элемент (alpha = zeta + zeta^{-1}):
0 + x^1 + x^10 + x^11 + x^14 + x^15 + x^16 + x^17 + x^19 + x^21
---------------------------------------------------------------

n= 12

 1 тип:

Матрицы: T и A
 m=0  T : (i) = (6)  A : (i, j) =  (6, 0) (5, 11) (4, 10)
(3, 9) (2, 8) (1, 7) (0, 6) (11, 5) (10, 4) (9, 3) (8, 2) (7, 1)
    m=1  T : (i, j) = (0, 1)    A : (i, j) = (11, 11)
    m=2  T : (i, j) = (1, 4)    A : (i, j) = (9, 8)
    m=3  T : (i, j) = (4, 2)    A : (i, j) = (2, 10)
    m=4  T : (i, j) = (2, 9)    A : (i, j) = (5, 3)
    m=5  T : (i, j) = (9, 5)    A : (i, j) = (4, 7)
    m=6  T : (i, j) = (5, 11)    A : (i, j) = (6, 1)
    m=7  T : (i, j) = (11, 3)    A : (i, j) = (8, 9)
    m=8  T : (i, j) = (3, 8)    A : (i, j) = (7, 4)
    m=9  T : (i, j) = (8, 10)    A : (i, j) = (10, 2)
    m=10  T : (i, j) = (10, 7)    A : (i, j) = (3, 5)
    m=11  T : (i, j) = (7, 6)    A : (i, j) = (1, 6)

 полином:  1 + x^3 + x^12

порождающий элемент (zeta):
1 + x^1 + x^2 + x^3 + x^8 + x^10 + x^11
---------------------------------------------------------------
n= 14

 2 тип:

Матрицы: T и A
    m=0 T : (i, j) = (0, 1)    A : (i, j) = (13, 13)
    m=1 T : (i, j, j) = (1, 5, 0)    A : (i, j) = (10, 9) (1, 0)
    m=2 T : (i, j, j) = (2, 8, 5)    A : (i, j) = (8, 6) (11, 9)
    m=3 T : (i, j, j) = (3, 10, 12)    A : (i, j) = (7, 4) (5, 2)
    m=4 T : (i, j, j) = (4, 7, 13)    A : (i, j) = (11, 7) (5, 1)
    m=5 T : (i, j, j) = (5, 2, 1)    A : (i, j) = (3, 12) (4, 13)
    m=6 T : (i, j, j) = (6, 12, 8)    A : (i, j) = (8, 2) (12, 6)
    m=7 T : (i, j, j) = (7, 4, 11)    A : (i, j) = (3, 10) (10, 3)
    m=8 T : (i, j, j) = (8, 2, 6)    A : (i, j) = (6, 12) (2, 8)
    m=9 T : (i, j, j) = (9, 10, 11)    A : (i, j) = (13, 4) (12, 3)
    m=10 T : (i, j, j) = (10, 9, 3)    A : (i, j) = (1, 5) (7, 11)
    m=11 T : (i, j, j) = (11, 9, 7)    A : (i, j) = (2, 5) (4, 7)
    m=12 T : (i, j, j) = (12, 3, 6)    A : (i, j) = (9, 11) (6, 8)
    m=13 T : (i, j, j) = (13, 13, 4)    A : (i, j) = (0, 1) (9, 10)
 полином:  1 + x^1 + x^28
zeta =
```

```
0 + x^1 + x^2 + x^8 + x^9 + x^11 + x^13 + x^17 + x^19 + x^24
+ x^25 + x^26
zeta{-1} =
1 + x^9 + x^10 + x^11 + x^12 + x^13 + x^14 + x^15 + x^17
+ x^19 + x^20 + x^21 + x^24 + x^25 + x^27
порождающий элемент(alpha = zeta + zeta^{-1}):
1 + x^1 + x^2 + x^8 + x^10 + x^12 + x^14 + x^15 + x^20 + x^21
+ x^26 + x^27
---------------------------------------------------------------

n= 18

 1 тип:

Матрицы: T и A
m=0 T : (i) = (9) A: (i, j) = (9, 0) (8, 17) (7, 16) (6, 15)
(5, 14) (4, 13) (3, 12) (2, 11) (1, 10) (0, 9) (17, 8) (16, 7)
(15, 6) (14, 5) (13, 4) (12, 3) (11, 2) (10, 1)
    m=1  T : (i, j) = (0, 1)    A : (i, j) = (17, 17)
    m=2  T : (i, j) = (1, 13)    A : (i, j) = (6, 5)
    m=3  T : (i, j) = (13, 2)    A : (i, j) = (11, 16)
    m=4  T : (i, j) = (2, 16)    A : (i, j) = (4, 2)
    m=5  T : (i, j) = (16, 14)    A : (i, j) = (2, 4)
    m=6  T : (i, j) = (14, 6)    A : (i, j) = (8, 12)
    m=7  T : (i, j) = (6, 3)    A : (i, j) = (3, 15)
    m=8  T : (i, j) = (3, 8)    A : (i, j) = (13, 10)
    m=9  T : (i, j) = (8, 17)    A : (i, j) = (9, 1)
    m=10  T : (i, j) = (17, 12)    A : (i, j) = (5, 6)
    m=11  T : (i, j) = (12, 15)    A : (i, j) = (15, 3)
    m=12  T : (i, j) = (15, 5)    A : (i, j) = (10, 13)
    m=13  T : (i, j) = (5, 7)    A : (i, j) = (16, 11)
    m=14  T : (i, j) = (7, 11)    A : (i, j) = (14, 7)
    m=15  T : (i, j) = (11, 4)    A : (i, j) = (7, 14)
    m=16  T : (i, j) = (4, 10)    A : (i, j) = (12, 8)
    m=17  T : (i, j) = (10, 9)    A : (i, j) = (1, 9)

 полином:  1 + x^3 + x^18

порождающий элемент (zeta):
1 + x^1 + x^2 + x^5 + x^7 + x^11 + x^12
 2 тип:

Матрицы: T и A
m=0 T : (i, j) = (0, 1)    A : (i, j) = (17, 17)
m=1 T : (i, j, j) = (1, 8, 0) A : (i, j) = (11, 10) (1, 0)
m=2 T : (i, j, j) = (2, 5, 8) A : (i, j) = (15, 13) (12, 10)
m=3 T : (i, j, j) = (3, 16, 14) A : (i, j) = (5, 2) (7, 4)
m=4 T : (i, j, j) = (4, 7, 13) A : (i, j) = (15, 11) (9, 5)
m=5 T : (i, j, j) = (5, 2, 9)  A : (i, j) = (3, 16) (14, 9)
m=6 T : (i, j, j) = (6, 16, 12) A : (i, j) = (8, 2) (12, 6)
m=7 T : (i, j, j) = (7, 17, 4)  A : (i, j) = (8, 1) (3, 14)
m=8 T : (i, j, j) = (8, 1, 2)   A : (i, j) = (7, 17) (6, 16)
```

```
m=9 T : (i, j, j) = (9, 5, 14)  A : (i, j) = (4, 13) (13, 4)
m=10 T : (i, j, j) = (10, 12, 11) A : (i, j) = (16, 6) (17, 7)
m=11 T : (i, j, j) = (11, 15, 10) A : (i, j) = (14, 3) (1, 8)
m=12 T : (i, j, j) = (12, 6, 10)  A : (i, j) = (6, 12) (2, 8)
m=13 T : (i, j, j) = (13, 4, 15)  A : (i, j) = (9, 14) (16, 3)
m=14 T : (i, j, j) = (14, 9, 3)   A : (i, j) = (5, 9) (11, 15)
m=15 T : (i, j, j) = (15, 11, 13) A : (i, j) = (4, 7) (2, 5)
m=16 T : (i, j, j) = (16, 6, 3)   A : (i, j) = (10, 12) (13, 15)
m=17 T : (i, j, j) = (17, 17, 7)  A : (i, j) = (0, 1) (10, 11)
 полином:  1 + x^9 + x^36
zeta =
1 + x^1 + x^5 + x^6 + x^8 + x^11 + x^16 + x^17 + x^20 + x^21 +
x^23 + x^24 + x^26 + x^29 + x^30 + x^31
zeta{-1} =
1 + x^1 + x^4 + x^5 + x^13 + x^14 + x^16 + x^17 + x^19 + x^24
+ x^27 + x^30
порождающий элемент (alpha = zeta + zeta^{-1}):
0 + x^4 + x^6 + x^8 + x^11 + x^13 + x^14 + x^19 + x^20 + x^21
+ x^23 + x^26 + x^27 + x^29 + x^31
----------------------------------------------------------------
n= 23

 3 тип:

Матрицы: T и A
m=0 : (i, j)    = (0, 1)    A : (i, j) = (1,1)
m=1 : (i, j, j) = (1, 19, 0)  A : (i, j) = (5, 4) (1, 0)
m=2 : (i, j, j) = (2, 9, 19)  A : (i, j) = (16, 14) (6, 4)
m=3 : (i, j, j) = (3, 15, 12) A : (i, j) = (11, 8) (14, 11)
m=4 : (i, j, j) = (4, 6, 5)   A : (i, j) = (21, 17) (22, 18)
m=5 : (i, j, j) = (5, 13, 4)  A : (i, j) = (15, 10) (1, 19)
m=6 : (i, j, j) = (6, 16, 4)  A : (i, j) = (13, 7) (2, 19)
m=7 : (i, j, j) = (7, 21, 13) A : (i, j) = (9, 2) (17, 10)
m=8 : (i, j, j) = (8, 18, 11) A : (i, j) = (13, 5) (20, 12)
m=9 : (i, j, j) = (9, 2, 20)  A : (i, j) = (7, 21) (12, 3)
m=10 : (i, j, j) = (10, 15, 17)  A : (i, j) = (18, 8) (16, 6)
m=11 : (i, j, j) = (11, 14, 8)   A : (i, j) = (20, 9) (3, 15)
m=12 : (i, j, j) = (12, 3, 20)   A : (i, j) = (9, 20) (15, 3)
m=13 : (i, j, j) = (13, 5, 7)    A : (i, j) = (8, 18) (6, 16)
m=14 : (i, j, j) = (14, 16, 11)  A : (i, j) = (21, 7) (3, 12)
m=15 : (i, j, j) = (15, 10, 3)   A : (i, j) = (5, 13) (12, 20)
m=16 : (i, j, j) = (16, 14, 6)   A : (i, j) = (2, 9) (10, 17)
m=17 : (i, j, j) = (17, 10, 21)  A : (i, j) = (7, 13) (19, 2)
m=18 : (i, j, j) = (18, 8, 22)   A : (i, j) = (10, 15) (19, 1)
m=19 : (i, j, j) = (19, 2, 1)    A : (i, j) = (17, 21) (18, 22)
m=20 : (i, j, j) = (20, 12, 9)   A : (i, j) = (8, 11) (11, 14)
m=21 : (i, j, j) = (21, 7, 17)   A : (i, j) = (14, 16) (4, 6)
m=22 : (i, j, j) = (22, 18, 22)  A : (i, j) = (4, 5) (0, 1)
 полином:  1 + x^1 + x^46
zeta =
1 + x^3 + x^6 + x^11 + x^12 + x^13 + x^14 + x^15 + x^21 + x^24
+ x^26 + x^27 + x^29 + x^31 + x^39 + x^42 + x^44 + x^45
```

```
zeta{-1} =
1 + x^1 + x^5 + x^6 + x^9 + x^11 + x^12 + x^13 + x^15 + x^19 +
x^21 + x^23 + x^27 + x^30 + x^31 + x^32 + x^33 + x^34 + x^37 +
x^38 + x^43 + x^44 + x^45
порождающий элемент (alpha = zeta + zeta^{-1}):
0 + x^1 + x^3 + x^5 + x^9 + x^14 + x^19 + x^23 + x^24 + x^26
+ x^29 + x^30 + x^32 + x^33 + x^34 + x^37 + x^38 + x^39 +
x^42 + x^43
-------------------------------------------------------------
n= 26

 2 тип:

Матрицы: T и A
m=0 T : (i, j) = (0, 1)  A : (i, j) = (25, 25)
m=1 T : (i, j, j) = (1, 17, 0)  A : (i, j) = (10, 9) (1, 0)
m=2 T : (i, j, j) = (2, 21, 17) A : (i, j) = (7, 5) (11, 9)
m=3 T : (i, j, j) = (3, 8, 14) A : (i, j) = (21, 18) (15, 12)
m=4 T : (i, j, j) = (4, 10, 12) A : (i, j) = (20, 16) (18, 14)
m=5 T : (i, j, j) = (5, 23, 7)  A : (i, j) = (8, 3) (24, 19)
m=6 T : (i, j, j) = (6, 19, 22) A : (i, j) = (13, 7) (10, 4)
m=7 T : (i, j, j) = (7, 13, 5)  A : (i, j) = (20, 13) (2, 21)
m=8 T : (i, j, j) = (8, 3, 22)  A : (i, j) = (5, 23) (12, 4)
m=9 T : (i, j, j) = (9, 10, 11) A : (i, j) = (25, 16) (24, 15)
m=10 T : (i, j, j) = (10, 9, 4) A : (i, j) = (1, 17) (6, 22)
m=11 T : (i, j, j) = (11, 9, 23) A : (i, j) = (2, 17) (14, 3)
m=12 T : (i, j, j) = (12, 4, 15) A : (i, j) = (8, 22) (23, 11)
m=13 T : (i, j, j) = (13, 7, 20) A : (i, j) = (6, 19) (19, 6)
m=14 T : (i, j, j) = (14, 3, 18) A : (i, j) = (11, 23) (22, 8)
m=15 T : (i, j, j) = (15, 12, 24) A : (i, j) = (3, 14) (17, 2)
m=16 T : (i, j, j) = (16, 20, 25) A : (i, j) = (22, 6) (17, 1)
m=17 T : (i, j, j) = (17, 2, 1)   A : (i, j) = (15, 24) (16, 25)
m=18 T : (i, j, j) = (18, 14, 21) A : (i, j) = (4, 12) (23, 5)
m=19 T : (i, j, j) = (19, 24, 6)  A : (i, j) = (21, 2) (13, 20)
m=20 T : (i, j, j) = (20, 16, 13) A : (i, j) = (4, 10) (7, 13)
m=21 T : (i, j, j) = (21, 2, 18)  A : (i, j) = (19, 24) (3, 8)
m=22 T : (i, j, j) = (22, 8, 6)  A : (i, j) = (14, 18) (16, 20)
m=23 T : (i, j, j) = (23, 11, 5) A : (i, j) = (12, 15) (18, 21)
m=24 T : (i, j, j) = (24, 15, 19) A : (i, j) = (9, 11) (5, 7)
m=25 T : (i, j, j) = (25, 25, 16) A : (i, j) = (0, 1) (9, 10)
 полином:  1 + x^3 + x^52
zeta =
0 + x^1 + x^2 + x^3 + x^8 + x^9 + x^10 + x^12 + x^13 + x^15 +
x^24 + x^31 + x^32 + x^34 + x^36 + x^37 + x^38 + x^40 + x^43 +
x^45 + x^48 + x^49
zeta{-1} =
0 + x^2 + x^3 + x^4 + x^5 + x^8 + x^9 + x^10 + x^11 + x^14 +
x^17 + x^23 + x^24 + x^25 + x^31 + x^32 + x^33 + x^36 + x^39 +
x^40 + x^41 + x^42 + x^43 + x^44 + x^46 + x^47 + x^48 + x^49
порождающий элемент (alpha = zeta + zeta^{-1}):
0 + x^1 + x^4 + x^5 + x^11 + x^12 + x^13 + x^14 + x^15 + x^17 +
x^23 + x^25 + x^33 + x^34 + x^37 + x^38 + x^39 + x^41 + x^42 +
```

```
x^44 + x^45 + x^46 + x^47
------------------------------------------------------------------

n= 28

 1 тип:

Матрицы: T и A
m=0 T : (i) = (14)  A : (i, j) =  (14, 0) (13, 27) (12, 26)
(11, 25) (10, 24) (9, 23) (8, 22) (7, 21) (6, 20) (5, 19)
(4, 18) (3, 17) (2, 16) (1, 15) (0, 14) (27, 13) (26, 12)
(25, 11) (24, 10) (23, 9) (22, 8) (21, 7) (20, 6) (19, 5)
(18, 4) (17,3) (16, 2) (15, 1)
    m=1  T : (i, j) = (0, 1)    A : (i, j) = (27, 27)
    m=2  T : (i, j) = (1, 5)    A : (i, j) = (24, 23)
    m=3  T : (i, j) = (5, 2)    A : (i, j) = (3, 26)
    m=4  T : (i, j) = (2, 22)    A : (i, j) = (8, 6)
    m=5  T : (i, j) = (22, 6)    A : (i, j) = (16, 22)
    m=6  T : (i, j) = (6, 12)    A : (i, j) = (22, 16)
    m=7  T : (i, j) = (12, 3)    A : (i, j) = (9, 25)
    m=8  T : (i, j) = (3, 10)    A : (i, j) = (21, 18)
    m=9  T : (i, j) = (10, 23)    A : (i, j) = (15, 5)
    m=10  T : (i, j) = (23, 25)    A : (i, j) = (26, 3)
    m=11  T : (i, j) = (25, 7)    A : (i, j) = (18, 21)
    m=12  T : (i, j) = (7, 18)    A : (i, j) = (17, 10)
    m=13  T : (i, j) = (18, 13)    A : (i, j) = (5, 15)
    m=14  T : (i, j) = (13, 27)    A : (i, j) = (14, 1)
    m=15  T : (i, j) = (27, 4)    A : (i, j) = (23, 24)
    m=16  T : (i, j) = (4, 21)    A : (i, j) = (11, 7)
    m=17  T : (i, j) = (21, 11)    A : (i, j) = (10, 17)
    m=18  T : (i, j) = (11, 9)    A : (i, j) = (2, 19)
    m=19  T : (i, j) = (9, 24)    A : (i, j) = (13, 4)
    m=20  T : (i, j) = (24, 17)    A : (i, j) = (7, 11)
    m=21  T : (i, j) = (17, 26)    A : (i, j) = (19, 2)
    m=22  T : (i, j) = (26, 20)    A : (i, j) = (6, 8)
    m=23  T : (i, j) = (20, 8)    A : (i, j) = (12, 20)
    m=24  T : (i, j) = (8, 16)    A : (i, j) = (20, 12)
    m=25  T : (i, j) = (16, 19)    A : (i, j) = (25, 9)
    m=26  T : (i, j) = (19, 15)    A : (i, j) = (4, 13)
    m=27  T : (i, j) = (15, 14)    A : (i, j) = (1, 14)

полином:  1 + x^1 + x^28

порождающий элемент (zeta):
1 + x^1 + x^3 + x^7 + x^9 + x^10 + x^11 + x^12 + x^15 + x^16
+ x^20 + x^21 + x^22 + x^24 + x^26
------------------------------------------------------------------

n= 29

 2 тип:
```

```
Матрицы: T и A
m=0 T : (i, j) = (0, 1)    A : (i, j) = (28, 28)
m=1 T : (i, j, j) = (1, 21, 0)    A : (i, j) = (9, 8) (1, 0)
m=2 T : (i, j, j) = (2, 6, 21)    A : (i, j) = (25, 23) (10, 8)
m=3 T : (i, j, j) = (3, 13, 18)    A : (i, j) = (19, 16) (14, 11)
m=4 T : (i, j, j) = (4, 11, 27)    A : (i, j) = (22, 18) (6, 2)
m=5 T : (i, j, j) = (5, 17, 20)    A : (i, j) = (17, 12) (14, 9)
m=6 T : (i, j, j) = (6, 22, 2)    A : (i, j) = (13, 7) (4, 27)
m=7 T : (i, j, j) = (7, 25, 13)    A : (i, j) = (11, 4) (23, 16)
m=8 T : (i, j, j) = (8, 10, 9)    A : (i, j) = (27, 19) (28, 20)
m=9 T : (i, j, j) = (9, 14, 8)    A : (i, j) = (24, 15) (1, 21)
m=10 T : (i, j, j) = (10, 26, 8)    A : (i, j) = (13, 3) (2, 21)
m=11 T : (i, j, j) = (11, 4, 14)    A : (i, j) = (7, 25) (26, 15)
m=12 T : (i, j, j) = (12, 17, 24)    A : (i, j) = (24, 12) (17, 5)
m=13 T : (i, j, j) = (13, 3, 7)    A : (i, j) = (10, 26) (6, 22)
m=14 T : (i, j, j) = (14, 11, 9)    A : (i, j) = (3, 18) (5, 20)
m=15 T : (i, j, j) = (15, 24, 26)    A : (i, j) = (20, 5) (18, 3)
m=16 T : (i, j, j) = (16, 23, 19)    A : (i, j) = (22, 6) (26, 10)
m=17 T : (i, j, j) = (17, 12, 5)    A : (i, j) = (5, 17) (12, 24)
m=18 T : (i, j, j) = (18, 3, 22)    A : (i, j) = (15, 26) (25, 7)
m=19 T : (i, j, j) = (19, 27, 16)    A : (i, j) = (21, 2) (3, 13)
m=20 T : (i, j, j) = (20, 28, 5)    A : (i, j) = (21, 1) (15, 24)
m=21 T : (i, j, j) = (21, 1, 2)    A : (i, j) = (20, 28) (19, 27)
m=22 T : (i, j, j) = (22, 6, 18)    A : (i, j) = (16, 23) (4, 11)
m=23 T : (i, j, j) = (23, 25, 16)    A : (i, j) = (27, 4) (7, 13)
m=24 T : (i, j, j) = (24, 15, 12)    A : (i, j) = (9, 14) (12, 17)
m=25 T : (i, j, j) = (25, 23, 7)    A : (i, j) = (2, 6) (18, 22)
m=26 T : (i, j, j) = (26, 15, 10)    A : (i, j) = (11, 14) (16, 19)
m=27 T : (i, j, j) = (27, 19, 4)    A : (i, j) = (8, 10) (23, 25)
m=28 T : (i, j, j) = (28, 28, 20)    A : (i, j) = (0, 1) (8, 9)
 полином:  1 + x^19 + x^58
zeta =
1 + x^1 + x^3 + x^5 + x^6 + x^7 + x^8 + x^10 + x^11 + x^12 + x^15
+ x^18 + x^24 + x^26 + x^29 + x^31 + x^33 + x^34 + x^36 + x^37
+ x^38 + x^39 + x^41 + x^43 + x^45 + x^51 + x^54 + x^55 + x^57
zeta{-1} =
0 + x^1 + x^4 + x^6 + x^7 + x^8 + x^10 + x^11 + x^12 + x^13 +
x^14 + x^15 + x^16 + x^17 + x^18 + x^19 + x^21 + x^22 + x^23 +
x^26 + x^27 + x^32 + x^34 + x^35 + x^36 + x^37 + x^38 + x^41 +
x^42 + x^43 + x^44 + x^45 + x^48 + x^52 + x^53 + x^54 + x^56
порождающий элемент (alpha = zeta + zeta^{-1}):
1 + x^3 + x^4 + x^5 + x^13 + x^14 + x^16 + x^17 + x^19 + x^21
+ x^22 + x^23 + x^24 + x^27 + x^29 + x^31 + x^32 + x^33 + x^35
+ x^39 + x^42 + x^44 + x^48 + x^51 + x^52 + x^53 + x^55 + x^56
+ x^57
-----------------------------------------------------------------

n= 30

 2 тип:

Матрицы: T и A
```

```
m=0 T : (i, j) = (0, 1)    A : (i, j) = (29, 29)
m=1 T : (i, j, j) = (1, 6, 0)    A : (i, j) = (25, 24) (1, 0)
m=2 T : (i, j, j) = (2, 22, 6)    A : (i, j) = (10, 8) (26, 24)
m=3 T : (i, j, j) = (3, 12, 19)    A : (i, j) = (21, 18) (14, 11)
m=4 T : (i, j, j) = (4, 17, 28)    A : (i, j) = (17, 13) (6, 2)
m=5 T : (i, j, j) = (5, 21, 29)    A : (i, j) = (14, 9) (6, 1)
m=6 T : (i, j, j) = (6, 2, 1)    A : (i, j) = (4, 28) (5, 29)
m=7 T : (i, j, j) = (7, 19, 22)    A : (i, j) = (18, 11) (15, 8)
m=8 T : (i, j, j) = (8, 10, 15)    A : (i, j) = (28, 20) (23, 15)
m=9 T : (i, j, j) = (9, 14, 27)    A : (i, j) = (25, 16) (12, 3)
m=10 T : (i, j, j) = (10, 8, 20)    A : (i, j) = (2, 22) (20, 10)
m=11 T : (i, j, j) = (11, 14, 18)    A : (i, j) = (27, 16) (23, 12)
m=12 T : (i, j, j) = (12, 23, 3)    A : (i, j) = (19, 7) (9, 27)
m=13 T : (i, j, j) = (13, 26, 17)    A : (i, j) = (17, 4) (26, 13)
m=14 T : (i, j, j) = (14, 9, 11)    A : (i, j) = (5, 21) (3, 19)
m=15 T : (i, j, j) = (15, 8, 23)    A : (i, j) = (7, 22) (22, 7)
m=16 T : (i, j, j) = (16, 27, 25)    A : (i, j) = (19, 3) (21, 5)
m=17 T : (i, j, j) = (17, 4, 13)    A : (i, j) = (13, 26) (4, 17)
m=18 T : (i, j, j) = (18, 21, 11)    A : (i, j) = (27, 9) (7, 19)
m=19 T : (i, j, j) = (19, 7, 3)    A : (i, j) = (12, 23) (16, 27)
m=20 T : (i, j, j) = (20, 10, 28)    A : (i, j) = (10, 20) (22, 2)
m=21 T : (i, j, j) = (21, 18, 5)    A : (i, j) = (3, 12) (16, 25)
m=22 T : (i, j, j) = (22, 7, 2)    A : (i, j) = (15, 23) (20, 28)
m=23 T : (i, j, j) = (23, 15, 12)    A : (i, j) = (8, 15) (11, 18)
m=24 T : (i, j, j) = (24, 25, 26)    A : (i, j) = (29, 5) (28, 4)
m=25 T : (i, j, j) = (25, 24, 16)    A : (i, j) = (1, 6) (9, 14)
m=26 T : (i, j, j) = (26, 24, 13)    A : (i, j) = (2, 6) (13, 17)
m=27 T : (i, j, j) = (27, 16, 9)    A : (i, j) = (11, 14) (18, 21)
m=28 T : (i, j, j) = (28, 4, 20)    A : (i, j) = (24, 26) (8, 10)
m=29 T : (i, j, j) = (29, 29, 5)    A : (i, j) = (0, 1) (24, 25)
 полином:  1 + x^1 + x^60
zeta =
0 + x^1 + x^2 + x^3 + x^4 + x^6 + x^8 + x^10 + x^11 + x^12 + x^14 +
x^15 + x^17 + x^19 + x^21 + x^23 + x^26 + x^28 + x^29 + x^32 + x^33 +
x^36 + x^38 + x^42 + x^44 + x^46 + x^47 + x^52 + x^54 + x^55 +
x^57 + x^59
zeta{-1} =
0 + x^1 + x^2 + x^4 + x^7 + x^9 + x^10 + x^12 + x^14 + x^15 + x^17 +
x^19 + x^21 + x^22 + x^24 + x^25 + x^26 + x^29 + x^32 + x^33 + x^37 +
x^39 + x^43 + x^45 + x^46 + x^47 + x^49 + x^50 + x^52 + x^53 + x^54 +
x^56 + x^57 + x^58
порождающий элемент (alpha = zeta + zeta^{-1}):
0 + x^3 + x^6 + x^7 + x^8 + x^9 + x^11 + x^22 + x^23 + x^24 + x^25 +
x^28 + x^36 + x^37 + x^38 + x^39 + x^42 + x^43 + x^44 + x^45 + x^49 +
x^50 + x^53 + x^55 + x^56 + x^58 + x^59
```

2.3.3 Алгоритм генерации оптимальных нормальных базисов первого типа и доказательство их оптимальности

Для поиска значений n, при которых существует базис первого типа, надо сгенерировать таблицу простых чисел от 1 до заданной границы поиска N с помощью решета Эратосфена и для очередного простого p проверить, что

$$q^{(p-1)/\delta} \bmod p \neq 1$$

для каждого простого делителя δ числа $p-1$. Факторизация этого числа тривиальным методом требует времени $O(\sqrt{p})$ (с учетом уже построенной таблицы простых), а проверка всех $O(\log p)$ указанных неравенств — времени $O(\log^2 p)$. Общее время работы этого подалгоритма — $O(N^{3/2})$.

Далее для каждого найденного $n = p-1$ ищем в поле $GF(q^n)$ примитивный корень ζ p-й степени из 1. Так как в силу простоты p все неединичные корни указанной степени степени являются примитивными, а $q^n - 1$ кратно p согласно выбору n, то для любого ненулевого элемента β из поля $GF(q^n)$ его степень $\beta^{(q^n-1)/p}$, в случае если она не равна 1, будет искомым корнем, так как $\beta^{q^n-1} = 1$ по малой теореме Ферма. Вероятность того, что для случайного β будет справедливо равенство $\beta^{(q^n-1)/p} = 1$, равна $1/p$, так как количество корней $(q^n-1)/p$ степени из 1 равно $(q^n-1)/p$, значит нужный нам корень после, например 5 попыток, мы получим с почти единичной вероятностью за время $O(n \log q)^3$, если использовать стандартный быстрый алгоритм возведения в степень и алгоритм сложности $O(n \log q)^2$ для умножения в поле $GF(q^n)$. В качестве представления элементов поля можно взять многочлены степени $n-1$ с коэффициентами из поля $GF(q)$, а операции проводить по модулю неприводимого полинома, желательно «малочлена».

Приведем доказательство оптимальности базисов первого типа. Заметим, что система $\{\zeta, \ldots, \zeta^n\}$ совпадает с точностью до перестановки с нормальным базисом

$$\{\zeta, \zeta^q, \zeta^{q^2}, \ldots, \zeta^{q^{n-1}}\},$$

так как последовательность степеней $1, q, q^2, \ldots, q^{n-1}$, вычисленных по модулю p, совпадает с некоторой перестановкой $\pi(1), \ldots, \pi(n)$ множества чисел $\{1, \ldots, n\}$ в силу того, что q — примитивный корень по модулю p, значит равенство q^i и q^j при $0 \geq i < j < n$ по модулю p невозможно, так как иначе бы q^{j-i} было бы равно 1 по модулю p, что противоречило бы примитивности q по этому модулю.

Линейная независимость системы $\{\zeta, \ldots, \zeta^n\}$ вытекает из того, что ζ не может быть корнем никакого многочлена над полем $GF(q)$ степени меньшей n, так как ζ является корнем неприводимого над полем $GF(q)$ многочлена

$$1 + x + \ldots + x^n = \frac{x^p - 1}{x - 1},$$

который неприводим в силу того, что если он имеет собственный делитель $f(x)$, то его корнем будет один из корней ζ^{q^i} многочлена $1+x+\ldots+x^n$, а значит его корнями будут все элементы вида $\alpha^{q^i}, i = 1, 2, \ldots$, где $\alpha = \zeta^{q^k}$, а значит и все элементы $\zeta^{q^{i+k}}, i = 1, 2, \ldots$, где сумма $i + k$ вычисляется по модулю n, так как

$$q^n = 1 \bmod p, \zeta^{q^n} = \zeta,$$

и поэтому корнями многочлена $f(x)$ являются все n корней многочлена $1 + x + \ldots + x^n$, что невозможно.

Вычислим теперь матрицу T. Для этого надо найти разложение

$$\zeta\zeta^{q^i} = \sum_{j=0}^{n-1} t_{i,j}\zeta^{q^j}.$$

Согласно равенствам $\zeta\zeta^m = \zeta^{m+1}, m = 1, \ldots, n$ при $m < n$ сумма

$$\sum_{j=0}^{n-1} t_{i,j}\zeta^{q^j}$$

будет равна ζ^{m+1} только если при i таком, что $q^i = m \bmod p$, (то есть при $i = \pi^{-1}(m)$) равенство $t_{i,j} = 1$ будет справедливо лишь при j таком, что $q^j = m + 1 \bmod p$, то есть при

$$j = \pi^{-1}(m+1) = \pi^{-1}(\pi(i)+1),$$

а остальные коэффициенты $t_{i,j}$ будут равны нулю.

Если же $m = n$, то

$$\zeta\zeta^m = \zeta^{n+1} = \zeta^p = 1 = \sum_{k=1}^{n} \zeta^k = \sum_{j=0}^{n-1} \zeta^{q^j},$$

значит при $i = \pi^{-1}(n)$ все коэффициенты $t_{i,j} = 1$. В итоге общее число ненулевых коэффициентов равно $2n-1$, значит построенный базис оптимальный нормальный.

Приведем в заключение в явном виде

Алгоритм построения оптимальных нормальных базисов первого типа и таблицы умножения в них при $q = 2$

Пусть надо построить оптимальный нормальный базис в поле $GF(2^n)$.

1. Проверяем, является ли $p = n + 1$ простым числом.

2. Если да, то проверяем является ли 2 примитивным корнем по модулю p. Для этого проверяем, что

$$2^{(p-1)/\delta} \bmod p \neq 1$$

для каждого простого делителя δ числа $p-1$. Если это выполнено, то 2 является примитивным корнем по модулю p.

3. Ищем в поле $GF(2^n)$ примитивный корень ζ p-й степени из 1. Для этого случайным образом генерируем ненулевой элемент β из поля $GF(2^n)$ и вычисляем его степень $\beta^{(q^n-1)/p}$. Если эта степень не равна 1, то полагаем ζ равным этой степени, в противном случае генерируем новый элемент β, и так далее, пока не получим $\beta^{(q^n-1)/p} \neq 1$, после чего полагаем $\zeta = \beta^{(q^n-1)/p}$.

Заметим однако, что для вычисления нужной нам матрицы T пункт 3 не нужен.

4.Вычисляем матрицу $T = (t_{i,j})$. Выбираем очередной индекс i, $0 \leq i < n$. Сначала находим $m = 2^i \bmod p$. Если $m < n$, то находим $0 \leq j < n$, такое что $2^j = m + 1 \bmod p$, и полагаем $t_{i,j} = 1$, а при $k \neq j$ $t_{i,k} = 0$. Если же $m = n$, то полагаем $t_{i,j} = 1$ при всех j, $0 \leq j < n$.

5. Вычисление матрицы A осуществляется с помощью равенств $a_{i,j} = t_{i-j,-j}$, где $i - j$ и $-j$ вычисляются по модулю n. Эта матрица является выходом алгоритма, так как именно она используется в алгоритме умножения Massey-Omura.

Упражнение 2.3.2 Докажите, что $t_{i,j} = 1$ если и только если $1 + 2^i = 2^j \bmod n$ или $i = n/2$.

Упражнение 2.3.3 Докажите, что $a_{i,j} = 1$ если и только если $2^j + 2^i = 1 \bmod n$ или $i - j = n/2 \bmod n$.

2.3.4 Алгоритм генерации оптимальных нормальных базисов второго типа и доказательство их оптимальности

Базис второго типа возникает, когда $2n+1=p$ — простое число, а условие на q такое же, как в первом случае, где роль числа n играет число $2n$. Элемент ζ тогда будет примитивным корнем степени p из 1 в поле $GF(q^{2n})$, но в качестве порождающего элемента оптимального нормального базиса нужно взять $\alpha=\zeta+\zeta^{-1}$. Так как

$$q^n=-1 \bmod p,$$

то

$$\alpha^{q^n}=\zeta^{q^n}+\zeta^{-q^n}=\zeta+\zeta^{-1}=\alpha,$$

значит

$$\alpha^{q^n}=\alpha,$$

поэтому α принадлежит подполю $GF(q^n)$ поля $GF(q^{2n})$. Система

$$\{\alpha,\alpha^q,\alpha^{q^2},\ldots,\alpha^{q^{n-1}}\}$$

линейно независима, так как

$$\alpha^{q^k}=\zeta^{q^k}+\zeta^{-q^k}=\zeta^{q^k}+\zeta^{q^{k+n}}$$

в силу равенства

$$q^{k+n}=-q^k \bmod p,$$

а система

$$\{\zeta,\zeta^q,\zeta^{q^2},\ldots,\zeta^{q^{2n-1}}\}$$

линейно независима. Можно проверить, что при $n>k>0$

$$\alpha\alpha^{q^k}=(\zeta+\zeta^{-1})(\zeta^{q^k}+\zeta^{-q^k})=\zeta^{1+q^k}+\zeta^{-1-q^k}+\zeta^{1-q^k}+\zeta^{-1+q^k}=$$

$$=\alpha^{q^s}+\alpha^{q^t},$$

значит соответствующая этому базису матрица T при $q=2^l$ содержит $2n-1$ единицу, потому что при $k=0$ разложение произведения $\alpha\alpha^{q^0}$ по базису состоит из одного слагаемого, ведь

$$\alpha^2=(\zeta+\zeta^{-1})^2=\zeta^2+\zeta^{-2}=\alpha^{q^s}.$$

Для выполнения указанной проверки заметим, что согласно определению последовательность $q^k \bmod p, k=0,\ldots,2n-1$ является перестановкой перестановкой $\pi(1),\ldots,\pi(2n)$ множества чисел $\{1,\ldots,2n\}$ в силу того, что q — примитивный корень по модулю $p=2n+1$, причем в силу соотношения

$$q^{k+n}=-q^k \bmod p, k=0,\ldots,n-1$$

справедливо равенство

$$\pi(k)+\pi(k+n)=p, k=0,\ldots,n-1,$$

а так как все неупорядоченные пары

$$(1+q^k \bmod p, -1-q^k \bmod p)$$

при любом $k = 0, \ldots, n-1$, отличны от пары $(0,0)$, то их последовательность является состоит из пар

$$(\pi(k), \pi(k+n)), k = 0, \ldots, n-1$$

, т.е. из неупорядоченных пар вида $(u, p-u), 0 < u \geq n$, значит существует такое отображение $(\sigma(1), \ldots, \sigma(n))$ множества чисел $1, \ldots, n$, в себя, что

$$(1 + q^k \bmod p, -1 - q^k \bmod p) = (\pi(\sigma(k+1)), p - \pi(\sigma(k+1))),$$

и значит

$$\zeta^{1+q^k} + \zeta^{-1-q^k} = \zeta^{q^{\sigma(k+1)}} + \zeta^{-q^{\sigma(k+1)}} = \alpha^{q^{\sigma(k+1)}}.$$

Аналогично определяется отображение $(\mu(1), \ldots, \mu(n))$ множества $1, \ldots, n$, в себя такое, что

$$\zeta^{1-q^k} + \zeta^{-1+q^k} = \zeta^{q^{\mu(k)}} + \zeta^{-q^{\mu(k)}} = \alpha^{q^{\mu(k)}}$$

при любом $n \geq k > 0$.

Так как

$$1 - q^k \neq 1 + q^k, 1 - q^k \neq -1 - q^k$$

по модулю p, то при любом $n > k > 0$ справедливо неравенство

$$\mu(k) \neq \sigma(k+1).$$

Вспоминая определение матрицы T посредством равенств

$$\alpha\alpha^{q^i} = \sum_{j=0}^{n-1} t_{i,j}\alpha^{q^j},$$

и сравнивая их с полученными равенствами

$$\alpha\alpha^{q^i} = \alpha^{q^{\sigma(i+1)}} + \alpha^{q^{\mu(i)}}, i > 0,$$

$$\alpha\alpha^{q^0} = \zeta^{1+q^0} + \zeta^{-1-q^0} + \zeta^{1-q^0} + \zeta^{-1+q^0} =$$

$$= \zeta^{1+q^0} + \zeta^{-1-q^0} + \zeta^0 + \zeta^0 = \zeta^{1+q^0} + \zeta^{-1-q^0} == \alpha^{q^{\sigma(1)}},$$

имеем

$$t_{i,j} = \delta_{\sigma(i+1),j} + \delta_{\mu(i),j}, i \neq 0,$$

$$t_{0,j} = \delta_{\sigma(1),j},$$

где $\delta_{k,s}$ — дельта-символ Кронекера, равный 1 при $k = s$ и нулю в противном случае.

Отметим, что в случае нечетного q указанный базис не будет, строго говоря, оптимальным, но будет иметь линейную сложность. Действительно, тогда

$$\alpha^2 = (\zeta + \zeta^{-1})^2 = \zeta^2 + \zeta^{-2} + 2 = \alpha^{q^s} + 2,$$

а элемент 2 из поля $GF(q)$ представим в виде

$$c\sum_{k=0}^{n-1} \alpha^{q^k}$$

при ненулевом c из поля $GF(q)$. Приведем в заключение в явном виде

Алгоритм построения оптимальных нормальных базисов второго типа и таблицы умножения в них при $q = 2$

Пусть надо построить оптимальный нормальный базис в поле $GF(2^n)$.

1. Проверяем, является ли $p = 2n + 1$ простым числом.
2. Если да, то проверяем является ли 2 примитивным корнем по модулю p.
3.Если да, то вычисляем матрицу $T = (t_{i,j})$.

Для этого сначала строим массив

$$\pi(k) = 2^k \bmod q, k = 0, \ldots, 2n-1$$

Потом строим массив $(\sigma(1), \ldots, \sigma(n))$ такой, что $1 \leq \sigma(k) \leq n$ и

$$(1 + 2^k \bmod p, -1 - 2^k \bmod p) = (\pi(\sigma(k+1)), p - \pi(\sigma(k+1))), k = 0, \ldots, n-1.$$

Потом строим массив $(\mu(1), \ldots, \mu(n))$, такой что $1 \leq \mu(k) \leq n$ и

$$(1 - 2^k \bmod p, -1 + 2^k \bmod p) = (\pi(\mu(k)), p - \pi(\mu(k))), k = 1, \ldots, n$$

И наконец вычисляем для всех $0 < i < n$ и всех j, $0 \leq j < n$

$$t_{i,j} = \delta_{\sigma(i+1),j} + \delta_{\mu(i),j},$$

$$t_{0,j} = \delta_{\sigma(1),j},$$

где $\delta_{k,s}$ — дельта-символ Кронекера.

2.3.5 Алгоритм генерации оптимальных нормальных базисов третьего типа и доказательство их оптимальности

Базис третьего типа возникает, когда n нечетно, $2n + 1 = p$ — простое число, а условие на q заменяется на то, что $q^n = 1 \bmod p$, и при любом $0 < k < n$ $q^k \neq 1 \bmod p$ (другими словами, число q имеет по модулю p порядок n, а не $2n$, как во втором случае, и тогда автоматически существует такое r, что $q = r^2 \bmod p$, т.е. q — квадратичный вычет по модулю p , и поэтому все его степени $q^k \bmod p, k = 0, \ldots, n-1$ образуют перестановку множества всех квадратичных вычетов по модулю p, так как их ровно n штук).

А так как p равно 3 по модулю 4, то -1 является квадратичным невычетом по модулю p, так как в противном случае существовало бы такое число r, что $-1 = r^2 \bmod p$, и тогда получилось бы противоречие с малой теоремой Ферма:

$$r^{p-1} = (r^2)^{(p-1)/2} = (-1)^{(p-1)/2} = -1 \bmod p.$$

Поэтому из того, что произведение вычета на невычет является невычетом, следует, что последовательность $-q^k \bmod p, k = 0, \ldots, n-1$ образуют перестановку множества всех квадратичных невычетов по модулю p,

Как и в случае базиса второго типа в качестве элемента ζ берется примитивный корень степени p из 1 в поле $GF(q^{2n})$, а в качестве порождающего элемента оптимального нормального базиса — элемент $\alpha = \zeta + \zeta^{-1}$.

Однако доказательство линейной независимости системы

$$\{\alpha, \alpha^q, \alpha^{q^2}, \ldots, \alpha^{q^{n-1}}\}$$

отличается от второго случая. Допустим противное, то есть

$$\sum_{j=0}^{n-1} a_j \alpha^{q^j} = 0$$

для некоторого ненулевого вектора (a_j) с координатами из поля $GF(q)$. Подставляя равенства

$$\alpha^{q^i} = \zeta^{q^i} + \zeta^{-q^i},$$

получаем, что

$$\sum_{j=0}^{n-1} a_j(\zeta^{q^j} + \zeta^{-q^j}) = 0.$$

Последнее равенство можно переписать в виде

$$\sum_{i=0}^{2n-1} b_i\zeta^i = 0,$$

если определить b_i как a_j, где $i = \pm q^j \bmod p$ (как было указано выше, последнее условие определяет число j однозначно; знак плюс соответствует случаю, когда число i — вычет, а знак минус — случаю, когда i является квадратичным невычетом по модулю p), значит многочлен

$$f(x) = \sum_{i=0}^{2n-1} b_i x^i$$

имеет корни ζ и ζ^{-1} в поле $GF(q^{2n})$, а так как эти элементы имеют минимальные многочлены степени n каждый, причем они взаимно просты (в указанном поле корнями первого из них являются $\zeta^{q^i}, i = 0, \ldots, n-1$, а корнями второго — элементы $\zeta^{-q^i}, i = 0, \ldots, n-1$), то многочлен $f(x)$ над полем $GF(q)$ должен делится на их произведение, что невозможно, так как его степень меньше $2n$.

Проверка оптимальности построенного базиса почти ничем не отличается от второго случая. Достаточно проверить, что при $k < n$

$$1 + q^k = \pm q^{\sigma(k+1)} \bmod p$$

и

$$1 - q^k = \pm q^{\mu(k)} \bmod p$$

(в последнем случае только при $k > 0$), тогда

$$\zeta^{1+q^k} + \zeta^{-1-q^k} = \zeta^{q^{\sigma(k+1)}} + \zeta^{-q^{\sigma(k+1)}} = \alpha^{q^{\sigma(k+1)}},$$

$$\zeta^{1-q^k} + \zeta^{-1+q^k} = \zeta^{q^{\mu(k)}} + \zeta^{-q^{\mu(k)}} = \alpha^{q^{\mu(k)}}, k > 0.$$

Также как и в случае второго типа при $n > k > 0$ имеем

$$\alpha\alpha^{q^k} = (\zeta + \zeta^{-1})(\zeta^{q^k} + \zeta^{-q^k}) = \zeta^{1+q^k} + \zeta^{-1-q^k} + \zeta^{1-q^k} + \zeta^{-1+q^k} =$$

$$= \alpha^{q^{\sigma(k+1)}}, +\alpha^{q^{\mu(k)}},$$

а при $k = 0$

$$\alpha\alpha^{q^0} = \zeta^{1+q^0} + \zeta^{-1-q^0} + \zeta^{1-q^0} + \zeta^{-1+q^0} =$$

$$= \zeta^{1+q^0} + \zeta^{-1-q^0} + \zeta^0 + \zeta^0 = \zeta^{1+q^0} + \zeta^{-1-q^0} == \alpha^{q^{\sigma(1)}}.$$

Опять, как и в случае второго типа, вспоминая определение матрицы T посредством равенств

$$\alpha\alpha^{q^i} = \sum_{j=0}^{n-1} t_{i,j}\alpha^{q^j},$$

и сравнивая их с полученными равенствами

$$\alpha\alpha^{q^i} = \alpha^{q^{\sigma(i+1)}} + \alpha^{q^{\mu(i)}}, i > 0,$$

$$\alpha\alpha^{q^0} = \alpha^{q^{\sigma(1)}},$$

имеем

$$t_{i,j} = \delta_{\sigma(i+1),j} + \delta_{\mu(i),j}, i \neq 0,$$

$$t_{0,j} = \delta_{\sigma(1),j},$$

где $\delta_{k,s}$ — дельта-символ Кронекера.

Отметим, что при нечетном q указанный базис не будет, как и в случае второго типа, оптимальным, но будет иметь линейную сложность.

В заключение приведем в явном виде

Алгоритм построения оптимальных нормальных базисов третьего типа и таблицы умножения в них при $q = 2$

Пусть надо построить оптимальный нормальный базис в поле $GF(2^n)$.

1. Проверяем, что n нечетно. Если да, то проверяем, является ли $p = 2n + 1$ простым числом.

2. Если да, то проверяем, что

$$2^n = 1 \bmod p,$$

и

$$2^{n/\delta} \bmod p \neq 1$$

для каждого простого делителя δ числа n.

3. Если да, то вычисляем матрицу $T = (t_{i,j})$.

Сначала строим массив

$$\pi(k) = 2^k \bmod q, k = 0, \ldots, 2n - 1$$

Потом строим массив $(\sigma(1), \ldots, \sigma(n))$ такой, что $1 \leq \sigma(k) \leq n$ и

$$1 + 2^k \bmod p = \pm\pi(\sigma(k + 1)), k = 0, \ldots, n - 1.$$

Потом строим массив $(\mu(1), \ldots, \mu(n - 1))$, такой что $1 \leq \mu(k) \leq n$ и

$$1 - 2^k \bmod p = \pi(\mu(k)), k = 1, \ldots, n - 1$$

И наконец как и в случае второго типа вычисляем для всех $0 < i < n$ и всех j, $0 \leq j < n$

$$t_{i,j} = \delta_{\sigma(i+1),j} + \delta_{\mu(i),j},$$

$$t_{0,j} = \delta_{\sigma(1),j},$$

где $\delta_{k,s}$ — дельта-символ Кронекера.

Следующий факт имеется в [138].

Упражнение 2.3.4 Докажите, что для матрицы A оптимального нормального базиса в поле $GF(2^n)$ второго или третьего типа $a_{i,j} = 1$ тогда и только тогда, когда выполняется одно из четырех соотношений

$$2^i \pm 2^j = \pm 1 \bmod 2n + 1.$$

В [79] имеется следующее описание матрицы A

Упражнение 2.3.5 Докажите, что все единичные элементы матрицы A определяются как $a_{x_i,x_{i+1}}$ или a_{x_{i+1},x_i}, $i = 0, \ldots, n - 1$, где $x_i = \log_2(i + 1) \bmod n, i < n - 1, x_{n-1} = n - 1$, логарифмы вычисляются в поле $GF(p), p = 2n + 1$.

2.3.6 Некоторые примеры примитивных элементов по модулю p

Для поиска оптимальных нормальных базисов в полях $GF(2^n)$ полезно знать, когда двойка или минус двойка является примитивным элементом по модулю p. Действительно, для того чтобы в поле $GF(2^n)$ существовал базис третьего типа необходимо, чтобы n было нечетным и мультипликативный порядок двойки по модулю $p = 2n + 1$ был равен $(p - 1)/2 = n$.

Упражнение 2.3.6 Докажите, что указанное выше условие равносильно тому, что минус двойка является примитивным элементом по модулю p.

До сих пор неизвестно, будет ли двойка примитивным элементом для бесконечно многих простых p. Гипотеза Артина утверждает, что любое целое число a, не равное ± 1 и не являющееся квадратом, будет примитивным элементом для бесконечно многих простых p. Гипотеза Артина была доказана Хис-Брауном для всех простых a, кроме быть может двух, однако эти два исключения неизвестны.

В связи с этим представляет интерес следующая теорема, принадлежащая П.Л.Чебышеву.

Теорема 2.3.1 *Пусть простые числа $p, q > 2$. Тогда справедливы следующие утверждения:*

- *если $p = 4q + 1$, то 2 есть примитивный элемент по модулю p,*
- *если $p = 2q + 1, q = 4n + 1$, то 2 есть примитивный элемент по модулю p,*
- *если $p = 2q + 1, q = 4n + 3$, то -2 есть примитивный элемент по модулю p.*

2.4 Оптимизация преобразований базисов

2.4.1 О комбинированном использовании полиномиального и нормального базисов

Имплементированные стандартные алгоритмы умножения в оптимальных нормальных базисах оказались медленнее ([12],[13]) алгоритмов умножения в стандартных базисах даже в полях небольших размерностей (в диапазоне $150 - 350$), а с ростом размерности они становятся еще хуже. Сравнение двух типов базисов, стандартного и нормального, наводит на мысль об ускорении арифметики в конечных полях за счет использования выгодных сторон каждого из них. Действительно, умножение быстрее производить в стандартном представлении поля $GF(2^n)$), а возведение в степень — в нормальном представлении.

Для реализации этой этой идеи понадобятся матрицы перехода от нормального базиса к стандартному и обратно, которые могут оказаться не разряженными, а плотными, и тогда сложность перехода от одного базиса к другому в худшем случае будет $O(n^2/\log n)$ (если использовать для умножения матрицы на вектор метод «четырех русских»).

Но в удачном случае сложность перехода может оказаться даже не выше линейной, например, в случае, если число ненулевых элементов в матрицах переходов будет $O(n)$. Таким образом возникает задача поиска нормальных базисов с «простыми» матрицами переходов к стандартным базисам и обратно.

Эта задача легко решается в случае оптимальных нормальных базисов первого типа. Напомним, что первый тип нормальных базисов возникает лишь в случае, когда $n + 1 = p$ – простое и q – примитивный корень по модулю p.

Далее под сложностью понимается арифметическая сложность в классе схем из операций сложения и умножения в основном подполе $GF(q)$.

Теорема 2.4.1 *Переход от стандартного базиса поля $GF(q^n)$ к соответствующему (с тем же генератором) оптимальному нормальному базису первого типа (если, конечно, он существует для данного n) и обратно можно выполнить с линейной сложностью.*

Доказательство. Легко видеть, что в этом случае базис $\{\zeta, \ldots, \zeta^n\}$ (не совсем стандартный) совпадает с точностью до перестановки с оптимальным нормальным базисом

$$\{\zeta, \zeta^q, \zeta^{q^2}, \ldots, \zeta^{q^{n-1}}\},$$

так как последовательность чисел $1, q, q^2, \ldots, q^{n-1}$, вычисленных по модулю p, совпадает с некоторой перестановкой

$$\{1, 2, \ldots, n\}$$

в силу того, что ζ есть примитивный корень p-й степени из единицы в поле $GF(q^n)$, а q – примитивный корень по модулю p.

Поэтому, очевидно, переход от базиса $\{\zeta, \ldots, \zeta^n\}$ к нормальному базису

$$\{\zeta, \zeta^q, \zeta^{q^2}, \ldots, \zeta^{q^{n-1}}\}$$

и обратно выполняется с оценкой сложности не более, чем n.

Отметим попутно, что ζ является корнем неприводимого над полем $GF(q)$ многочлена

$$1 + x + \ldots + x^n = \frac{x^p - 1}{x - 1}\,.$$

Стандартным базисом для перехода будет базис $\{1, \zeta, \ldots, \zeta^{n-1}\}$. Очевидно, он с линейной сложностью выражается через базис $\{\zeta, \ldots, \zeta^n\}$, поскольку

$$\zeta^n = 1 + \zeta + \ldots + \zeta^{n-1},$$

и обратно базис $\{\zeta, \ldots, \zeta^n\}$, благодаря формуле

$$1 = \zeta + \ldots + \zeta^{n-1} + \zeta^n,$$

с линейной сложностью выражается через базис $\{1, \zeta, \ldots, \zeta^{n-1}\}$. В результате имеем, что переход от оптимального нормального базиса

$$\{\zeta, \zeta^q, \zeta^{q^2}, \ldots, \zeta^{q^{n-1}}\}$$

к стандартному базису $\{1, \zeta, \ldots, \zeta^{n-1}\}$ и обратно выполняется с линейной сложностью.

Умножение в стандартном базисе $\{1, \zeta, \ldots, \zeta^{n-1}\}$, как обычно, сводится к умножению многочленов над полем $GF(q)$ и последующей редукции по модулю неприводимого многочлена $f(x) = 1 + x + \ldots + x^n$, соответствующего этому базису. Отметим, что деление с остатком на этот многочлен можно выполнить с линейной сложностью.

На самом деле удобно не переходить к базису $\{1, \zeta, \ldots, \zeta^{n-1}\}$, а работать в базисе $\{\zeta, \ldots, \zeta^n\}$. Тогда нужно будет перемножать многочлены вида $a_1 x + \ldots + a_n x^n$ и полученный результат приводить по модулю $f(x)$ (так как переход к стандартному базису не изменяет многочлен по этому модулю.) Умножение таких многочленов очевидно сводится к умножению многочленов степени $n - 1$. Полученное произведение $g(x)$ степени $2n$ надо разделить на $f(x)$ и найти остаток (только он нам и нужен), т. е. надо представить $g(x)$ в виде $g(x) = f(x)h(x) + r(x)$, где степень $r(x)$ меньше n. Умножив обе части равенства на $x - 1$, получим $g(x)(x-1) = (x^p-1)h(x)+r(x)(x-1)$, а так как степень $r(x)(x-1)$ меньше $n+1 = p$, то $r(x)(x-1)$ — остаток от деления $g(x)(x-1)$ на $x^p - 1$. Для умножения $g(x)$ на $x-1$ достаточно сделать $2n-2$ сложений (в поле $GF(q)$), а для деления результата на $x^p - 1$ достаточно выполнить $n - 2$ сложений и, наконец, для деления полученного остатка $r(x)(x-1)$ на $x-1$ с помощью схемы Горнера достаточно будет сделать n сложений.

Упражнение 2.4.1 Докажите, что сложность умножения в поле $GF(2^n)$ в случае использования оптимального нормального базиса первого типа не превосходит $M(n) + 4n - 4$, где $M(n)$— сложность умножения двоичных многочленов степени $n - 1$.

2.4.2 Пример выполнения алгоритма перехода от оптимального базиса первого типа к стандартному и обратно

Переход от оптимального нормального базиса первого типа поля $GF(2^4)$ к стандартному.

Пусть дано: $n = 4$, оптимальный нормальный базис первого типа $B = \{\zeta^{2^k}, k = 0, 1, 2, 3\}$, элемент, записанный в этом базисе, $f = 1 \cdot \zeta^{2^0} + 0 \cdot \zeta^{2^1} + 1 \cdot \zeta^{2^2} + 1 \cdot \zeta^{2^3}$.

Надо найти: запись f в стандартном базисе $A = \{1, \zeta, \zeta^2, \zeta^3\}$.

Шаг 1. *Переход от базиса нормального к почти стандартному базису* $A' = \{\zeta, \ldots, \zeta^n\}$.

В начале найдем перестановку $\pi(i)$ множества чисел $\{1, \ldots, n\}$, совпадающей с последовательностью степеней $1, 2, 2^2, \ldots, 2^{n-1}$, вычисленной по модулю $p = n + 1$:

$$\pi = \begin{pmatrix} 2^0 & 2^1 & 2^2 & 2^3 & (mod 5) \\ 1 & 2 & 4 & 3 & (mod 5) \end{pmatrix}, \tag{4.2.1}$$

т.е. в базисе A' многочлен f запишется следующим образом:

$$f = 1 \cdot \zeta + 0 \cdot \zeta^2 + 1 \cdot \zeta^4 + 1 \cdot \zeta^3.$$

Шаг 2. *Переход от почти стандартного базиса к стандартному базису.* Выразим единицу, и тогда в стандартном базисе A многочлен f запишется следующим образом:

$$f = (1 + 1)\zeta + (0 + 1)\zeta^2 + (1 + 1)\zeta^3 + 1 \cdot \zeta^0$$

т.е. $f = 1 + \zeta^2$.

Переход от стандартного базиса поля $GF(2^4)$ к оптимальному нормального базису 1*-го типа.*

Пусть дано: $n = 4$, стандартный базис $A = \{1, \zeta, \zeta^2, \zeta^3\}$, многочлен, записанный в этом базисе, $f = 1 + x^2 + x^3$.

Надо найти: запись f в оптимальном нормальном базис 1-го типа $B = \{\zeta^{2^k}, k = 0, \ldots, 3\}$.

Шаг 1. *Переход от стандартного базиса к почти стандартному базису.*

Выразим коэффициент при ζ^n и тогда в почти стандартном базисе A' многочлен f запишется следующим образом:

$$f = (0 + 1)\zeta + (1 + 1)\zeta^2 + (1 + 1)\zeta^3 + 1 \cdot \zeta^4$$

т.е. $f = \zeta + \zeta^4$.

Шаг 2. *Переход от почти стандартного базиса к оптимальному нормальному базису.*

Сделав обратную перестановку к перестановке 2.4.2, перейдем к записи многочлена в оптимальном нормальном базисе B:

$$f = \zeta + \zeta^{2^2}.$$

2.4.3 Примеры построения комбинированных схем для умножения в стандартном и оптимальном нормальном базисе первого типа

Здесь и далее под схемами понимаются *логические схемы*, построенные из двухвходовых элементов, реализующих умножение (конъюнкцию) и сложение по модулю два (операция XOR — по-русски «исключающее или»). Элементы схемы соединяются друг с другом и с входами и выходами схемы произвольным образом с естественным *запретом на образование ориентированных циклов* (которые могут существовать в автоматных схемах и используются для

создания обратной связи). *Сложностью схемы* называется число элементов в ней. *Глубиной схемы* называется максимальное число элементов, какое может быть в цепи, соединяющей один из входов схемы с одним из ее выходов. Глубина пропорциональна задержке схемы. В рассматриваемых далее примерах мы не стремимся минимизировать глубину, и оцениваем ее только для повышения наглядности в условиях отсутствия рисунков. Сложность схемы очевидно пропорциональна времени работы программы, имплементирующей тот же алгоритм, который реализуется схемой. Поэтому понятие сложности схемы по-существу совпадает с понятием так называемой битовой сложности соответствующего алгоритма.

Для примера разберем случай поля $GF(2^{10}$. В нем существует оптимальный нормальный базис первого типа, и можно сравнить схемы для этого базиса со стандартными схемами. Сначала возьмем стандартный базис $1, \alpha, \alpha^2, \ldots \alpha^9$ с минимальным многочленом $p(x) = 1 + x^3 + x^10$, $p(\alpha) = 0$. Есть еще неприводимый трехчлен $1 + x^7 + x^10$, он взаимен к этому, но для деления удобен именно первый из них, потому что в нем степень среднего члена меньше половины степени старшего. Схема для умножения произвольных элементов

$$x_0 + x_1\alpha + \ldots + x_9\alpha^9, y_0 + y_1\alpha + \ldots + y_9\alpha^9,$$

состоит из схемы для обычного умножения двоичных многочленов, дающей результат

$$u_0 + u_1\alpha + \ldots + u_{18}\alpha^{18},$$

и схемы, находящей остаток от деления этого результата на трехчлен $p(x)$. Эту схему проще построить непосредственно, не сводя деление к умножению, как в общем случае деления на произвольный многочлен. Сложность этой схемы будет линейная, а глубина — константная. Например, для этой схемы можно выписать формулы для ее выходов $z_0, \ldots, z_9$ (w_i обозначают промежуточные результаты):

$$z_0 = w_0 + u_0 = u_{10} + u_{17} + u_0,$$

$$z_1 = w_1 + u_1 = u_{11} + u_{18} + u_1,$$

$$z_2 = w_2 + u_2 = u_{12} + u_2,$$

$$z_3 = w_0 + (w_3 + u_3) = u_{10} + u_{17} + u_{13} + u_3,$$

$$z_4 = w_1 + (w_4 + u_4) = u_{11} + u_{18} + u_{14} + u_4,$$

$$z_{5+i} = w_{5+i} + u_{5+i} + w_{2+i} = u_{15+i} + u_{12+i} + u_{5+i}, i = 0, \ldots, 3$$

$$z_9 = w_6 + u_9 = u_{16} + u_9.$$

Сложность стандартной схемы для умножения многочленов степени 9 равна $10^2 + 9^2 = 181$, а глубина равна 5, причем такую глубину имеют только выходы u_8, u_9, u_{10}. Глубина $w_1 = u_{11} + u_{17}$ тоже равна 5 так как это билинейная форма

$$x_9y_2 + \ldots + x_2y_9 + x_8y_9 + x_9y_8$$

с 10 слагаемыми. Глубина $w_4 + u_4 = u_{14} + u_4$ по той же причине Поэтому глубина $z_4 = w_1 + (w_4 + u_4)$ равна 6. Аналогично глубина z_3 равна 6. Глубина остальных выходов тоже не больше 6. Сложность всей схемы приведения по модулю $p(x)$ равна 18, а полная сложность равна 199.

Полином $1+x+\ldots+x^{10}$ неприводим. Можно его использовать вместо $p(x)$. Удобно далее вместо базиса $1, \alpha, \alpha^2, \ldots \alpha^9$ с минимальным многочленом $p(x)$, $p(\alpha) = 0$ использовать базис $\alpha, \alpha^2, \ldots \alpha^{10}$.

Схема для умножения произвольных элементов

$$x_0\alpha + x_1\alpha^2 + \ldots + x_9\alpha^{10}, y_0\alpha + y_1\alpha^2 + \ldots + y_9\alpha^{10},$$

состоит из схемы для обычного умножения двоичных многочленов, дающей результат

$$u_0\alpha^2 + u_1\alpha^3 + \ldots + u_{18}\alpha^{20},$$

и схемы, находящей остаток от деления этого результата на многочлен $p(alpha)$ в виде $z_0\alpha + \ldots + z_9\alpha^{10}$. Для такого деления можно выполнить обычное деление многочлена

$$u_0\alpha + u_1\alpha^2 + \ldots + u_{18}\alpha^{19}$$

на многочлен $p(\alpha)$ в с обычным остатком виде $z_0 + \ldots + z_9\alpha^9$. Чтобы выполнить это деление, сначала разделим

$$u_0\alpha + u_1\alpha^2 + \ldots + u_{18}\alpha^{19}$$

на $\alpha^{11} + 1$ и найдем остаток, равный

$$u_{10} + (u_0 + u_{11})\alpha + (u_1 + u_{12})\alpha^2 + \ldots + (u_7 + u_{18})\alpha^8 + u_8\alpha^9 + u_9\alpha^{10},$$

а потом поделим этот остаток на $p(\alpha)$, для чего нужно отнять от него $u_9 p(\alpha)$, и получим

$$z_0 + z_1\alpha + \ldots + z_9\alpha^9 =$$

$$u_{10} + u_9 + (u_0 + u_{11} + u_9)\alpha + (u_1 + u_{12} + u_9)\alpha^2 + \ldots + (u_7 + u_{18} + u_9)\alpha^8 + (u_8 + u_9)\alpha^9,$$

— остаток от деления рассматриваемого произведения на $p(\alpha)$.

Глубина $z_0 = u_{10} + u_9$ равна 6. Аналогично глубина z_9 равна 6. Глубина остальных выходов тоже не больше 6, так как $u_i + u_{11_i}$ это билинейная форма с 9 слагаемыми. Сложность всей схемы приведения по модулю $p(x)$ равна 18, а полная сложность равна 199.

Нормальный базис, порожденный элементом α состоит из элементов

$$\{\alpha, \alpha^2, \alpha^4, \alpha^8, \alpha^{16}, \ldots, \alpha^{2^9}\}.$$

Он же будет оптимальным нормальным базисом. На самом деле он просто является перестановкой базиса

$$\{\alpha, \alpha^2, \ldots \alpha^{10}\}, 1 + \alpha + \ldots \alpha^{10} = 0.$$

Действительно $\alpha^{11} = 1, \alpha^k \neq 1, k = 1, \ldots, 10$, а так как

$$2^4 = 5 \bmod 11,$$

$$2^5 = 10 \bmod 11,$$

$$2^6 = 9 \bmod 11,$$

$$2^7 = 7 \bmod 11,$$

$$2^8 = 3 \bmod 11,$$

$$2^9 = 6 \bmod 11,$$

$$2^{10} = 1 \bmod 11,$$

то

$$\alpha^{16} = \alpha^5, \alpha^{32} = \alpha^{10}, \alpha^{64} = \alpha^9,$$

$$\alpha^{128} = \alpha^7, \alpha^{256} = \alpha^3, \alpha^{512} = \alpha^6.$$

Поэтому переход от координат $(x_1, \ldots, x_{10})$ в базисе

$$\{\alpha, \alpha^2, \alpha^4, \alpha^8, \alpha^{16}, \ldots, \alpha^{2^9}.\}$$

к координатам в базисе

$$\{\alpha, \alpha^2, \dots \alpha^{10}\}$$

сводится к их перестановке в порядке $(x_1, x_2, x_4, x_8, x_5, x_{10}, x_9, x_7, x_3, x_6)$ Обратный переход сводится к обратной перестановке.

Общая оценка сложности мультиплера Месси-Омура равна $n(C + n - 1)$, и в рассматриваемом случае она равна 280, а глубина равна 6.

Однако в оптимальном нормальном базисе первого типа можно построить схему той же глубины и меньшей сложности, выполнив вначале переход к соответствующему стандартному базису с помощью указанной выше перестановки координат, потом выполнить описанное умножение в стандартном базисе со сложностью 199 и глубиной 6, а потом вернуться в нормальный базис, выполнив обратную перестановку.

Теперь можно описать как делать инвертирование с помощью нормальных базисов. Пусть

$$\xi = \sum_{i=0}^{n-1} x_i \alpha^{2^i}$$

произвольный ненулевой элемент поля $GF(2^n)$, записанный в любом нормальном базисе

$$\{\alpha, \alpha^2, \alpha^4, \alpha^8, \alpha^{16}, \dots, \alpha^{2^9}\}.$$

По теореме Ферма обратный элемент можно вычислить по формуле $\xi^{-1} = \xi^{2^n-2}$. Для примера сделаем это при $n = 10$. Для возведения в степень надо построить аддитивную цепочку для числа $2^n - 2$, содержащую минимальное число сложений (а удвоения можно не учитывать). Для этого сначала надо построить минимальную линейную аддитивную цепочку для числа $n - 1 = 9$. (линейную — значит в ней каждое число получается или удвоением предыдущего числа или прибавлением к нему какого-то более раннего числа из этой цепочки). Например, ею будет цепочка

$$1, 2, 4, 8, 9.$$

После этого выписываются в ряд числа

$$2^1 - 1, 2^2 - 1, 2^4 - 1, 2^8 - 1, 2^9 - 1,$$

и между ними вставляются последовательности удвоений

$$1 = 2^1 - 1, 2, 3 = 2^2 - 1, 3 \cdot 2, 3 \cdot 4, 15 = 2^4 - 1, 2(2^4 - 1), 4(2^4 - 1), 8(2^4 - 1), 16(2^4 - 1),$$

$$255 = 2^8 - 1, 2(2^8 - 1), 511 = 2^9 - 1, 1022 = 2(2^9 - 1).$$

Так как удвоения реализуются путем соответствующей коммутации входов, то сложность инвертирования в поле $GF(2^{10})$ оценивается как

$$I(10) \le 4M(10),$$

где $M(10)$ сложность умножения в нормальном базисе. Аналогично глубина оценивается как

$$DI(10) \le 4DM(10).$$

Если использовать указанную выше схему для умножения в нормальном базисе, то сложность инвертирования будет $4 \cdot 199 = 796$, а глубина $4 \cdot 6 = 24$.

Используя метод Карацубы, умножение двух двоичных многочленов степени 9 можно свести к трем умножениям многочленов степени 4, и сложность будет 159, а глубина 6. Если добавить схему приведения по модулю многочлена $1 + x + \dots + x^10$, то получается схема сложности 178, а глубины 7.

2.4.4 Оценка сложности перехода от оптимальных нормальных базисов второго и третьего типа к стандартным и обратно

Вначале докажем несколько вспомогательных утверждений (следующая далее лемма выполняется и для общего случая).

Пусть $\alpha = \alpha_1$ – генератор оптимального нормального базиса

$$\{\alpha_1, \ldots, \alpha_n\}$$

второго типа поля $GF(q^n)$, он же генератор соответствующего стандартного базиса

$$\{\alpha^0, \ldots, \alpha^{n-1}\}$$

т.е. $\alpha_1 = \alpha = \zeta + \zeta^{-1}$, и при любом $k \leq n$

$$\alpha_k = \alpha^{q^{k-1}} = \zeta^{q^{k-1}} + \zeta^{-q^{k-1}},$$

где ζ – примитивный корень из 1 степени $p = 2n + 1$ в поле $GF(q^{2n})$.

Пусть $q = p^r$, где p – простое число и $\alpha = \zeta + \zeta^{-1}$. Рассмотрим вспомогательную последовательность $a_0, a_1, \ldots$, порожденную элементом α. Положим $a_0 = 2$ (в случае $p = 2$ естественно $a_0 = 0$) и, далее, $a_k = \zeta^k + \zeta^{-k}, k = 1, 2, \ldots, a_1 = \alpha_1$.

Лемма 2.4.1 *Для любого $k \geq 1$ имеют место следующие рекуррентные соотношения:*

1. $a_{k+1} = a_k a_1 - a_{k-1}$,
2. $a_{p^k} = a^{p^k}$,
3. $a_{p^k+i} = a_i a^{p^k} - a_{p^k-i}$.

Доказательство. Для доказательства первой формулы заметим, что

$$(\zeta^k + \zeta^{-k})(\zeta + \zeta^{-1}) = \zeta^{k-1} + \zeta^{-k+1} + \zeta^{k+1} + \zeta^{-k-1},$$

т.е.

$$a_k a_1 - a_{k-1} = a_{k+1}.$$

Вторая формула очевидна.

Третья формула проверяется непосредственно:

$$a_{p^k+i} = \zeta^{p^k+i} + \zeta^{-p^k-i} = (\zeta^{p^k} + \zeta^{-p^k})(\zeta^i + \zeta^{-i}) - (\zeta^{p^k-i} + \zeta^{-p^k+i}) =$$

$$= a_{p^k} a_i - a_{p^k-i} = a^{p^k} a_i - a_{p^k-i}.$$

Рекуррентные соотношения, доказанные в лемме, позволяют записать формулы перехода от нормального базиса к почти стандартному базису

$$\{\alpha^1, \ldots, \alpha^n\}.$$

В случае $p = 2$ вычитание в лемме естественно можно заменить сложением.

Из леммы следует, что для любого $i \geq 1$ элемент нормального базиса α_i выражается в виде значения некоторого многочлена степени i над полем $GF(q)$, т.е.

$$\alpha_i = f_i(\alpha) = \sum_{j=1}^{i} f_{i,j} \alpha^j.$$

Выразив таким образом все α_i в нормальном базисе, получим матрицу перехода $F_n = (f_{i,j})$ от почти стандартного базиса к нормальному. В этом случае (т.е. когда $p = 2$) плотность $S(F_n)$ (количество ненулевых элементов) матрицы F_n оценивается следующим образом.

Теорема 2.4.1 *Плотность матрицы перехода от стандартного базиса поля $GF(2^n)$ к нормальному базису второго или третьего типа равна $O(n^{\log_2 3})$.*

Из этой теоремы видно, что матрица перехода является разряженной, и уже это позволяет осуществлять быстрое умножение в этом базисе. Далее будет получена еще лучшая оценка сложности вычисления перехода от стандартного базиса к нормальному и обратно, чем непосредственно вытекающая из этой теоремы. Хотя сама по себе теорема использоваться не будет, приведем все же кратко ее доказательство, так в его ходе проясняется структура матрицы перехода.

Согласно формулам леммы 5.4.1, матрица F_n, построенная с их помощью, имеет при $n = 2^k - 1$ вид:

$$F_{2n+1} = \begin{pmatrix} F_n & o_n & O_n \\ 0 \ldots 0 & 1 & 0 \ldots 0 \\ G_n & o_n & F_n \end{pmatrix},$$

где o_n – нулевой вектор-столбец высоты n, O_n – нулевая $n \times n$ матрица, матрица G_n есть симметричное отражение матрицы F_n относительно средней строки, т.е. $G_n = I_n F_n$, где $I_n = (\delta_{i,n-i+1})$ – матрица с единицами на побочной диагонали и нулями в остальных местах. На пример, матрица F_7 выглядит так:

$$F_7 = \begin{pmatrix} 1 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & 0 & 0 \\ 1 & 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 & 0 \\ 1 & 0 & 1 & 0 & 1 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & 1 & 0 \\ 1 & 0 & 0 & 0 & 1 & 0 & 1 \end{pmatrix}.$$

Если разбить матрицу на 4 квадратных подматрицы, удалив среднюю строку и средний столбец, получим, что левый верхний квадрат 3×3 симметричен относительно горизонтали левому нижнему и равен нижнему правому квадрату. Средняя строка и средний столбец содержат ровно одну единицу, лежащую на их пересечении. Матрица является нижнетреугольной с единицами на главной диагонали.

Действительно, в общем случае согласно лемме 5.4.1 при $0 \le i \le 2^k$

$$\sum_{j=1}^{2^k+i} f_{2^k+i,j}\alpha^j = a_{2^k+i} = a_i a^{2^k} + a_{2^k-i} =$$

$$= \sum_{j=1}^{i} f_{i,j}\alpha^{2^k+j} + \sum_{j=1}^{2^k-i} f_{2^k-i,j}\alpha^j,$$

откуда имеем $f_{2^k+i,j} = f_{2^k-i,j}$, при $0 \le j \le 2^k$, и $f_{2^k+i,2^k+j} = f_{i,j}$, при $1 \le j \le 2^k$.

Опираясь на это представление, можно получить следующую рекуррентную формулу для вычисления плотности последовательности матриц F_n

$$S(F_{2n+1}) = 3S(F_n) + 1\,, n \ge 3\,, S(F_3) = 4,$$

из которой вытекает рекуррентная формула

$$S(F_{2^k-1}) = 3S(F_{2^{k-1}-1}) + 1,\ k \ge 2,\ S(F_3) = 4.$$

Полагая $l(k) = S(F_{2^k-1})$, перейдем к линейному рекуррентному соотношению $l(k) = 3l(k-1)+1, l(2) = 4$, решением которого будет $l(k) = (3^k-1)/2$. Обозначив $n = 2^k - 1$ и выразив $3^k = (2^k)^{\log_2 3} = (n+1)^{\log_2 3}$, получим, что

$$S(F_n) = l(k) = ((n+1)^{\log_2 3} - 1)/2 = O(n^{\log_2 3}).$$

В общем случае равенство $S(F_n) = O(n^{\log_2 3})$ сохраняется, так как выбрав k таким образом, что $2^k - 1 < n \le 2^{k+1} - 1$, можно заметить, что матрица F_n является главной подматрицей матрицы $F_m, m = 2^k - 1$, откуда имеем

$$S(F_n) \le S(F_m) = O(m^{\log_2 3}) = O(n^{\log_2 3}).$$

Аналогично можно оценить плотность матрицы F'_n перехода от стандартного базиса поля $GF(2^n)$ к нормальному. Обозначим B_n матрицу перехода от стандартного базиса к почти стандартному, тогда $F'_n = F_n \cdot B_n$, и непосредственно проверяется, что в матрице B_n все элементы нулевые, кроме наддиагональных элементов $b_{i,i+1} = 1$ и некоторых элементов нижней строки $b_{n,j}$, и справедливы следующие равенства для элементов матрицы F'_n

$$f'_{i,1} = f_{i,n}b_{n,1} = \delta_{i,n}b_{n,1},\ f'_{i,j} = f_{i,j-1} + f_{i,n}b_{n,j} = f_{i,j-1} + \delta_{i,n}b_{n,j},$$

так как в силу нижней треугольности матрицы F_n для ее элементов имеем $f_{i,n} = \delta_{i,n}$, где $\delta_{i,n}$ – дельта-символ Кронекера. Складывая эти равенства, получим, что

$$S(F'_n) = \sum_{i=1,j=1}^{n,n-1} f_{i,j} + \sum_{j=1}^{n} b_{n,j} = S(F_n) - 1 + \sum_{j=1}^{n} b_{n,j} \le$$

$$\le S(F_n) - 1 + n = O(n^{\log_2 3}).$$

Обозначим через $L(F_n)$ сложность линейного преобразования, определяемого матрицей F_n (в данном случае – это наименьшее число операций сложения по модулю два, необходимых для вычисления этого преобразования).

Теорема 2.4.2 $L(F_n) \le \frac{n}{2}\log_2 n + 2n = O(n \log_2 n)$.

Доказательство. Нам будет удобно оценивать сложность преобразования, задаваемого транспонированной матрицей F_n^T, которая равна $L(F_n)$ согласно известной лемме о взаимосвязи сложности транспонированных матриц (см., например [24]). Впрочем, для дальнейшего нам нужно будет оценить сложность перехода от координат в нормальном базисе $\sum_{i=1}^n x_i\alpha_i$ к координатам в почти стандартном базисе $\sum_{i=1}^n y_i\alpha^i$,, которое определяется как раз матрицей F_n^T. Пусть $2^k \le n \le 2^{k+1}$, тогда с использованием формул из леммы 5.4.1 имеем

$$\sum_{i=1}^{n} y_i\alpha^i = \sum_{i=1}^{n} x_i a_i =$$

$$= \sum_{i=1}^{2^k} x_i a_i + \sum_{i=2^k+1}^{n} x_i a_i = \sum_{i=1}^{2^k} x_i a_i + \sum_{i=1}^{n-2^k} x_{i+2^k} a_{i+2^k} =$$

$$= \sum_{i=1}^{2^k} x_i a_i + \sum_{i=1}^{n-2^k} x_{i+2^k}(a^{2^k} a_i + a_{2^k-i}) =$$

$$= x_{2^k}a_{2^k} + \sum_{i=1}^{2^{k+1}-n-1} x_i a_i + \sum_{i=2^{k+1}-n}^{2^k-1} (x_i + x_{2^{k+1}-i})a_i + a^{2^k}\sum_{i=1}^{n-2^k} x_{i+2^k}a_i,$$

где $a_0 = 0$. Далее, определяя вектора-столбцы

$$X = \{x_1, \ldots, x_n\}^T,$$

$$X_1 = \{x_1, \ldots, x_{2^{k+1}-n-1}, x_{2^{k+1}-n} + x_n, \ldots, x_{2^k-1} + x_{2^k+1}, x_{2^k}\}^T,$$

где, естественно, в случае $n = 2^{k+1} - 1$

$$X_1 = \{x_1 + x_n, \ldots, x_{2^k-1} + x_{2^k+1}, x_{2^k}\}^T,$$

а в случае $n = 2^{k+1}$

$$X_1 = \{x_1 + x_{n-1}, \ldots, x_{2^k-1} + x_{2^k+1}, x_{2^k}\}^T,$$

и во всех случаях

$$X_2 = \{x_{1+2^k}, \ldots, x_n\}^T,$$

и вектора-строки

$$Y_2 = \{y_{1+2^k}, \ldots, y_n\},$$

$$Y_1 = \{y_1, \ldots, y_{2^k}\},$$

получим

$$\sum_{i=1}^{n} y_i \alpha^i = \sum_{i=1}^{n} x_i a_i = \sum_{i=1}^{2^k} X_{1,i} a_i + \alpha^{2^k} \sum_{i=1}^{n-2^k} X_{2,i} a_i =$$

$$= \sum_{i=1}^{2^k} Y_{1,i} a^i + \sum_{i=1}^{n-2^k} Y_{2,i+2^k} a_{i+2^k},$$

значит

$$F_n^T \otimes X = (y_1' \ldots y_n') = (Y_1, Y_2) = (F_{2^k}^T \otimes X_1, F_{n-2^k}^T \otimes X_2.), \tag{2.1}$$

где $\otimes$ – операция умножения матрицы на вектор в поле $GF(2)$. Разумеется, последнее равенство можно было получить, основываясь только на структуре матрицы F_n. Осталось индуктивно оценить сложность преобразования координат, определяемого матрицей F_n^T. Согласно равенству (2.1)

$$L(2^{m+1}) \leq 2^m - 1 + 2L(2^m)\,, m \leq k-1,\ L(2) = 2.$$

По индукции непосредственно проверяется, что

$$L(2^m) = 2^{m-1} m + 1.$$

Для произвольного n в пределах $2^k < n < 2^{k+1}$ получим

$$L(n) \leq L(2^k) + L(n - 2^k) + n - 2^k.$$

Записывая n в двоичной системе

$$n = 2^{k_s} + \ldots + 2^{k_1},$$

где $k_s > \ldots > k_1$, получаем, что

$$L(n) \leq 2^{k_s - 1} k_s + \ldots + 2^{k_1 - 1} k_1 + s - 1 + (n - 2^{k_s}) + \ldots + (n - 2^{k_s} - \ldots - 2^{k_2}) \leq$$

$$\leq \frac{n}{2} \log_2 n + c\frac{n}{2},$$

где $c = 1\frac{2}{2} + \frac{3}{4} + \frac{4}{8} + \frac{5}{16} + \ldots < 4$.

Теорема 2.4.3 *Сложность $B(n)$ перехода в поле $GF(2^n)$ от оптимального нормального базиса второго или третьего типа к соответствующему стандартному и наоборот удовлетворяет неравенству $B(n) \leq \frac{n}{2} log_2 n + 3n$.*

Доказательство. Обозначим $\alpha = \zeta + \zeta^{-1}$ генератор стандартного базиса

$$A = \{1, \alpha^1, \ldots, \alpha^{n-1}\}$$

и базиса второго или третьего типа

$$B = \{\alpha^{2^0}, \alpha^{2^1}, \ldots, \alpha^{2^{n-1}}\}.$$

(Переход от нормального к стандартному.) Переход будет осуществляться через цепочку из четырех базисов

$$B \to B' \to A' \to A$$

и преобразования координат элемента f поля $GF(2^n)$ в этих базисах

$$f = \sum_{i=1}^{n} x_i \alpha^{2^{i-1}} = \sum_{i=1}^{n} x'_i a_i = \sum_{i=1}^{n} y'_i \alpha^i = \sum_{i=1}^{n} y_i \alpha^{i-1},$$

где a_i обозначает $\zeta^i + \zeta^{-i}$, $B' = \{a_1, \ldots, a_n\}$, $A' = \{\alpha^1, \ldots, \alpha^n\}$.

Переход от базиса B к базису B' и преобразование координат $\widetilde{x}$ к координатам $\widetilde{x}'$. Этот переход осуществляется перестановкой базисных элементов, и действительно, существует такая перестановка $\pi(i)$ чисел $\{1, \ldots, n\}$, что для любого $i = 1 \ldots 2n$ выполняется равенство

$$2^i \bmod p = \pm\pi(i) \in \{1, \ldots, n\}. \tag{2.2}$$

В самом деле, для базиса второго типа последовательность степеней

$$1, 2, 2^2, \ldots, 2^{2n-1},$$

вычисленных по модулю p, совпадает с некоторой перестановкой

$$\pi(1), \ldots, \pi(2n)$$

множества чисел $\{1, \ldots, 2n\}$ в силу того, что 2 – примитивный корень по модулю p. А в силу равенства

$$2^{k+n} = -2^k \pmod p,$$

(по теореме Ферма $2^{2n} = 1 \pmod p$, значит $2^n = -1 \pmod p$) окончательно получим формулу (2.2).

В случае же базиса третьего типа число 2 является квадратичным вычетом по модулю p, поэтому все степени $2^k \pmod p$, $k = 1, \ldots, n-1$ образуют перестановку множества всех квадратичных вычетов по модулю p, так как их ровно n штук. Добавив к этому тот факт, что p равно 3 по модулю 4, и поэтому -1 является квадратичным невычетом по модулю p, так как в противном случае, существовало бы такое число r, что $-1 = r^2 \bmod p$, а это бы приводило к противоречию с теоремой Ферма:

$$r^{p-1} = (r^2)^{(p-1)/2} = (-1)^{(p-1)/2} = -1 \bmod p,$$

и тот факт, что произведение вычета на невычет является невычетом, получаем, что последовательность $-2^k \bmod p, k = 0, \ldots, n-1$ образует перестановку множества всех квадратичных невычетов по модулю p, в итоге получаем, что и в этом случае верна формула (2.2).

Таким образом, мы получили, что базис $\{a_1, \ldots, a_n\}$ есть просто перестановка базиса $\{\alpha, \ldots, \alpha^{2^{n-1}}\}$. Отрицательные индексы можно заменить на соответствующие положительные

благодаря равенству $a_i = \zeta^i + \zeta^{-i} = a_{-i}$ по формуле $\alpha^{2^i} = a_{|\pi(i)|}$, (или для координат $x_i = x'_{|\pi(i)|}$). Поэтому далее везде будем считать индексы положительными, и знак модуля числа будем опускать.

Очевидно B' – тоже базис, т.к. он есть просто перестановка базиса B. Можно считать, что сложность перехода $L_{BB'}(n) = O(n)$, а если рассматривать не программную, а схемную имплементацию, то даже $L_{BB'}(n) = 0$.

Переход от базиса B' к базису A' и преобразование координат $\widetilde{x}'$ к координатам $\widetilde{y}'$. Согласно теореме 2.4.2 его сложность оценивается как

$$L(n) \leq \frac{n}{2} log_2 n + 2n.$$

Переход от базиса A' к базису A и преобразование координат $\widetilde{y}'$ к координатам $\widetilde{y}$. По рекуррентной формуле 2.1 можно в явном виде построить минимальный аннулирующий многочлен m_α (он строится лишь однажды, до применения алгоритма перехода, поэтому сложность его построения в сложности алгоритма не учитывается), т.е.

$$m_\alpha = f_n(\alpha) = b_1\alpha^0 + \ldots + b_n\alpha^n = 0, \tag{2.3}$$

где α — генератор этих базисов, и не все коэффициенты b_i равны нулю. Сделаем явный переход от координат $\widetilde{y}'$ в почти стандартном базисе A' к координатам $\widetilde{y}$ в стандартном A. Учитывая выражение (2.3),

$$\sum_{i=1}^{n} y'_i\alpha^i = \sum_{i=1}^{n-1} y'_i\alpha^i + y'_n(\sum_{i=0}^{n-1} b_i\alpha^i)\,,$$

после перегруппировки коэффициентов имеем

$$\sum_{i=1}^{n} y'_i\alpha^i = \sum_{i=1}^{n-1} \alpha^i(y'_i + b_i y'_n) + y'_n\alpha^0 = \sum_{i=1}^{n-1} y_i\alpha^i + y_0\alpha^0,$$

откуда видно, что сложность перехода от почти стандартного базиса A' к стандартному A оценивается как $L_{A'A}(n) \leq (n-1)$.

Итак, мы оценили сложность каждого преобразования, теперь осталось их сложить:

$$L_{BA}(n) = L_{BB'}(n) + L_{B'A'}(n) + L_{A'A}(n) \leq \frac{n}{2} log_2 n + 3n.$$

(Переход от стандартного к нормальному.) Теперь покажем, что теорема верна и при обратном переходе, т.е. от стандартного базиса A к базису второго или третьего типа B. Опять определим цепочку переходов, но уже в обратном порядке:

$$A \to A' \to B' \to B.$$

Переход от базиса A к базису A' и от координат $\widetilde{y}$ к $\widetilde{y}'$. Опять же в силу рекуррентной формулы (2.1) у нас есть минимальный аннулирующий многочлен (2.3), из явного вида которого мы можем выразить α^0, а именно:

$$\alpha^0 = \sum_{i=1}^{n} b_i\alpha^i. \tag{2.4}$$

Сделаем явный переход от координат $\widetilde{y}$ в стандартном базисе A к координатам $\widetilde{y}'$ в почти стандартном A'. Приняв во внимание выражение (2.4)

$$\sum_{i=0}^{n-1} y_i\alpha^i = \sum_{i=1}^{n-1} y_i\alpha^i + y_0(\sum_{i=1}^{n} b_i\alpha^i),$$

после перегруппировки коэффициентов получим

$$\sum_{i=0}^{n-1} y_i\alpha^i = \sum_{i=1}^{n-1} \alpha^i(y_i + b_i y_0) + y_0\alpha^n = \sum_{i=1}^{n-1} y'_i\alpha^i + y'_n\alpha^0,$$

откуда видно, что сложность перехода от стандартного базиса A к почти стандартному базису A' оценивается как $L_{AA'}(n) \le n-1$.

Переход от базиса A' к базису B' и преобразование координат $\widetilde{y}'$ к координатам $\widetilde{x}'$. Вычислим преобразование

$$(x'_1 \dots x'_n) = (F_n^{-1})^T \otimes \begin{pmatrix} y''_1 \\ \dots \\ y''_n \end{pmatrix},$$

умножения вектора Y^T на матрицу $(F_{2^{k+1}}^{-1})^T$ (обратную к матрице из теоремы 2.4.2) с помощью (2.1). В этой формуле рекуррентно выполнялись преобразования с векторами X_1 и X_2,, переводящие их в вектора $Y_1 = (y'_1, \dots, y''_{2^k})$ и $Y_2 = (y'_{2^k+1}, \dots, y'_n)$, составляющие вектор $Y = (y'_1, \dots, y'_n)$. Естественно, что обратное преобразование $(F_{n-2^k}^{-1})^T \otimes {Y_2}^T$ переведет вектор Y_2 в вектор X_2, а чтобы получить вектор X_1, надо сделать преобразование $(F_{2^k}^{-1})^T \otimes {Y_1}^T$. Так как

$$X_1 = \{x_1, \dots, x_{2^{k+1}-n-1}, x_{2^{k+1}-n} + x_n, \dots, x_{2^k-1} + x_{2^k+1}, x_{2^k}\}^T,$$
$$X_2 = \{x_{1+2^k}, \dots, x_n\}^T,$$

то для восстановления вектора $X = \{x_1, \dots, x_n\}$ по этим векторам достаточно к соответствующим $n - 2^k$ компонентам вектора X_1 прибавить (по mod2) компоненты вектора X_2.

Индуктивно продолжая описанное преобразование, мы перейдем к координатам $\widetilde{x}'$ в почти нормальном базисе B' со сложностью перехода

$$L^{-1}(n)) \le n - 2^k + L^{-1}(2^k) + L^{-1}(n - 2^k),$$

и, как и раньше, придем к оценке

$$L_{B'A'}(n) \le \frac{n}{2} log_2 n + 2n.$$

Переход от базиса B' к базису B и координат $\widetilde{x}'$ к $\widetilde{x}$. Переход получается посредством обратной перестановки к $\pi(i)$ (см. (2.3)), а его сложность оценивается, как и раньше, т.е. $L_{B'B}(n) = 0$.

В результате имеем

$$L_{AB}(n) = L_{AA'}(n) + L_{A'B'}(n) + L_{B'B}(n) \le \frac{n}{2} log_2 n + 3n.$$

Заметим, что при вычислении обратного преобразования $(F_n^{-1})^T$ при $n = 2^{k+1} - 1$ мы фактически получили матричное тождество

$$(F_n^T)^{-1} = \begin{pmatrix} (F_m^T)^{-1} & o_m & G_m \\ 0 \dots 0 & 1 & 0 \dots 0 \\ O_m & o_m & (F_m^T)^{-1} \end{pmatrix},$$

где $m = (n-1)/2$, а матрица $G_m = I_m(F_m^T)^{-1}$ есть симметричное отражение матрицы $(F_m^T)^{-1}$ относительно средней строки. Это тождество, также как и аналогичное тождество

$$F_n^{-1} = \begin{pmatrix} F_m^{-1} & o_m & O_m \\ 0 \dots 0 & 1 & 0 \dots 0 \\ G_m & o_m & F_m^{-1} \end{pmatrix},$$

(здесь $m = (n-1)/2$, а матрица $G_m = F_m^{-1} I_m$ есть симметричное отражение матрицы F_m^{-1} относительно среднего столбца) можно промерить и непосредственно. С помощью этих тождеств также как и в теореме 2.4.1 можно доказать, что плотность обратной матрицы $S(F_n^{-1}) = O(n^{\log_2 3})$ и такую же плотность имеет матрица перехода от нормального базиса второго или третьего типа к стандартному базису поля $GF(2^n)$.

2.4.5 О явном вычислении формул перехода и минимальных многочленов для оптимальных нормальных базисов

Рассмотрим общий случай $q = p^r$. Определим последовательность многочленов $f_i(x)$ над полем $GF(q)$ рекуррентным соотношением

$$f_{i+1}(x) = x f_i(x) - f_{i-1}(x)\,, f_1(x) = x\,, f_0 = 2.$$

По индукции легко доказать, что

$$f_i(\zeta + \zeta^{-1}) = \zeta^i + \zeta^{-i}$$

для любого элемента ζ.

Упражнение 2.4.2 Докажите это.

Как уже отмечалось, эти многочлены задают формулы перехода от стандартных базисов к нормальным в случае базисов второго и третьего типов.

Последовательность $f_i(x)$ можно формально продолжить и для отрицательных индексов, при этом, очевидно $f_{-i}(x) = f_i(x)$.

Указанная последовательность является некоторым аналогом последовательности многочленов Чебышева (их иногда называют в рассматриваемом случае многочленами Диксона), и также как и для них легко проверяется, что многочлены с четными номерами сами четные, т. е. содержат только одночлены с четными степенями, а многочлены с нечетными номерами – сами нечетные. Поэтому их удобно представлять в виде

$$f_i(x) = \sum_{j=0}^{\lfloor i/2 \rfloor} a_{i,j} (-1)^j x^{i-2j},$$

где коэффициенты будут указаны в лемме 2.4.2.

Для рассматриваемой последовательности справедливо тождество Чебышева

$$f_{ij}(x) = f_i(f_j(x)) = f_j(f_i(x)),$$

для проверки которого достаточно положить $x = \zeta + \zeta^{-1}$:

$$f_i(f_j(\zeta + \zeta^{-1})) = f_i(\zeta^j + \zeta^{-j})) = \zeta^{ij} + \zeta^{-ij} = f_{ij}(\zeta + \zeta^{-1}).$$

Легко проверить еще одно тождество Чебышева

$$f_{i+j}(x) + f_{i-j}(x) = f_i(x) f_j(x).$$

Действительно

$$\zeta^{i+j} + \zeta^{-i-j} + \zeta^{i-j} + \zeta^{-i+j} = (\zeta^i + \zeta^{-i})(\zeta^j + \zeta^j).$$

Лемма 2.4.2 *Справедливы следующие равенства (в которых все коэффициенты вычисляются по модулю p):*

1. $f_i(x) = \sum_{j=0}^{\lfloor i/2 \rfloor} a_{i,j}(-1)^j x^{i-2j}$, $a_{i,j} = \binom{i-j}{j} + \binom{i-j-1}{j-1}$,
2. $x^i = \sum_{j=0}^{\lfloor i/2 \rfloor} \binom{i}{j} f_{i-2j}$, *где для удобства здесь положим* $f_0 = 1$, *и биномиальные коэффициенты с отрицательными индексами считаем равными нулю.*

Доказательство. База индукции проверяется непосредственно. Для обоснования шага индукции ввиду тождества

$$f_{i+1}(x) = xf_i(x) - f_{i-1}(x)$$

достаточно проверить, что всегда $a_{i+1,j} = a_{i,j} + a_{i-1,j-1}$, а это вытекает из тождества Паскаля:

$$a_{i,j} + a_{i-1,j-1} = \binom{i-j}{j} + \binom{i-j-1}{j-1} + \binom{i-j}{j-1} + \binom{i-j-1}{j-2} =$$

$$= \binom{1+i-j}{j} + \binom{i-j}{j-1} = a_{i+1,j}.$$

Для доказательства второго тождества достаточно умножить обе его части на x и опять применить тождество Паскаля:

$$x^{i+1} = xx^i = \sum_{j=0}^{\lfloor i/2 \rfloor} \binom{i}{j} x f_{i-2j} = \sum_{j=0}^{\lfloor i/2 \rfloor} \binom{i}{j} (f_{1+i-2j} + f_{i-2j-1}) =$$

$$= \sum_{j=1}^{\lfloor i/2 \rfloor} (\binom{i}{j} + \binom{i}{j-1}) f_{i-2j-1} f_{i+1} = \sum_{j=0}^{\lfloor i/2 \rfloor} \binom{i+1}{j} f_{1+i-2j}.$$

Справедлива также

Теорема 2.4.4 *Пусть* $2n+1$ *– простое, и* $q = p^r$ *– примитивный корень по модулю* $2n+1$, *или* $2n+1 \equiv 3 \pmod 4$ *и* q *порождает все квадратичные вычеты по модулю* $2n+1$, *тогда многочлен* $g_n(x)$ *над* $GF(q)$, *определенный рекуррентной формулой*

$$g_0 = 1, \quad g_1(x) = x+1, \quad g_k(x) = xg_{k-1}(x) - g_{k-2}(x) \quad \text{для} \quad k \geq 2$$

совпадает с минимальным аннулирующим элемент α *многочленом* m_α, *и его корни образуют оптимальный нормальный базис второго или, соответственно, третьего типа поля* $GF(q^n)$. *Связь между многочленами* g_n *и* f_n *дается формулой*

$$g_n = 1 + \sum_{i=1}^{n} f_i = g_{n-1} + f_n.$$

Доказательство. По индукции можно проверить, что при $n \geq 1$

$$g_n = 1 + \sum_{i=1}^{n} f_i.$$

Действительно, при $n = 1$ имеем $g_1 = x + 1 = 1 + f_1$, при $n = 2$ имеем $g_2 = xg_1 - g_0 = x^2 + x - 1 = 1 + f_1 + f_2$. Шаг индукции обосновывается равенством

$$g_{n+1} = 1 + \sum_{i=1}^{n+1} f_i = 1 + f_1 + \sum_{i=2}^{n+1} xf_{i-1}(x) - f_{i-2}(x) =$$

$$= 1 + f_1 + x\sum_{i=1}^{n} f_i - \sum_{i=0}^{n-1} f_i =$$

$$x(1+\sum_{i=1}^{n} f_i) - 1 - \sum_{i=1}^{n-1} f_i = xg_n - g_{n-1}.$$

Теперь ясно, что

$$g_n(\alpha) = 1 + \sum_{i=1}^{n} f_i(\zeta + \zeta^{-1}) = 1 + \sum_{i=1}^{n}(\zeta^i + \zeta^{-i}) = \zeta^{-n}\sum_{i=0}^{2n}\zeta^i = \frac{\zeta^{-n}(\zeta^{2n+1}-1)}{\zeta - 1} = 0$$

так как $\zeta^{2n+1} = 1$. Минимальность аннулирующего α многочлена g_n следует из того, что его степень равна n, и в противном случае α было бы корнем неприводимого двоичного многочлена степени меньшей n, что противоречило бы доказанной ранее линейной независимости элементов

$$\{1, \alpha, \ldots, \alpha^{n-1}\}$$

над полем $GF(2)$.

В дополнение к этой теореме мы укажем, как можно явно вычислить коэффициенты минимального многочлена. Действительно, если

$$g_n(x) = \sum_{j=0}^{n} g_{n,j}x^j,$$

то согласно теореме 2.4.4 и лемме 2.4.2

$$g_n(x) = 1 + \sum_{i=1}^{n} f_i = 1 + \sum_{i=1}^{n}\sum_{j=0}^{\lfloor i/2\rfloor} a_{i,j}(-1)^j x^{i-2j},$$

откуда

$$g_{n,j} = \sum_{i=0}^{\lfloor (n-j)/2\rfloor} a_{2i+j,i}(-1)^i == \sum_{i=0}^{\lfloor (n-j)/2\rfloor} (-1)^i \left(\binom{i+j}{i} + \binom{i+j-1}{i-1}\right).$$

После очевидных сокращений остается только первое слагаемое:

$$g_{n,j} = (-1)^{\lfloor (n-j)/2\rfloor}\binom{\lfloor (n+j)/2\rfloor}{j}$$

(все вычисления проводятся по модулю p).

В случае $p = 2$ множитель $(-1)^{\lfloor (n-j)/2\rfloor}$, естественно, можно опустить.

В силу известной теоремы Люка (см., например, [25]) биномиальный коэффициент $\binom{a}{b}$ по модулю p можно вычислить за $O(\nu_p(b))$ операций в поле $GF(p)$ по формуле

$$\prod_{i=1}^{m}\binom{a_i}{b_i},$$

где $0 \le a_i, b_i < p, 1 \le i \le m$ – цифры $p-$ичного разложения a и b соответственно,

$$\nu_p(b) = \sum_{i=1}^{m} b_i.$$

Поэтому, последовательно вычисляя путем прибавления единицы p-ичные разложения $\lfloor (n+j)/2\rfloor$ и j и используя сделанное замечание, можно вычислить все коэффициенты многочлена $g_n(x)$ со сложностью $O(pn\log_p n)$

Такое же утверждение, естественно, справедливо и для сложности вычисления многочлена $f_n(x)$. Для вычисления всей последовательности многочленов $f_1(x), \ldots, f_n(x)$ или

$g_1(x), \ldots, g_n(x)$ естественно, конечно, использовать определяющую ее рекуррентную формулу с квадратичной оценкой сложности.

В случае $p = 2$ биномиальные коэффициенты вычисляются по модулю два и согласно теореме Куммера их можно вычислить следующим образом: $i-$й коэффициент равен 1 тогда и только тогда когда все двоичные цифры верхнего индекса, т.е. $\lfloor (n+i)/2 \rfloor$, не меньше соответствующих цифр нижнего индекса, т.е. i.

Применяя это правило, можно вычислить, что

$$g_{23} = 1 + x^4 + x^6 + x^7 + x^8 + x^{16} + x^{20} + x^{22} + x^{23},$$

и вообще при $n = 3 \cdot 2^k - 1$ получится $2k+3-$член

$$g_n = 1 + x^{2^{k-1}} + x^{2^{k-1}+2^{k-2}} + x^{2^{k-1}+2^{k-2}+2^{k-3}} +$$

$$+ \ldots + x^{2^k-1} + x^{2^k} + x^{2^{k+1}} + x^{2^{k+1}+2^{k-1}} + x^{2^{k+1}+2^{k-1}+2^{k-2}} + x^{2^{k+1}+2^{k-1}+2^{k-2}+2^{k-3}} + \ldots + x^{2^{k+1}+2^k-1} =$$

$$(1 + x^{2^{k+1}})h_k + x^{2^k}, h_k = 1 + x^{2^{k-1}} + x^{2^{k-1}+2^{k-2}} + x^{2^{k-1}+2^{k-2}+2^{k-3}} + \ldots + x^{2^k-1}$$

2.4.6 Оценка сложности перехода от оптимальных нормальных базисов второго и третьего типа к стандартным в общем случае

Пусть теперь $2n+1$ — простое число, а $q = p^r$ или примитивный корень по модулю $2n+1$ или $2n+1 \equiv 3 \pmod 4$ и q порождает все квадратичные вычеты по модулю $2n+1$. Пусть ζ — примитивный корень из единицы степени $2n+1$ в поле $GF(q^{2n})$, $\alpha_i = \zeta^i + \zeta^{-i}, \alpha = \alpha_1$, тогда, как отмечалось выше, $\{\alpha_1, \ldots, \alpha_n\}$ является оптимальным нормальным базисом (точнее, его перестановкой) в поле $GF(q^n)$, которое можно рассматривать как векторное пространство над полем $GF(q)$.

Рассмотрим в нем m-мерное ($m \leq n$) подпространство, порожденное базисом $\{\alpha_0, \ldots, \alpha_{m-1}\}$ или эквивалентным базисом $\{1, \alpha, \ldots, \alpha^{m-1}\}$.

Обозначим через $L(m)$ сложность преобразования координат от базиса $\{\alpha_0, \ldots, \alpha_{m-1}\}$ к базису $\{1, \alpha, \ldots, \alpha^{m-1}\}$ и обратно (под сложностью понимается число арифметических операций в поле $GF(q)$).

Теорема 2.4.5 *При* $m \leq n$

$$L(m) = O(pm \log_p m).$$

Обозначим $B(n)$ сложность перехода от оптимального нормального базиса второго или третьего типа к соответствующему стандартному или наоборот в поле $GF(q^n)$. Справедливо следующее обобщение теоремы 2.4.3.

Теорема 2.4.6

$$B(n) = O(pn \log_p n).$$

Так как с точки зрения практического применения случай $p > 2$ не представляет существенного интереса, обе эти теоремы мы оставим без доказательства, отсылая читателя к статье [14]

2.4.7 Пример выполнения алгоритма перехода от оптимального базиса 2-го или 3-го типа к стандартному и обратно.

Переход от оптимального нормального базиса второго или третьего типа к стандартному:

Пусть дано: $n = 5$, оптимальный нормальный базис второго типа $B = \{\alpha^{2^k}, k = 0,1,2,3,4\}$, элемент, записанный в этом базисе, $f = 1\cdot\alpha^{2^0} + 1\cdot\alpha^{2^1} + 0\cdot\alpha^{2^2} + 1\cdot\alpha^{2^3} + 1\cdot\alpha^{2^4}$, а т.е. и вектор координат $\widetilde{x}^T = \{1,1,0,1,1\}$.

Надо найти: $\widetilde{y}$ — вектор координат f в стандартном полиномиальном базисе $A = \{\alpha^0, \alpha^1, \alpha^2, \alpha^3, \alpha^4\}$

Шаг 1. *Переход от нормального базиса B к почти нормальному базису B' и координат $\widetilde{x}$ к $\widetilde{x}'$.*

а) В начале найдем перестановку $\pi(i)$ такую что

$$2^i (mod 11) = \pm\pi(i) \in \{1,\dots,5\}$$

т.е.

$$\pi = \begin{pmatrix} 2^0 & 2^1 & 2^2 & 2^3 & 2^4 & 2^5 & 2^6 & 2^7 & 2^8 & 2^9 & (mod 11) \\ 1 & 2 & 4 & -3 & 5 & -1 & -2 & -4 & 3 & -5 & (mod 11) \end{pmatrix}, \tag{5.6.1}$$

b) Пользуясь перестановкой 2.4.7, найдем базис $B' = \{\alpha_1, \alpha_2, \alpha_4, \alpha_3, \alpha_5\}$, т.е. элемент в этом базисе запишется так: $f = 1\cdot\alpha_1 + 1\cdot\alpha_2 + 0\cdot\alpha_4 + 1\cdot\alpha_3 + 1\cdot\alpha_5$, (т.е. поменяем местами 3-ю и 4-ю координаты и от координат $(\widetilde{x})^T = (1,1,0,1,1)$ перейдем к координатам $(\widetilde{x}')^T = (1,1,1,0,1)$).

Шаг 2. *Переход от базиса B' к базису B'' и координат $\widetilde{x}'$ к $\widetilde{x}''$.*

Т.к. $2^2 < 5 < 2^3$, то перейдем к $B'' = \{\alpha_k, k = 1,\dots,8\}$ путем добавления нулей, т.е. $\widetilde{x}'' = (\widetilde{x}', 0, 0, 0) = (\underbrace{1,1,1,0}_{2^2}, \underbrace{1,0,0,0}_{2^3-2^2})$.

Шаг 3. *Переход от базиса B'' к базису A'' и координат $\widetilde{x}''$ к $\widetilde{y}''$. $A'' = \{\alpha^k, k = 1,\dots,8\}$.*

По рекуррентной формуле 2.1 умножение матрицы $(F_8)^T$ на 8-мерный вектор $\widetilde{x}''$ сводится к двум умножениям 4-мерных векторов на матрицу $(F_4)^T$ т.е.

$$(\widetilde{y}'')^T = (F_8)^T \otimes \widetilde{x}'' = (F_4)^T \otimes \begin{pmatrix} x_1'' + x_7'' \\ x_2'' + x_6'' \\ x_3'' + x_5'' \\ x_4'' \end{pmatrix} + \alpha^{2^2} (F_4)^T \otimes \begin{pmatrix} x_5'' \\ x_6'' \\ x_7'' \\ x_8'' \end{pmatrix}, \tag{5.6.2}$$

каждое из которых, в свою очередь, сводится к двум умножениям 2-мерных векторов на матрицу $(F_2)^T$ т.е.

$$\widetilde{y}'' = (F_8)^T \otimes \widetilde{x}'' = (F_2)^T \otimes \begin{pmatrix} (x_1'' + x_7'') + (x_3'' + x_5'') \\ (x_2'' + x_6'') \end{pmatrix}, + \alpha^2 (F_2)^T \otimes \begin{pmatrix} x_3'' + x_5'' \\ x_4'' \end{pmatrix},$$

$$+ \alpha^4 (F_2)^T \otimes \begin{pmatrix} x_5'' + x_7 \\ x_6'' \end{pmatrix}, + \alpha^6 (F_2)^T \otimes \begin{pmatrix} x_7'' \\ x_8'' \end{pmatrix},$$

где (F_2) — единичная матрица, т.е. $(\widetilde{y}'') = (1,1,0,0,1,0,0,0)$.

Шаг 4. *Переход от базиса A'' к базису A' и координат $\widetilde{y}''$ к $\widetilde{y}'$. $A' = \{\alpha^k, k = 1,\dots,5\}$.*

А теперь удалим последние координаты: $\widetilde{y}' = (1,1,0,0,1)$

Шаг 5. *Переход от базиса A' к базису A и координат $\widetilde{y}'$ к $\widetilde{y}$. $A = \{\alpha^k, k = 0,\dots,4\}$.*

Построим минимальный аннулирующий многочлен m_α для генератора α. По рекуррентной формуле (5.4.1) получим:

$$\begin{aligned} f_0 &= 0 \\ f_1 &= x+1 \\ f_2 &= x^2+x \\ f_3 &= x^3+x^2+x+1 \\ f_4 &= x^4+x^3 \\ f_5 &= x^5+x^4+x^3+x^2+x+1 \end{aligned}$$

т.е.

$$m_\alpha := f_5(\alpha) = a_5\alpha^5 + a_4\alpha^4 + a_3\alpha^3 + a_2\alpha^2 + a_1\alpha^1 + a_0\alpha^0 = 0, \tag{5.6.3}$$

где $a_0 = a_1 = a_2 = a_3 = a_4 = a_5 = 1$. Применяя формулу 5.5.2 получим

$$\sum_{i=0}^{n-1} y_i\alpha^i = 1\cdot\alpha^0 + (1+1)\alpha^1 + (1+1)\alpha^2 + (0+1)\alpha^3 + (0+1)\alpha^4,$$

откуда и получаем вектор координат в стандартном базисе $\widetilde{y} = (1,0,0,1,1)$. Т.о. мы от представления многочлена в оптимальном нормальном базисе

$$f = 1 + \alpha^{2^1} + \alpha^{2^3} + \alpha^{2^4} \in B$$

перешли к представлению многочлена в стандартном базисе:

$$f = 1 + \alpha + \alpha^3 + \alpha^4 \in A.$$

Переход от стандартного базиса к оптимальному нормальному базису второго или третьего типа:

Пусть дано: $n = 5$, стандартный базис $A = \{\alpha^0, \alpha^1, \alpha^2, \alpha^3, \alpha^4\}$, и элемент в этом базисе $f = \sum_{i=0}^{4} y_i\alpha^i$, с координатами $\widetilde{y} = \{1,0,0,1,1\}$.

Надо найти: $\widetilde{x}$ — вектор координат f в оптимальном нормальном базисе 2-го или 3-го типа $B = \{\alpha^{2^0}, \alpha^{2^1}, \alpha^{2^2}, \alpha^{2^3}, \alpha^{2^4}\}$

Т.к. условия на n удовлетворяют определению оптимального базиса 2-го типа, значит можно применять теорему 2.4.3.

Шаг 1. *Переход от базиса A к базису A' и координат $\widetilde{y}$ к $\widetilde{y}'$.* $A' = \{\alpha^{2^1}, \alpha^{2^2}, \alpha^{2^3}, \alpha^{2^4}, \alpha^{2^5}\}$

Минимальный аннулирующий многочлен $m_\alpha = 0$ для $n = 5$ уже нами построен (т.е. a_i и найдены) (см. 2.4.4), подставляя a_i в ?? имеем

$$\sum_{i=1}^{n} y_i\alpha^i = (0+1)\alpha^1 + (0+1)\alpha^2 + (1+1)\alpha^3 + (1+1)\alpha^4 + 1\cdot\alpha^5,$$

откуда и получается вектор координат в почти стандартном базисе A' $\widetilde{y} = (1,1,0,0,1)$.

Шаг 2. *Переход от базиса A' к системе A'' и координат $\widetilde{y}'$ к $\widetilde{y}''$.*

Переход осуществляется путем формального добавления нулей в конец вектора $\widetilde{y}'$: $(\widetilde{y}'')^T = (1,1,1,0,1,0,0,0)$.

Шаг 3. *Переход от системы A'' к системе B'' и координат $\widetilde{y}''$ к координатам $\widetilde{x}''$.*

Умножение матрицы $(F_8^T)^{-1}$ на 8-мерный вектор $\widetilde{y}''$ сводится к двум умножениям 4-мерных векторов на матрицу $(F_4^T)^{-1}$ и «перепутанному» их сложению т.е.:

$$(\widetilde{x}'')^T = (\widetilde{X}_1, \widetilde{X}_2)^T = (F_8^T)^{-1} \otimes \widetilde{y}'' = (F_8^T)^{-1} \otimes (\widetilde{Y}_1, \widetilde{Y}_2)$$

(где вектор $\widetilde{Y}_1 = (y_1'', y_2'', y_3'', y_4'')^T$, а вектор $\widetilde{Y}_2 = (y_5'', y_6'', y_7'', y_8'')^T$) сводится к

$$(\widetilde{y}'') = (v_1 + w_3, v_2 + w_2, v_3 + w_1, v_4, w_1, w_2, w_3, w_4),$$

где вектора $\widetilde{v}$ и $\widetilde{w}$ считаются по формулам:

$$(v_1, v_2, v_3, v_4) = (F_4^T)^{-1} \otimes (\widetilde{X}_1) \tag{5.6.4}$$

и

$$(w_1, w_2, w_3, w_4) = (F_4^T)^{-1} \otimes (\widetilde{X}_2) \tag{5.6.5}$$

Каждая из формул 2.4.7, 2.4.7, в свою очередь рекуррентно сводится тоже к двум умножениям 2-мерных векторов на матрицу $(F_2^T)^{-1} \equiv E$ (единичная матрица 2x2) и их «перепутанному» сложению,т.е.

$$(v_1, v_2, v_3, v_4) = (p_1 + r_1, p_2, r_1, r_2),$$
$$(w_1, w_2, w_3, w_4) = (s_1 + t_1, s_2, t_1, t_2),$$

где

$$(p_1, p_2) = E \otimes (v_1, v_2)^T,$$
$$(r_1, r_2) = E \otimes (v_3, v_4)^T,$$
$$(s_1, s_2) = E \otimes (w_1, w_2)^T,$$
$$(t_1, t_2) = E \otimes (w_3, w_4)^T,$$

где $(\widetilde{y}'') = (p_1, p_2, r_1, r_2, s_1, s_2, t_1, t_2)$, т.е. $p_1 = 1$, $p_2 = 1$, $r_1 = 0$, $r_2 = 0$, $s_1 = 1$, $s_2 = 0$, $t_1 = 0$, $t_2 = 0$. Поднимаясь рекурсивно вверх получим, что$(\widetilde{x}'') = (1, 1, 1, 0, 1)$.

Шаг 4. *Переход от системы B'' к базису B' и координат $\widetilde{x}''$ к координатам $\widetilde{x}'$.*

Удалим последние координаты: $(\widetilde{x}')^T = \{1, 1, 1, 0, 1\}$

Шаг 5. *Переход от базиса B' к базису B и координат $\widetilde{x}'$ к координатам $\widetilde{x}$.*

Выполняя обратную перестановку к перестановке π 2.4.7 получим $\widetilde{x} = (1, 1, 0, 1, 1)$.

Таким образом от представления многочлена в стандартном базисе

$$f = 1 + \alpha^3 + \alpha^4$$

мы перешли к представлению многочлена в оптимальном нормальном базисе 2-го типа:

$$f = 1 + \alpha^{2^1} + \alpha^{2^3} + \alpha^{2^4}.$$

2.4.8 Замечание о программной имплементации

Разумеется, при рекурсивном выполнении алгоритма перехода от нормального базиса к стандартному и наоборот, рекурсию можно остановить в момент появления матриц размерности $n \leq 32$, так как представляя столбцы этих матриц в виде 32-битных целых чисел, умножение матрицы на вектор можно свести к n операциям умножения числа на бит и $n - 1$ операции XOR (побитового сложения чисел), или даже иногда меньшего количества операций XOR, если их выполнять только для тех чисел-столбцов матрицы, которые соответствуют единичным компонентам данного вектора. Чтобы это стало возможным, надо «битовые» столбцы первоначальной матрицы тоже представлять в виде столбцов из 32-битных чисел, и весь рекурсивный алгоритм оформить, ориентировав на работу не с битами, а 32-битными целыми. В результате и используемая память и время работы уменьшатся примерно в 32 раза.

При умножении 32-*мерной* битовой матрицы на 32 мерный вектор можно вместо стандартного способа применить модификацию «алгоритма четырех русских,» разбив матрицы на три подматрицы ширины 11 каждая и предварительно загрузив в память машины три таблицы, содержащих результаты умножения каждой из этих матриц на любой 11- битный вектор. Тогда результат умножения данной матрицы на произвольный 32- битный вектор можно будет получить с помощью трех операций извлечения 32-битных из соответствующих массивов и двух операций побитового сложения этих чисел. Общий объем используемых массивов примерно равен 24 *Кб,* что позволяет использовать для их хранения кэш процессора и

ускоряет доступ к этим массивам. За счет этого приема достигается дополнительное ускорение примерно в шесть раз. Если размеры кэша позволяют, можно использовать разбиение на две подматрицы ширины 16 каждая.

В первоначальном варианте алгоритма перехода вектор X представлялся как n-мерный булевский вектор и в ходе алгоритма к нему применялись операции разбиения на две равные части, перестановки координат и покоординатного сложения по модулю два. Используемая перестановка координат является композицией инверсии (перестановки, в которой первая координата ставится на последнее место, вторая на предпоследнее и т.д.) и сдвига на одну координату. Скорость работы алгоритма можно увеличить, если n-мерный булевский вектор X представить как $\lceil n/32 \rceil$ вектор, состоящий из 32-битных целых чисел. Тогда покоординатное сложение по модулю два заменяется на покомпонентное выполнение операции побитового XOR с целыми числами. Операция сдвига координат n-мерного булевского вектора сводится к выполнению такого же сдвига битов каждого из 32-битовых чисел и переноса одного бита из каждого числа в соседнее. Поэтому скорость ее выполнения существенно выше при использовании не битового, а словарного (32-битового) представления данного вектора. То же самое верно и для операции выполнения инверсной перестановки. Она сводится к инверсной перестановке массива из $\lceil n/32 \rceil$ 32-битных чисел и инверсной перестановке битов каждого числа из этого массива. Для выполнения инверсной перестановки битов разбиваем данное 32-битное число на два 16 битных числа, делаем в каждом из них подобную перестановку, используя заранее загруженную в память таблицу объемом 128 *Кб,* и выполняем еще три операции с 32-битными числами, чтобы получить окончательный результат.

Для выполнения умножения в стандартном базисе можно применить алгоритмы, указанные в соответствующих разделах нашей книги ////// вставить ссылку!!!!

2.4.9 О сложности арифметических операций в конечных полях

Пусть конечное поле $GF(q^n)$ представлено стандартным базисом B_α, порожденным корнем α неприводимого над $GF(q)$ многочлена $g(x)$ степени n. Сложность операции умножения многочленов степени $n-1$ над полем $GF(q)$ обозначим $M_q(n)$. Согласно [147], $M_q(n) = O(n \log n \log\log n)$.

Обозначим $M^s_{q,g(n)}$ сложность операции умножения в указанном представлении поля $GF(q^n)$. Так как умножение в рассматриваемом случае сводится к умножению многочленов над полем $GF(q)$ и последующему делению результата на многочлен $g(x)$ с остатком, то независимо от выбора $g(x)$ известна оценка $M^s_{q,g}(n) = 3M_q(n) + O(n)$, если пренебречь сложностью предварительного вычисления некоторого многочлена $f(x)$, зависящего только от $g(x)$. Далее иногда будем опускать индексы в формуле $M^s_{q,g}(n)$.

В некоторых случаях эту оценку можно улучшить. Действительно, если $g(x)$ содержит k одночленов, то сложность редукции по модулю $g(x)$ оценивается как kn и поэтому имеем $M^s_{q,g}(n) \le M_q(n) + (2k+1)n$, а в случае $q = 2$ $M^s_{q,g}(n) \le M_q(n) + kn$.

С высокой вероятностью в качестве k можно взять 3, а если неприводимого трехчлена степени n не существует, то, как проверено экспериментально, но не доказано теоретически, всегда можно взять $k = 5$.

В указанных выше оценках сложность аддитивных операций и умножения в поле $GF(q)$ принимается за единицу. В случае $q = 2$ это естественно. Если же $q = p^k$, где p – простое число, то естественно также оценить битовую сложность операции умножения в поле $GF(q)$ как

$$M(GF(q)) \le M(GF(p))M^s_q(k),\ M(GF(p)) \le 3M(\log_2 p) + O(\log_2 p),$$

где $M(m) = O(m \log m \log\log m)$ – битовая сложность умножения m-разрядных двоичных чисел. Отметим еще, что для p, являющихся простыми Мерсенна или Ферма, справедлива

оценка

$$M(GF(p)) \le M(\log_2 p) + O(\log_2 p).$$

Битовая сложность операции умножения в поле $GF(q^n)$ (при условии задания элементов поля их координатами в данном базисе B над полем $GF(q)$) оценивается

$$M(GF(q^n)) \le M(GF(q))M^s_{q,g}(n).$$

Возведение в степень q элемента поля $GF(q^n)$ производится путем вставки нулей в последовательность коэффициентов полинома, представляющего этот элемент, после чего производится редукция по модулю g, в результате (с учетом сделанного выше замечания) имеем (теоретически не доказанную) оценку сложности возведения в степень $K_q(n) = O(n)$.

Оценим сложность умножения в поле $GF(q^n)$ в случае использования оптимального нормального базиса первого типа и операций из поля $GF(q)$. Для выполнения умножения переходим к стандартному базису согласно теореме 2.4.1 со (схемной) сложностью $2n-2$, затем делаем умножение в стандартном базисе и переходим обратно к нормальному базису со сложностью $n-1$.

Согласно сделанному после теоремы 2.4.1 замечанию, получаем оценку

$$M^{O1}(GF(q^n)) \le M^s_q(n) + 7n - 8.$$

В случае базисов второго и третьего типа аналогично получаем оценку

$$M^{O2}(GF(q^n)) \le 3M^s_q(n) + \frac{3n}{2} log_2 n + O(n),$$

которая оказывается асимптотически в три раза хуже, так как в соответствующих стандартных базисах порождающий их неприводимый многочлен (вычисленный в теореме 2.4.4) оказывается нетривиальным, и для редукции по его модулю приходится использовать для оценки $M_g(n)$ указанную выше общую оценку. Можно, однако, привести примеры, когда в базисах второго и третьего типа сложность будет почти такой же, как и в стандартных базисах.

Далее через $M^O(GF(q^n))$ обозначаем минимальную сложность умножения в поле $GF(q^n)$ при условии задании элементов поля в некотором нормальном базисе.

2.4.10 Об оценках сложности возведения в степень и инвертирования в конечных полях

Ранее были получены такие оценки для стандартных базисов. Оценим сложность инвертирования в нормальных базисах. Возведение в степень q выполняется бесплатно, поэтому

$$I^O(GF(q^n)) \le l(n-1)M^O(GF(q^n)) + I(q) + (n + C_B)M(GF(q)),$$

где C_B — сложность используемого нормального базиса B. Напомним, что для инвертирования в поле $GF(q^n)$ можно вычислить $y = x^{(q^n-1)/(q-1)} \in GF(q)$ и применить формулу

$$x^{-1} = (x^{(q^{n-1}-1)/(q-1)})^q y^{-1}.$$

Вычисление y^{-1} по определению выполняется со сложностью $I(q)$, а умножение на него в поле $GF(q^n)$ выполняется со сложностью $nM(GF(q))$. Для вычисления $x^{(q^{n-1}-1)/(q-1)}$ используется $l(n-1)$ умножений в поле $GF(q^n)$, а для вычисления y нужна еще одна операция умножения, в которой вычисляется только одна координата произведения, и поэтому она выполняется со сложностью $C_B M(GF(q))$.

Аналогично, сложность возведения в произвольную степень $d < q^n$ в оптимальных нормальных базисах оценивается как

$$O\left(\frac{M^O(GF(q^n))\log d}{\log\log d}\right).$$

Эти оценки будут лучше полученных ранее оценок для стандартного базиса, если в поле $GF(q^n)$ выбрать нормальный базис, допускающий быстрое умножение, например такой, что $M^O(GF(q^n))) = o(n^{(\omega+1)/2})$. Выше было показано, что такими являются все оптимальные нормальные базисы. Другие примеры таких базисов приводятся далее.

2.5 Гауссовы нормальные базисы

2.5.1 О гауссовых нормальных базисах

В [80] были приведены примеры нормальных базисов B, у которых функция сложности C_B является линейной. Стандартный алгоритм умножения для таких базисов имеет квадратичную оценку сложности. Эти базисы B^α, получившие название *гауссовых нормальных базисов* (GNB), порождаются в полях $GF(q^n)$ элементами вида

$$\alpha = \zeta + \zeta^\gamma + \ldots + \zeta^{\gamma^{k-1}},$$

где $p = kn + 1$ — простое число, (в этом случае базис называется *базисом k-го типа*) ζ — примитивный корень p-й степени из единицы в поле $GF(q^{kn})$, являющемся расширением поля $GF(q^n)$, γ — примитивный корень k-ой степени из единицы в кольце вычетов порядка p, который вместе с q порождает всю мультипликативную группу ненулевых вычетов по модулю p.

Необходимым и достаточным условием существования GNB k-го типа, кроме простоты числа $p = kn+1$, не делящего q, является взаимная простота чисел kn/d и n, где d — порядок подгруппы, порожденной элементом q в мультипликативной группе. Условие не делимости q на p очевидно необходимо, чтобы q было отлично от нуля по модулю p. Оно очевидно выполняется при $q = 2$.

Заметим, что из условия $(nk/d, n) = 1$ следует, что d кратно n. Действительно, если в разложение числа n простое число p входит в степени α, то в разложение nk оно входит в не меньшей степени, а так как nk/d не кратно p, значит в разложение n простое p также входит в степени, не меньшей α. Так как $d = rn$, то условие $(kn/d, n) = 1$ можно переписать в виде $(k/r, n) = 1$.

Для доказательства указанного необходимого и достаточного условия заметим, что в мультипликативной группе найдется такой примитивный элемент β, что $q = \beta^{kn/d}$. Так как γ — примитивный корень k-ой степени из единицы в мультипликативной группе, то он представим виде $\gamma = \beta^{sn}, (s, n) = 1, 1 \le s \le k$. Тогда любой произведение вида $q^i\gamma^j \bmod p$ представимо в виде

$$\beta^{ikn/d+jsn \bmod kn}.$$

Так как любой элемент мультипликативной группы однозначно представим в виде $\beta^m, 0 \le m < kn$, то при $(kn/d, n) = r > 1$ только элементы вида β^{rj} могут быть представимы в виде произведения $q^i\gamma^j \bmod p$ что и доказывает необходимость условия $(n, nk/d) = 1$.

Если же kn/d и n взаимно просты, то мультипликативная группа порождается элементами q, γ, более того, любой ее элемент от 1 до $p - 1 = kn$ *представим единственным образом в виде* $q^i\gamma^j \bmod p, 0 \le j < k, 0 \le i < n$. Действительно, согласно китайской теореме об остатках любой вычет по модулю kn^2/d однозначно представим в виде

$$ikn/d + jn \bmod kn^2/d, , 0 \le j < kn/d, 0 \le i < n,$$

поэтому при $d = n$, выбирая $\gamma = \beta^n$, можно представить любой элемент мультипликативной группы в виде произведения $q^i\gamma^j \bmod p, 0 \le j < k, 0 \le i < n$. При $d = kn$ очевидно и без китайской теоремы, что любое число от 0 до $kn - 1$ представимо в виде

$$ikn/d + jn, 0 \le j < k, 0 \le i < n.$$

При $n < d < kn$ согласно китайской теореме для любого $x, 0 \le x < kn^2/d$ либо $x = ikn/d+jn$, либо $x+kn^2/d = ikn/d+jn$, где $0 \le j < kn/d\,, 0 \le i < n$. В первом случае любое $x+sn, 0 \le x < kn^2/d, 0 \le s \le k-kn/d$ представимо в виде $ikn/d+jn+sn = ikn/d+(j+s)n, 0 \le i < n, j+s < k$. Во втором случае любое $x + sn, 0 \le x < kn^2/d, kn/d \le s \le k - kn/d$ представимо в том же виде, а любое $x, 0 \le x < kn^2/d$ представимо в виде

$$ikn/d + (j - kn/d)n = ikn/d + (k + j - kn/d)n \bmod kn, 0 \le i < n,$$

$$0 \le j < k + j - kn/d < k.$$

Поэтому во всех возможных случаях можно представить любой элемент мультипликативной группы в виде произведения $q^i\gamma^j \bmod p\,, 0 \le j < k\,, 0 \le i < n$. Однозначность этого представления следует из равенства числа таких произведений и числа элементов в группе.

Заметим еще, что при $d = n$ взаимно простом с k условие существования GNB выполнено и более того, согласно известной теореме теории абелевых групп мультипликативная группа поля $GF(p)$ является произведением групп, порожденных элементами γ и q.

Упражнение 2.5.1 Докажите это.

Для проверки условия $(n, nk/d) = 1$ нет необходимости вычислять d, а достаточно проверить, что для любого простого делителя $l|n$ неверно, что $q^{nk/l} = 1 \bmod p$.

Упражнение 2.5.2 Докажите это.

В [161] доказано, что для $q = p^m$ в поле $GF(q^n)$ сущестует GNB какого-нибудь типа если и только если $(n, m) = 1$, n не делится на $2p$ при $p = 4l + 1$, и n не делится на $4p$ при $p = 4l + 2, 4l + 3$. Докажем это в особо интересующем нас случае $q = 2$. Действительно, согласно квадратичному закону взаимности $q = 2$ будет квадратичным вычетом по модулю p при $p - 1 = kn$ кратном 8. Поэтому согласно критерию Эйлера $q^{kn/2} = q^{(p-1)/2} = 1 \bmod p$, значит согласно утверждению 2.5.2 GNB k-го типа не существует в рассматриваемом случае т.е. когда n кратно 8 при любом k, или когда n кратно 4 и k четно, или когда n четно, а k кратно 4.

Докажем, что GNB являются базисами низкой сложности и укажем формулу умножения в этих базисах. Вычислим сложность матрицы T в схеме умножения Месси-Омура для GNB, порожденного элементом

$$\alpha = \zeta + \zeta^{\gamma} + \ldots + \zeta^{\gamma^{k-1}}, \gamma^k = 1 \bmod p, \gamma^i \ne 1 \bmod p, i = 1, \ldots, k-1, \zeta^p = 1.$$

Тогда

$$\alpha\alpha^{q^i} = \left(\zeta + \zeta^{\gamma} + \ldots + \zeta^{\gamma^{k-1}}\right)\left(\zeta^{q^i} + \zeta^{\gamma q^i} + \ldots + \zeta^{\gamma^{k-1}q^i}\right) = \sum_{j,l=0}^{k-1} \zeta^{\gamma^j q^i + \gamma^l}.$$

Если $\gamma^s q^i + 1 \ne 0 \bmod p, s < k, i < n$ то согласно определению GNB очевидно $\gamma^s q^i + 1 = \gamma^{a(s,i)} q^{b(s,i)}, s < k, i < n$, где $a(s,i), b(s,i)$ некоторые функции с аргументами $0 \le s < k, 0 \le i < n$, и значениями во множествах $\{0, \ldots, k-1\}$, $\{0, \ldots, n-1\}$ соответственно, значит $\gamma^j q^i + \gamma^l = \gamma^{l+a(s,i)} q^{b(s,i)}, s = j - l \bmod k$. Равенство $\gamma^j q^i + \gamma^l = 0 \bmod p$ возможно если и только если $\gamma^s q^i = -1 \bmod p$ при $s = j - l \bmod k$. Согласно определению GNB такая пара s, i существует и единственна.

Чтобы получить о ней некоторую информацию, возведем обе части равенства в степень $2k$, откуда имеем $q^{2ik} = 1, i < n$ что возможно только при $2ik$ кратном d, так как последовательность $q^j \bmod p$ имеет минимальный период $d = nr$. Тогда $2ik/d = 2i(k/r)/n$ целое число, а так как k/r и n взаимно просты, то $2i$ кратно n, следовательно $i = 0$ или $i = n/2$, а при нечетном n всегда $i = 0$. Но $i = 0$ возможно только при k четном и $s = k/2$, потому

что только тогда $\gamma^s = -1 \bmod p$. Обратно, если равенство $\gamma^s q^i = -1 \bmod p$ выполняется при $i = 0$, то k четно и $s = k/2$. Поэтому в силу взаимной однозначности представления элементов мультипликативной группы в виде $q^i\gamma^j \bmod p, 0 \le j < k, 0 \le i < n$ при четном k равенство $\gamma^s q^i = -1 \bmod p$ возможно, если и только если $s = k/2, i = 0$.

Покажем еще что нечетном k равенство $\gamma^s q^i = -1 \bmod p, i < n$ возможно лишь при четном $n, i = n/2$, s кратном k/r, где $d = rn$. Достаточно это сделать при четном n и $i = n/2$. Так как r нечетно, то из равенства $\gamma^s q^{n/2} = -1 \bmod p$ в силу $q^{d/2} = -1 \bmod p$ имеем $\gamma^s = q^{(d-n)/2} \bmod p$, откуда $\gamma^{sr} = q^{r(r-1)n/2} = q^{d(r-1)/2} = 1 \bmod p$, следовательно sr кратно k, значит s кратно k/r.

Объединяя слагаемые вида $\zeta^{\gamma^j q^i + \gamma^l}, t = j - l \bmod k$ при фиксированном t в одну сумму, имеем при $i = 0$ и четном k

$$\alpha\alpha^{q^i} = \sum_{j,l=0}^{k-1} \zeta^{\gamma^j q^i + \gamma^l} = \sum_{t \ne k/2} \sum_{l=0}^{k-1} \zeta^{(\gamma^t q^i + 1)\gamma^l} + \sum_{l=0}^{k-1} \zeta^{(\gamma^{k/2} q^i + 1)\gamma^l} =$$

$$\sum_{t \ne k/2} \sum_{l=0}^{k-1} \zeta^{(\gamma^t q^i + 1)\gamma^l} + k = \sum_{t \ne k/2} \sum_{l=0}^{k-1} \zeta^{\gamma^{l + a(t,i) \bmod k} q^{b(t,i)}} + k,$$

откуда, учитывая что

$$\alpha^{q^i} = \sum_{l=0}^{k-1} \zeta^{\gamma^l q^i},$$

получаем окончательно $i = 0$ и четном k

$$\alpha\alpha^{q^i} = k + \sum_{t \ne k/2} \sum_{j=0}^{k-1} \zeta^{\gamma^j q^{b(t,i)}} = k + \sum_{j \ne k/2} \alpha^{q^{b(j,i)}}.$$

В оставшемся случае аналогично находим, что

$$\alpha\alpha^{q^i} = \sum_{j=0}^{k-1} \alpha^{q^{b(j,i)}}.$$

При нечетном k также аналогично имеем для $i = n/2$

$$\alpha\alpha^{q^i} = k + \sum_{j \ne s} \alpha^{q^{b(j,i)}},$$

где s — то единственное число, для которого $\gamma^s q^{n/2} = -1 \bmod p$, и для $i \ne n/2$

$$\alpha\alpha^{q^i} = \sum_{j=0}^{k-1} \alpha^{q^{b(j,i)}}.$$

Далее рассмотрим для простоты случай четного q, т.е. $q = 2^m$. Тогда слагаемое k в поле $GF(q)$ равно $k \bmod 2$, т.е. равно нулю при нечетном k и равно 1 при четном k. Минус единицу в этом (при четном q просто единицу) поле можно представить как сумму всех элементов базиса (аналогично тому как это делалось при построении ONB первого типа). Так как i-я строка матрицы T равна по определению вектору коэффициентов разложения произведения $\alpha\alpha^{q^i}$ по элементам базиса, то число C ненулевых элементов в этой матрице (на самом деле число единиц) при нечетном k не больше

$$k(n-1) + k - 1 + n = (k+1)n - 1,$$

при четном k не больше

$$k(n-1)+k-1=kn-1.$$

В [?] со ссылкой на работы Beth, Geiselmann, Meyer даны для $q=2$ следующие нижние оценки для сложности GNB k-го типа: $C \geq kn-(k^2-3k+3)$, если k четно, и $C \geq (k+1)n-(k^2+k+1)$, если k нечетно, причем при $k=3,4$ всегда $C=4n-7$, при $k=5, n>2$ и $k=6, n>3$ всегда $C=6n-21$, при $k=7, n>4$ всегда $C=8n-43$.

Отметим еще, что в [83],[77], опираясь на расширенную гипотезу Римана показано, что для всех простых p и n, не кратных p, существуют в поле $GF(p^n)$ GNB типа $k=O(n^3\log^2 np)$.

2.5.2 Оценки сложности умножения в гауссовых нормальных базисах

Точно также, как и в доказательстве оптимальности базисов первого типа, можно проверить, что при мультипликативном порядке q равном $d=nk$ минимальным аннулирующим элемент ζ многочленом m_ζ над полем $GF(q)$ является круговой многочлен

$$F_{p-1}=x^{p-1}+\ldots+1,$$

и поле $GF(q^{kn})$ порождается стандартным базисом

$$B_\zeta=\{1,\zeta,\ldots,\zeta^{kn-1}\}.$$

Упражнение 2.5.3 Докажите это.

Упражнение 2.5.4 Докажите, также как и в случае ONB первого типа, что при $d=kn$ построенные нормальные базисы действительно линейно независимы. В общем случае это будет доказано ниже.

Так как в случае $d=nk$ элементы базиса B^α имеют вид

$$\alpha^{q^i}=\zeta^{q^i}+\zeta^{\gamma q^i}+\ldots+\zeta^{\gamma^{k-1}q^i},$$

т.е. являются суммами разных элементов почти стандартного базиса

$$B'_\zeta=\{\zeta,\ldots,\zeta^{p-1}\},$$

то матрица перехода от координат в базисе B^α к координатам в базисе B'_ζ имеет по одной единице в каждой строке и поэтому соответствующее преобразование имеет нулевую схемную сложность (и линейную сложность в случае программной имплементации). Обратное преобразование (определенное, конечно, только на соответствующем подпространстве) также по тем же причинам имеет нулевую схемную сложность. Сложность прямого и обратного перехода от стандартного базиса B_ζ к почти стандартному базису B'_ζ над полем $GF(q)$ была фактически оценена в доказательстве теоремы 2.4.1 и равна в рассматриваемом случае $kn-1$.

Согласно сделанному после теоремы 2.4.1 замечанию, получаем оценку сложности умножения в базисе B_ζ

$$M^s_{q,F_{kn}}(kn)\leq(M^s_q(kn)+4kn-5),$$

откуда имеем оценку битовой сложности умножения в нормальном базисе B^α

$$M^O(GF(q^n))\leq(M^s_q(kn)+7kn-8)M(GF(q)). \tag{2.5}$$

Применяя оценку Шенхаге для умножения многочленов, имеем для GNB k-го типа

$$M((GF(q^n)))=O(M(GF(q))(nk\log nk\log\log nk)).$$

Используя метод инвертирования в нормальном базисе, отсюда имеем

$$I((GF(q^n))) = O(logqn)M((GF(q^n))) = O(M(GF(q))(nk\log qn\log nk\log\log nk)).$$

Аналогично, сложность возведения в произвольную степень $d < q^n$ оценивается как

$$O\left(\frac{M(GF(q^n))\log d}{\log\log d}\right) = O(M(GF(q))\left(1+\frac{\log k}{\log n}\right)\log q(n^2k\log\log nk)).$$

В случае $d < kn$ эти утверждения будут доказаны ниже.

В случае $k = 1, 2$ рассматриваемые базисы являются оптимальными базисами первого и второго типов и при $k = 1$ полученная оценка превращается в оценку, приведенную в предыдущем пункте. Отметим, что идея использования в этом случае перехода к стандартному базису B_ζ появилась в 1997 г. в сообщении D.Dahm, опубликованном в Интернете. При $k = 2$ указанная оценка асимптотически несколько лучше оценки

$$M^{O2}(GF(q^n)) \le (3M_q^s(n) + \frac{3n}{2}log_2n + O(n))M(GF(q)),$$

полученной выше для базисов как второго, так и третьего типов, так как $M_q^s(2n)$ асимптотически меньше $3M_q^s(n)$, но практически вопрос их сравнения не так очевиден. При больших k эффективность оценки 2.5 снижается, но при $k = O(n^{1-\epsilon})$ она еще остается бесконечно малой в сравнении с квадратичной оценкой.

В [100] случае $d < kn$ получены такие же, как и выше, оценки сложности умножения, инвертирования и экспоненцирования в GNB. Отличие от приведенных выше оценок только в оценке для инвертирования, которое проводилось после перехода к стандартному базису в расширенном поле $GF(q^{kn})$ с помощью быстрого варианта расширенного алгоритма Евклида и поэтому оценка его сложности имела вид

$$I((GF(q^n))) = O(M(GF(q))(nk\log^2 nk\log\log nk)).$$

Очевидно, что при $q < k$ эта оценка формально чуть хуже, а при $q > k$ — чуть лучше оценки, приведенной нами выше. Отметим все же, что мультипликативный множитель в первой оценке меньше, и ее можно применять не только к программной имплементации, но и к построению логических схем. Заметим еще, что в тех случаях, когда k ограничено, его в упомянутых оценках можно отбросить. Так как в [104] доказано, что для любых q, k существует бесконечно много n, для которых в поле $GF(q^n)$ существует GNB типа k, то в [100] k сделан вывод, что для бесконечно многих n справедливы оценки

$$M((GF(q^n))) = O(M(GF(q))(n\log n\log\log n)),$$

$$I((GF(q^n))) = O(M(GF(q))(n\log^2 n\log\log n)),$$

$$E(M(GF(q^n)), d) = O(M(GF(q))\log q(n^2\log\log n)),$$

последняя из которых относится к общему экспоненцированию.

2.5.3 Еще о гауссовых нормальных базисах

Укажем, опираясь, на [100], еще один вариант изложения теории GNB, более элегантный, но несколько менее элементарный. Обозначим K порожденную элементом γ подгруппу порядка k в мультипликативной группе поля $GF(p)$, $p = kn+1$ и не делит q. Пусть $K_i, i = 0, \ldots, n-1$ смежные классы по подгруппе K и $\zeta \in GF(q^{kn}), \zeta^p = 1$. Последовательность

$$\alpha_i = \sum_{a\in K_i}\zeta^a, i = 0, \ldots, n-1$$

называется { гауссовым периодом} типа (n,k). Они появились в [106] для решения задачи о построении правильного 17-угольника. В [161] было доказано, что гауссов период типа (n,k) в поле $GF(q^n)$ образует нормальный базис если и только если $(n,nk/d)=1$, где d — мультипликативный порядок q в поле $GF(p)$. Действительно, циклические подгруппы, порожденные γ и q, имеют индексы $kn/k=n$ и kn/d в мультплипликативной группе поля $GF(p)$, поэтому γ и q порождают ее тогда и только тогда, когда $(n,nk/d)=1$. Если они ее порождают, то гауссов период есть нормальный базис, если они ее не порождают, то они порождают ее собственную подгруппу, и тогда можно доказать, что гауссов период не образует нормального базиса. Действительно, без ограничения общности можно начать последовательность K_i с классов вида $K_i=Kq^i, i=0,\ldots$. Если γ и q порождают мультплипликативную группу поля $GF(p)$, то в силу периодичности все различные классы имеют вид $K_i=Kq^i, i=0,\ldots,m-1$, где m делитель d, а так как их ровно n, то $m=n$. Тогда

$$\alpha_i=\sum_{a\in K_i}\zeta^a=\sum_{a\in K}\zeta^{q^i a}=\left(\sum_{a\in K_i}\zeta^a\right)^{q^i}=\alpha_0^{q^i}.$$

Так как $\alpha_0^{q^n}=\alpha_0$, то $\alpha_0\in GF(q^n)$, и $\alpha_i\in GF(q^n), i=1,\ldots,n-1$. Остается доказать линейную независимость системы $\{\alpha_i, i=0,\ldots,n-1,\}$ что выше было сделано только в предположении, что $d=kn$. В общем случае $(\zeta^a)^{q^d}=\zeta^{aq^d \bmod p}=\zeta^a$, значит система $\{\zeta^a, a=1,\ldots,kn\}$ лежит в поле $GF(q^d)$ и при $d<kn$ не образует в нем базиса. Круговой многочлен

$$f_{kn}=1+x+\ldots+x^{kn},$$

корнями которого являются элементы этой системы, в случае $d<kn$ уже не будет неприводимым, но разлагается в произведение kn/d неприводимых многочленов степени d

$$f_{kn}=\prod_{a\in L}m_{\zeta^a},$$

где $L\subset K$, m_{ζ^a} — минимальный аннулирующий многочлен элемента ζ^a. Действительно, у любого m_{ζ^a} корнями будут элементы $\{\zeta^b, b\in aQ,\}$ где Q подгруппа мультипликативной группы поля $GF(p)$, порожденная q. По условию ее порядок равен d. Можно проверить, что $m_{\zeta^a}=m_{\zeta^b}$ если и только если $b\in aQ$, т.е. b и a принадлежат одному и тому же смежному классу по подгруппе Q. Пусть список этих классов есть

$$\{Q_i, i=1,\ldots,kn/d\}=\{aQ, a\in L\}, L\subset K,$$

тогда очевидно

$$f_{kn}=\prod_{a\in L}m_{\zeta^a}.$$

Пусть равна нулю нетривиальная линейная комбинация

$$\sum_{i=0}^{n-1}a_i\alpha_i, a_i\in GF(q).$$

Очевидно она представляется в виде

$$\sum_{j=1}^{kn}b_j\zeta^j, b_j\in GF(q),$$

т.е. для ненулевого многочлена

$$f(x)=\sum_{j=0}^{kn-1}b_{j+1}x^j$$

имеем $f(\zeta) = 0$, и коэффициенты b_l и b_j будут всегда равны, если l, j принадлежат одному смежному классу K_i. Тогда для любого $a \in K$ имеем $f(\zeta^a) = 0$, так как

$$\sum_{j=1}^{kn} b_j \zeta^{aj} = \sum_{j=1}^{kn} b_j \zeta^j = 0.$$

Действительно, заменив в первой сумме индекс суммирования на $i = aj$, заметим, что индексы i, j принадлежат одному смежному классу по K, значит $b_i = b_j$ и обе суммы равны. Из равенства $f(\zeta^a) = 0$ следует, что многочлен f делится на многочлен m_{ζ^a}, поэтому он делится и на произведение

$$f_{kn} = \prod_{a \in L} m_{\zeta^a},$$

что невозможно, так как его степень равна $nk - 1$ и меньше степени многочлена f_{kn}. Противоречие доказывает линейную независимость элементов гауссова периода.

Осталось рассмотреть случай, когда γ и q не порождают мультипликативную группу поля $GF(p)$. Тогда в силу периодичности только часть классов имеет вид $K_i = Kq^i, i = 0, \ldots, m-1$, где $Kq^m = K, m < n$ и гауссов период строго содержит в себе систему вида $\alpha_i = \alpha_0^{q^i}, i = 0, \ldots, m-1$, лежащую в поле $GF(q^m)$, и не образующую базиса в поле $GF(q^n)$.

Упражнение 2.5.5 Докажите, что m делит n.

Как и в предыдущем разделе, элементы поля $GF(q^n)$, разложенные по GNB $\{\alpha_i\}$ можно представить в виде

$$\sum_{i=1}^{kn} a_i \zeta^i,$$

если их рассматривать как элементы поля $GF(q^d)$, причем переход к этому представлению выполняется бесплатно.

Упражнение 2.5.6 Докажите это. Докажите, что переход от базиса $\{\zeta^i, i = 0, \ldots, kn-1\}$ к базису $\{\zeta^i, i = 1, \ldots, kn\}$ и обратно делается с линейной сложностью.

Однако это представление неоднозначно. Но его можно сделать однозначным, если потребовать выполнение дополнительного условия, а именно равенства всех пар коэффициентов a_i, a_j, если индексы i, j лежат в одном смежном классе по K. Назовем это условие условие согласованности (коэффициентов). Выполнение этого условия очевидно можно гарантировать только для элементов подполя $GF(q^n)$ поля $GF(q^d)$. Однозначность доказывается также, как доказывалась линейная независимость данного GNB.

Упражнение 2.5.7 Докажите это. Докажите также, что восстановление по координатам в базисе $\{\zeta^i, i = 1, \ldots, kn\}$ координат в базисе $\{\alpha_i\}$ делается бесплатно при выполнении условия согласованности.

Умножение в поле $GF(q^n)$ можно производить, пользуясь координатами в базисе $\{\zeta^i, i = 1, \ldots, kn\}$. Как и в предыдущем разделе, рассматриваем это умножение, как умножение многочленов с нулевым свободным членом по модулю многочлена f_{kn}, с соблюдением равенства нулю свободного члена. Для этого сначала выполняется умножение по модулю $x^p - 1$, а потом вы полняется приведение по модулю f_{kn} к виду с нулевым свободным членом. Непосредственно проверяется, что если оба многочлена удовлетворяли условию согласованности, то и их произведение по модулю $x^p - 1$ тоже удовлетворяет условию согласованности.

Упражнение 2.5.8 Докажите это.

Указание. Пусть индексы таковы, что $i = jh \bmod p, h \in K$. Тогда $x_a = x_{ah \bmod p}, y_b = y_{bh \bmod p}$, при любых a, b и формально полагая $y_0 = 0$ имеем

$$z_i = \sum_{a=1}^{kn} x_a y_{i-a \bmod p} = \sum_{a=1}^{kn} x_a y_{hj-a \bmod p} = \sum_{a=1}^{kn} x_{ah \bmod p} y_{hj-ah \bmod p} =$$

$$= \sum_{a=1}^{kn} x_{a \bmod p} y_{j-a \bmod p} = z_j.$$

После этого ко всем коэффициентам z_i прибавляется z_0.

Поэтому утверждение о сложности умножения в GNB порядка k из предыдущего раздела доказано и в случае $d < kn$. Очевидно также, что любой ненулевой элемент поля $GF(q^n)$, будучи представлен в виде разложения по базису $\{\zeta^i, i = 1, \ldots, kn\}$ с выполнением условия согласованности, будет обратимым элементом в кольце многочленов по модулю f_{kn}, в которое мы вложили (на самом деле изоморфно) поле $GF(q^n)$. Действительно, обратному для него элементу поля соответствует при рассматриваемом вложении элемент кольца, который при умножении на первый элемент по модулю $x^p - 1$ будет равен единице (которая записывается в виде многочлена с одинаковыми коэффициентами), значит этот элемент обратим. Поэтому инвертирование в GNB базисе тоже сводится к инвертированию в рассматриваемом кольце, т.е. к инветированию по модулю многочлена f_{kn}, которое можно выполнить с помощью быстрого расширенного алгоритма Евклида.

На практике однако в случае $q = 2$ вместо быстрого алгоритма лучше применять, как замечено в работе [150], аналог бинарного алгоритма Евклида, не доводя вычисления до конца (так называемый полуинверсный алгоритм).

2.5.4 Еще один вывод таблицы умножения для GNB

Продолжая рассуждения, начатые в предыдущем разделе, дадим краткий вывод формулы для вычисления произведения элементов GNB, несколько более точную, чем была получена выше. Для этого напомним, что через s был обозначен индекс, такой, что $-1 \in K_s$ и было доказано, что при четном k $s = 0$, а при нечетном k $s = n/2$. Положим $\delta_j = 0$, если $j \neq s$, и $\delta_s = 1$. Обозначим также $c_{j,h}$ число общих элементов у K_h и $1 + K_j$ (эти числа называются циклотомическими числами). Тогда справедлива формула

$$\alpha_i \alpha_j = k\delta_{j-i} + \sum_{h=0}^{n-1} c_{j-i,h} \alpha_{h+i} = \sum_{h=0}^{n-1} (c_{j-i,h} - k\delta_{j-i}) \alpha_{h+i},$$

где операции с индексами проводятся по модулю n. Достаточно доказать первое равенство, так как второе следует из него, если учесть тот факт, что сумма элементов GNB равна -1. Подставляя вместо α_i их выражения в виде гауссовых сумм, имеем

$$\sum_{a \in K} \zeta^{q^i a} \sum_{b \in K} \zeta^{q^j b} = \sum_{a,b \in K} \zeta^{q^i a + q^j b} = \left(\sum_{a,b \in K} \zeta^{a(1+q^{j-i}b)} \right)^{q^i}.$$

Так либо $1 + q^{j-i}b = 0$, либо $1 + q^{j-i}b \in K_h$ для единственного $h, 0 \leq h < n$, то либо в первом случае

$$\sum_{a \in K} \zeta^{a(1+q^{j-i}b)} = k,$$

либо во втором случае

$$\sum_{a\in K}\zeta^{a(1+q^{j-i}b)}=\sum_{a\in K}\zeta^{q^h a}.$$

Но $1+q^{j-i}b=0$ возможно только при $j-i=s$, поэтому слагаемое, равное k в рассматриваемой сумме можно записать в виде $k\delta_{j-i}$. Остальные слагаемые при $b\neq i-j$ можно просуммировать по индексу $h, 0\leq h<n$, при этом коэффициент при сумме

$$\sum_{a\in K}\zeta^{q^h a}$$

будет равен числу таких b, что $1+q^{j-i}b\in K_h$, т.е. цикломатическому числу $c_{j-i,h}$. Поэтому

$$\left(\sum_{a,b\in K}\zeta^{a(1+q^{j-i}b)}\right)^{q^i}=\left(k\delta_{j-i}+\sum_{h=0}^{n-1}c_{j-i,h}\alpha_h\right)^{q^i}=k\delta_{j-i}+\sum_{h=0}^{n-1}c_{j-i,h}\alpha_h^{q^i}=$$

$$=k\delta_{j-i}+\sum_{h=0}^{n-1}c_{j-i,h}\alpha_{h+i}.$$

Очевидно, что число элементов в $1+K_j$ равно k, значит имеется не более k ненулевых коэффициентов $c_{j,h}$. Сравнивая полученные формулы с формулами, использовавшимися при выводе формул Месси-Омура

$$\alpha_0\alpha_i=\sum_{j=0}^{n-1}t_{i,j}\alpha_j,$$

имеем

$$t_{i,j}=c_{i,j}-k\delta_i.$$

2.5.5 Примеры гауссовых нормальных базисов в полях $GF(2^n)$

Построим GNB для размерности $n=4$ и $k=3$. Тогда $p=nk+1=13$ простое и в поле $GF(p)$ элемент $q=2$ имеет порядок 12 (является примитивным элементом), так как $2^4=3\neq 1 \bmod p, 2^6=-1\neq 1 \bmod p$. Условия существования базиса выполнены. Положим $\gamma=3$, тогда $\gamma^3=1 \bmod p$, т.е. γ примитивный корень степени $k=3$ из единицы.

Проверим, что любой элемент от 1 до 12 в поле $GF(13)$ однозначно представим виде $\gamma^i q^j=3^i 2^j, i<3, j<4$, а также вычислим функции $a(i,j), b(i,j)$ с аргументами $0\leq i<3, 0\leq j<4$, и значениями во множествах $\{0,1,2\}$, $\{0,1,2,3\}$, такие, что

$$\gamma^i q^j+1=3^i 2^j+1=\gamma^{a(i,j)}q^{b(i,j)}=3^{a(i,j)}2^{b(i,j)}, i<k, j<n.$$

Так как $3=2^4 \bmod 13$, то $\gamma^i q^j=3^i 2^j=2^{4i+j}$, а так как очевидно, что любое $k, 1\leq k\leq 12$, путем деления на 4 однозначно представимо в виде

$$k=4i+j, i<3, j<4, i=\lfloor k/4\rfloor, j=k-4i,$$

то, используя логарифмирование по основанию 2,, получаем, что любой ненулевой элемент x поля $GF(13)$ однозначно представим в виде

$$x=\gamma^i q^j=3^i 2^j=2^{4i+j}, i<3, j<4, i=\lfloor(\log_2 x)/4\rfloor, j=\log_2 x-4i,$$

где $\log_2 x$ есть дискретный логарифм в этом поле по основанию 2. Тогда функции a, b можно выразить формулами

$$a(i,j)=\lfloor\left(\log_2\left(3^i 2^j+1\right)\right)/4\rfloor, b(i,j)=\log_2\left(3^i 2^j+1\right)-4a(i,j).$$

Для составления таблицы этих функций очевидно проще всего вначале вычислить таблицу логарифмов в поле $GF(13)$. Она имеет вид

x	1	2	3	4	5	6	7	8	9	10	11	12
$\log_2 x$	0	1	4	2	9	5	11	3	8	10	7	6

Таблица функции $3^i 2^j + 1 \bmod 13, i < 3, j < 4$ имеет вид

	0	1	2	3
0	2	3	5	9
1	4	7	0	12
2	10	6	11	8

Из нее видно, что $3^s 2^i = -1 \bmod 13$ только при $s = 1, i = 2 = n/2$.

Таблица функции $\log_2(3^i 2^j + 1 \bmod 13), i < 3, j < 4$ имеет вид

	0	1	2	3
0	1	4	9	8
1	2	11		6
2	10	5	7	3

Таблица функций a, b имеет вид

	0	1	2	3
0	0,1	1,0	2,1	2,0
1	0,2	2,3		1,2
2	2,2	1,1	1,3	0,3

Порождающим элементом базиса является

$$\alpha = \zeta + \zeta^3 + \zeta^9, \zeta^{13} = 1, \zeta \in GF(2^{12}.$$

Все остальные элементы базиса представляются в виде

$$\alpha^2 = \zeta^2 + \zeta^6 + \zeta^{18} = \zeta^2 + \zeta^6 + \zeta^5,$$

$$\alpha^4 = \zeta^4 + \zeta^{12} + \zeta^{10},$$

$$\alpha^8 = \zeta^8 + \zeta^{24} + \zeta^{20} = \zeta^8 + \zeta^{11} + \zeta^7.$$

Очевидно

$$\alpha^{16} = \zeta^{16} + \zeta^{22} + \zeta^{14} = \zeta^3 + \zeta^9 + \zeta = \alpha,$$

поэтому $\alpha \in GF(2^4)$ и построенная система действительно лежит в поле $GF(2^4)$. Очевидно, что сумма базисных элементов (след элемента α в поле $GF(2)$)

$$\alpha + \alpha^2 + \alpha^4 + \alpha^8 = \sum_{i=1}^{12} \zeta^i = 1,$$

так как

$$\sum_{i=0}^{12} \zeta^i = \frac{\zeta^{13} - 1}{\zeta - 1} = 0.$$

Так как разложения элементов α^{2^i} по системе $\zeta^j, j = 1, \ldots, 12$ состоят из различных элементов, то линейная независимость системы $\alpha^{2^i}, i = 0, 1, 2, 3$ над полем $GF(2)$ следует из линейной

независимости над тем же полем системы $\zeta^j, j = 1, \ldots, 12$. Последнее же следует из линейной независимости системы $\zeta^j, j = 0, \ldots, 11$, которая была доказана, как уже отмечалось, в разделе о ONB первого типа.

Составим таблицу умножения в рассматриваемом базисе. Для краткости обозначаем α^{2^i} через α_i. Очевидно $\alpha_0\alpha_0 = \alpha_1$, и далее, разбивая слагаемые в произведениях на тройки, и учитывая, что $\zeta^{13} = 1$,

$$\alpha_0\alpha_1 = \left(\zeta + \zeta^3 + \zeta^9\right)\left(\zeta^2 + \zeta^6 + \zeta^5\right) =$$
$$\left(\zeta^3 + \zeta^9 + \zeta\right) + \left(\zeta^7 + \zeta^8 + \zeta^{11}\right) + \left(\zeta^6 + \zeta^5 + \zeta\right) = \alpha_0 + \alpha_1 + \alpha_3.$$

Сравнивая полученную формулу с выведенной выше общей формулой при нечетном k и $i \neq n/2$

$$\alpha\alpha^{q^i} = \sum_{j=0}^{k-1} \alpha^{q^{b(j,i)}},$$

видим, что они совпадают. Далее, используя таблицу для функции b, имеем

$$\alpha_0\alpha_3 = \alpha_0 + \alpha_2 + \alpha_3.$$

Проверим это равенство непосредственным умножением

$$\alpha_0\alpha_3 = \left(\zeta + \zeta^3 + \zeta^9\right)\left(\zeta^8 + \zeta^{11} + \zeta^7\right) =$$
$$\left(\zeta^9 + \zeta + \zeta^3\right) + \left(\zeta^{12} + \zeta^{10} + \zeta^4\right) + \left(\zeta^8 + \zeta^{11} + \zeta^5\right) = \alpha_0 + \alpha_2 + \alpha_3.$$

При $i = 2 = n/2, s = 1$, используя формулу

$$\alpha\alpha^{q^i} = k + \sum_{j \neq s} \alpha^{q^{b(j,i)}},$$

имеем

$$\alpha_0\alpha_2 = 1 + \alpha_1 + \alpha_3 = \alpha_0 + \alpha_2.$$

Проверим и эту формулу непосредственным вычислением

$$\alpha_0\alpha_2 = \left(\zeta + \zeta^3 + \zeta^9\right)\left(\zeta^4 + \zeta^{12} + \zeta^{10}\right) =$$
$$\left(\zeta^5 + \zeta^2 + \zeta^6\right) + \left(\zeta^0 + \zeta^0 + \zeta^0\right) + \left(\zeta^{11} + \zeta^7 + \zeta^8\right) =$$
$$= \alpha_1 + 3 + \alpha_3 = \alpha_1 + 1 + \alpha_3 = \alpha_0 + \alpha_2.$$

Окончательно таблица умножения имеет вид

$$\alpha_0\alpha_0 = \alpha_1, \alpha_0\alpha_1 = \alpha_0 + \alpha_1 + \alpha_3.\alpha_0\alpha_2 = \alpha_0 + \alpha_2.\alpha_0\alpha_3 = \alpha_0 + \alpha_2 + \alpha_3,$$

значит, матрица $T = (t_{i,j}), i < 4, j < 4$, определяемая равенством

$$\alpha_0\alpha_i = \sum_{j=0}^{n-1} t_{i,j}\alpha_j,$$

имеет вид

$$\begin{matrix} 0 & 1 & 0 & 0 \\ 1 & 1 & 0 & 1 \\ 1 & 0 & 1 & 0 \\ 1 & 0 & 1 & 1 \end{matrix}$$

Напомним, что формула умножения методом Месси-Омура задается равенствами

$$z = xy, z_i = A(S^i x, S^i y,)i = 0, 1, \ldots, n-1,$$

где $S^i(x) = S^i(x_0, \ldots, x_{n-1}) = (x_i, \ldots, x_{n-1}, x_0, \ldots, x_{i-1})$ — циклический сдвиг координат (шифт) на i позиций влево, а

$$A(x, y) = \sum_{i,j=0}^{n-1} a_{i,j} x_i y_j$$

билинейная форма с $n \times n$-матрицей $A = (a_{i,j}), a_{i,j} = t_{i-j \bmod n, -j \bmod n}$. В рассматриваемом случае эта матрица имеет вид

$$\begin{matrix} 0 & 1 & 1 & 1 \\ 1 & 0 & 1 & 0 \\ 1 & 1 & 0 & 0 \\ 1 & 0 & 0 & 1 \end{matrix}$$

а формула умножения имеет вид

$$z_0 = x_0(y_1 + y_2 + y_3) + x_1(y_0 + y_2) + x_2(y_0 + y_1) + x_3(y_0 + y_3),$$

$$z_1 = x_1(y_2 + y_3 + y_0) + x_2(y_1 + y_3) + x_3(y_1 + y_2) + x_0(y_1 + y_0),$$

$$z_2 = x_2(y_3 + y_0 + y_1) + x_3(y_2 + y_0) + x_0(y_2 + y_3) + x_1(y_2 + y_1),$$

$$z_3 = x_3(y_0 + y_1 + y_2) + x_0(y_3 + y_1) + x_1(y_3 + y_0) + x_2(y_3 + y_2).$$

Рассмотренный пример не очень интересен с прикладной точки зрения, так как в поле $GF(2^4)$ существует оптимальный нормальный базис первого типа $\{\xi, \xi^2, \xi^4, \xi^8, \}$, определяемый корнем многочлена $p(x) = x^4 + x^3 + x^2 + x + 1$, который имеет меньшую сложность и в котором, следовательно, умножение выполняется по более простым формулам.

Рассмотрим более интересный пример. Построим GNB для размерности $n = 13$, для которой не существует оптимального нормального базиса. Выберем $k = 4$. Тогда $p = nk + 1 = 13$ простое (при меньших k получаются составные числа) и в поле $GF(p)$ элемент $q = 2$ имеет порядок $p - 1 = 52$ (является примитивным элементом), так как $2^4 \neq 1 \bmod p, 2^{13} = 30 \neq 1 \bmod p$. Условия существования базиса выполнены. Положим $\gamma = 30 = -23 \bmod p$, тогда $\gamma^4 = 1 \bmod p, \gamma^2 = -1 \bmod p$, т.е. γ примитивный корень степени $k = 4$ из единицы.

Проверим, что любой элемент от 1 до 52 в поле $GF(53)$ однозначно представим виде $\gamma^i q^j = 30^i 2^j, i < 4, j < 13$, а также вычислим функции $a(i,j), b(i,j)$ с аргументами $0 \leq i < 4, 0 \leq j < 13$, и значениями во множествах $\{0, 1, 2, 3\}$, $\{0, \ldots, 12\}$, такие, что

$$\gamma^i q^j + 1 = 30^i 2^j + 1 = \gamma^{a(i,j)} q^{b(i,j)} = 30^{a(i,j)} 2^{b(i,j)}, i < k, j < n.$$

Так как $30 = 2^{13} \bmod 13$, то $\gamma^i q^j = 30^i 2^j = 2^{13i+j}$, а так как очевидно, что любое $k, 1 \leq k \leq 52$, путем деления на 13 однозначно представимо в виде

$$k = 13i + j, i < 4, j < 13, i = \lfloor k/13 \rfloor, j = k - 13i,$$

то, используя логарифмирование по основанию 2,, получаем, что любой ненулевой элемент x поля $GF(52)$ однозначно представим в виде

$$x = \gamma^i q^j = 30^i 2^j = 2^{13i+j}, i < 4, j < 13, i = \lfloor (\log_2 x)/13 \rfloor, j = \log_2 x - 13i,$$

где $\log_2 x$ есть дискретный логарифм в этом поле по основанию 2. Тогда функции a, b можно выразить формулами

$$a(i,j) = \lfloor \left(\log_2 \left(30^i 2^j + 1\right)\right) / 13 \rfloor, b(i,j) = \log_2 \left(30^i 2^j + 1\right) - 13a(i,j).$$

Для составления таблицы этих функций вычислим таблицу логарифмов в поле $GF(53)$. Достаточно ее вычислить для $x \leq 26$, так как при $x > 26$ имеем $\log_2 x = \log_2(53 - x) + 26 \bmod 52$.

Она имеет вид

x	1	2	3	4	5	6	7	8	9	10	11	12	13
$\log_2 x$	0	1	17	2	47	18	14	3	34	48	6	19	24
x	14	15	16	17	18	19	20	21	22	23	24	25	26
$\log_2 x$	15	12	4	10	35	37	49	31	7	39	20	42	25

Таблица функции b имеет вид

	0	1	2	3	4	5	6	7	8	9	10	11	12
0	1	4	8	8	10	10	6	0	3	10	9	9	4
1	7	3	12	7	2	1	11	3	2	8	11	2	12
2		0	4	1	12	7	9	5	9	11	4	10	2
3	7	5	11	12	1	8	6	0	5	6	5	6	3

Из нее видно, что $30^s 2^i = -1 \bmod 53$ только при $s = 2 = k/2, i = 0$.

Порождающим элементом базиса является

$$\alpha_0 = \alpha = \zeta + \zeta^{30} + \zeta^{30^2} + \zeta^{30^3} = \zeta + \zeta^{30} + \zeta^{52} + \zeta^{23} = \zeta + \zeta^{-23} + \zeta^{-1} + \zeta^{23},$$

$$\zeta^{53} = 1, \zeta \in GF(2^{52}.$$

Все остальные элементы базиса представляются в виде

$$\alpha_1 = \zeta^2 + \zeta^{-2} + \zeta^7 + \zeta^{-7}, \alpha_2 = \zeta^4 + \zeta^{-4} + \zeta^{14} + \zeta^{-14}, \alpha_3 = \zeta^8 + \zeta^{-8} + \zeta^{25} + \zeta^{-25},$$

$$\alpha_4 = \zeta^{16} + \zeta^{-16} + \zeta^3 + \zeta^{-3}, \alpha_5 = \zeta^{21} + \zeta^{-21} + \zeta^6 + \zeta^{-6}, \alpha_6 = \zeta^{11} + \zeta^{-11} + \zeta^{12} + \zeta^{-12},$$

$$\alpha_7 = \zeta^{22} + \zeta^{-22} + \zeta^{24} + \zeta^{-24}, \alpha_8 = \zeta^9 + \zeta^{-9} + \zeta^5 + \zeta^{-5}, \alpha_9 = \zeta^{18} + \zeta^{-18} + \zeta^{10} + \zeta^{-10},$$

$$\alpha_{10} = \zeta^{17} + \zeta^{-17} + \zeta^{20} + \zeta^{-20}, \alpha_{11} = \zeta^{19} + \zeta^{-19} + \zeta^{13} + \zeta^{-13}, \alpha_{12} = \zeta^{15} + \zeta^{-15} + \zeta^{26} + \zeta^{-26},$$

Очевидно

$$\alpha_{13} = \alpha^{2^{13}} = \zeta^{23} + \zeta^{-23} + \zeta^1 + \zeta^{-1} = \alpha,$$

поэтому $\alpha \in GF(2^{13})$ и построенная система действительно лежит в поле $GF(2^{13})$. Очевидно, что сумма базисных элементов (след элемента α в поле $GF(2)$)

$$\sum_{i=0}^{12} \alpha_i = \sum_{i=1}^{26} (\zeta^i + \zeta^{-i}) = \sum_{i=1}^{52} \zeta^i = 1,$$

так как

$$\sum_{i=0}^{52} \zeta^i = \frac{\zeta^{53} - 1}{\zeta - 1} = 0.$$

Так как разложения элементов α_i по системе $\zeta^j, j = 1, \ldots, 52$, состоят из разных элементов, то как и в предыдущем примере проверяем, что система $\{\alpha_i, i = 0, \ldots, 12\}$ образует нормальный базис в поле $GF(2^{13})$. Составим таблицу умножения в рассматриваемом базисе. Очевидно $\alpha_0 \alpha_0 = \alpha_1$. Используя формулу при $i \neq 0$

$$\alpha \alpha^{q^i} = \sum_{j=0}^{k-1} \alpha^{q^{b(j,i)}},$$

и таблицу для функции b, получаем таблицу умножения в виде

$$\alpha_0 \alpha_0 = \alpha_1, \alpha_0 \alpha_1 = \alpha_0 + \alpha_3 + \alpha_4 + \alpha_5, \alpha_0 \alpha_2 = \alpha_4 + \alpha_8 + \alpha_{11} + \alpha_{12},$$

$$\alpha_0\alpha_3 = \alpha_1 + \alpha_7 + \alpha_8 + \alpha_{12}, \alpha_0\alpha_4 = \alpha_1 + \alpha_2 + \alpha_{10} + \alpha_{12}, \alpha_0\alpha_5 = \alpha_1 + \alpha_7 + \alpha_8 + \alpha_{10},$$

$$\alpha_0\alpha_6 = \alpha_9 + \alpha_{11}, \alpha_0\alpha_7 = \alpha_3 + \alpha_5, \alpha_0\alpha_8 = \alpha_2 + \alpha_3 + \alpha_5 + \alpha_9, \alpha_0\alpha_9 = \alpha_6 + \alpha_8 + \alpha_{10} + \alpha_{11},$$

$$\alpha_0\alpha_{10} = \alpha_4 + \alpha_5 + \alpha_9 + \alpha_{11}, \alpha_0\alpha_{11} = \alpha_2 + \alpha_6 + \alpha_9 + \alpha_{10}, \alpha_0\alpha_{12} = \alpha_2 + \alpha_3 + \alpha_4 + \alpha_{12}.$$

значит, матрица $T = (t_{i,j}), i < 4, j < 4$, определяемая равенством

$$\alpha_0\alpha_i = \sum_{j=0}^{n-1} t_{i,j}\alpha_j,$$

имеет вид

0	1	0	0	0	0	0	0	0	0	0	0	0
1	0	0	1	1	1	0	0	0	0	0	0	0
0	0	0	0	1	0	0	0	1	0	0	1	1
0	1	0	0	0	0	0	1	1	0	0	0	1
0	1	1	0	0	0	0	0	0	0	1	0	1
0	1	0	0	0	0	0	1	1	0	1	0	0
0	0	0	0	0	0	0	0	0	1	0	1	0
0	0	0	1	0	1	0	0	0	0	0	0	0
0	0	1	1	0	1	0	0	0	1	0	0	0
0	0	0	0	0	0	1	0	1	0	1	1	0
0	0	0	0	1	1	0	0	0	1	0	1	0
0	0	1	0	0	0	1	0	0	1	1	0	0
0	0	1	1	1	0	0	0	0	0	0	0	1

Матрицу $(t_{i,j})$ можно также получить, пользуясь ее выраженинем через цикломатические числа

$$t_{i,j} = c_{i,j} - k\delta_i,$$

где в рассматриваемом случае $k = 4$, $\delta_i = 1$ только при $i = k/2 = 2$, впрочем независимо от δ в данном случае $t_{i,j} = c_{i,j}$. Напомним, что $c_{i,j} = |(1+K_i) \cap K_j,|$ где $K_i = q^i K = 2^i K, i = 0, \ldots, 12$ — смежные классы группы $GF(53) \setminus \{0\}$ по подгруппе $K = \{\gamma^j, j = 0, 1, 2, 3\}$, где $\gamma = 30 = -23 \bmod 53$. Так как $K = K_0 = \{1, 30, 52, 23\} = \{1, -23, -1, 23\}$, то выше фактически уже было вычислено при нахождении α_i, что

$$K_1 = 2K = \{2, -2, 7, -7\}, K_2 = 4K = \{4, -4, 14, -14\}, K_3 = 8K = \{8, -8, 25, -25\},$$

$$K_4 = 2^4K = \{16, -16, 3, -3\}, K_5 = 2^5K = \{21, -21, 6, -6\}, K_6 = 2^6K = \{11, -11, 12, -12\},$$

$$K_7 = 2^7K = \{22, -22, 24, -24\}, K_8 = 2^8K = \{9, -9, 5, -5\}, K_9 = 2^9K = \{18, -18, 10, -10\},$$

$$K_{10} = 2^{10}K = \{17, -17, 20, -20\}, K_{11} = 2^{11}K = \{19, -19, 13, -13\},$$
$$K_{12} = 2^{12}K = \{15, -15, 26, -26\}.$$

Отсюда имеем $K_0 + 1\{2, -22, 0, 24\}$,

$$K_1 + 1 = \{3, -1, 8, -6\}, K_2 + 1 = \{5, -3, 15, -13\}, K_3 + 1 = \{9, -7, 26, -24\},$$

$$K_4 + 1 = \{17, -15, 4, -2\}, K_5 + 1 = \{22, -20, 7, -5\}, K_6 + 1 = \{12, -10, 13, -11\},$$
$$K_7 + 1 = \{23, -21, 25, -23\}, K_8 + 1 = \{10, -8, 6, -4\}, K_9 + 1 = \{19, -17, 11, -9\},$$
$$K_{10} + 1 = \{18, -16, 21, -19\}, K_{11} + 1 = \{20, -18, 14, -12\},$$

$$K_{12}+1=\{16,-14,-26,-25\}.$$

Перебирая попарные пересечения $1+K_i$ и K_j приходим к той же матрице. Интересно, что она является симметрической, что никак не было очевидно заранее.

Матрица A связана с матрицей T соотношением $a_{i,j}=t_{i-j \bmod n,-j \bmod n}$. Поэтому она получается из нее «косой транспозицией», при которой первый слева столбец (имеющий номер нуль) остается на месте, а остальные элементы попарно меняются местами, так, что второй столбец меняется местами с последним, при этом еще его элементы циклически сдвигаются на одну позицию вниз (и соответственно вверх, если рассматривается передвижение элементов второго столбца в последний столбец), третий столбец меняется с предпоследним и т.д. Матрица A, что тоже не очевидно было заранее, совпадает в данном случае с матрицей T (то, что она должна получится симметричной, заранее было очевидно), поэтому формула умножения в этом базисе имеет вид

$$z_i=(i,i\oplus 1)+(i\oplus 1,i\oplus 3)+(i\oplus 1,i\oplus 4)+(i\oplus 1,i\oplus 5)+(i\oplus 2,i\oplus 4)+(i\oplus 2,i\oplus 8)+$$
$$+(i\oplus 2,i\oplus 11)+(i\oplus 2,i\oplus 12)+(i\oplus 3,i\oplus 7)+(i\oplus 3,i\oplus 8)+(i\oplus 3,i\oplus 12)+$$
$$+(i\oplus 4,i\oplus 12)+(i\oplus 4,i\oplus 10)+(i\oplus 5,i\oplus 7)+(i\oplus 5,i\oplus 8)+(i\oplus 5,i\oplus 10)+$$
$$+(i\oplus 6,i\oplus 9)+(i\oplus 6,i\oplus 11)+(i\oplus 8,i\oplus 9)+(i\oplus 9,i\oplus 10)+$$
$$+(i\oplus 9,i\oplus 11)+(i\oplus 10,i\oplus 11)+(i\oplus 12),$$

где через (i,j) для краткости обозначено $x_iy_j+x_jy_i$, а $(i)=x_iy_i$, знак $\oplus$ означает сложение по модулю $n=13$. Поэтому сложность построенного GNB четвертого типа равна $22\cdot 2+1=45$.

2.5.6 Порядки генераторов базисов низкой сложности и быстрый алгоритм экспоненцирования

В [98] экспериментально показано, что генераторы ONB второго и третьего типов довольно часто бывают примитивными элементами или имеют мультипликативные полрядки вида $(2^n-1)/k$, где k — небольшое число (среди чисел от 2 до 1000 во всех случаях существования таких базисов оно как правило равно 1, редко 3, очень редко $7,9,11$, а максимум 381). Это дает возможность использовать эти элементы как базисные для имплементации ключевого обмена Диффи-Хеллмана [?]. В [98] доказано, что битовая сложность возведения в степень $e<2^n$ генератора α ONB второго и третьего типов оценивается как $O(n\nu_2(e)$, где $\nu_2(e)$ — число единиц в двоичном разложении числа e, и поэтому существенно меньше, чем полученная выше оценка для общего случая. Это дает возможность ускорения программной имплементации ключевого обмена Диффи-Хеллмана для конечных полей соответствующих размерностей. Далее мы получим указанную оценку более просто и для произвольного нормального базиса B низкой сложности, т.е. такого, что $C_B=O(n)$. Для GNB аналогичный факт доказан в [99].

Вначале докажем, что аддитивная сложность умножения в любом нормальном базисе B произвольного элемента $x=x_0\alpha_0+\dots x_{n-1}\alpha_{n-1}$ на произвольный элемент этого базиса α_i равна (C_B-n), а мультипликативная равна не боолее C_B, а в случае $q=2$ равна нулю. Действительно, для вычисления произведения

$$z_0\alpha_0+\dots z_{n-1}\alpha_{n-1}=x\alpha_i=x_0\alpha_0+\dots x_{n-1}\alpha_{n-1}\alpha_i$$

заметим, что z_0 в общем случае вычисляется в виде билинейной формы с матрицей A, содержащей C_B ненулевых элементов, а остальные z_i вычисляются также, только в билинейной форме делается циклический сдвиг переменных. Так как в данном случае вектор координат сомножителя α_i содержит единицу на i-м месте, а в остальных местах нули (такой вектор обозначаем далее e_i и называем единичным), при подстановке этого вектора в рассматриваемую билинейную форму из нее получается линейная форма, коэффициенты которой берутся из

i-го столбца, и поэтому аддитивная сложность этой формы равна $c_i - 1$, а мультипликативная не больше c_i, где c_i — число единиц в i-м столбце матрицы A (а в случае $q = 2$ равна нулю.) Аналогичным образом оцениваем аддитивную сложность вычисления произвольного z_j как $c_{i-j \bmod n} - 1$, а мультипликативную — как $c_{i-j \bmod n}$, так как при соответствующем циклическом сдвиге вектор e_i переходит в вектор $e_{i-j \bmod n}$. Поэтому полная аддитивная сложность равна

$$\sum_{i=0}^{n-1}(c_{i-j \bmod n} - 1) = \sum_{i=0}^{n-1} c_i - n = C_B - n,$$

а мультипликативная не больше C_B.

В случае $q = 2$ и ONB второго или третьего типа в [98] формула умножения вычислена в явном виде:

$$\sum_{k=1}^{n} a_k \alpha_{k+1} \alpha_i = \sum_{k=1}^{s}(a_{i-k} + a_{k+i})\alpha_k + \sum_{k=s+1}^{d} f_k \alpha_k + \sum_{k=d+1}^{n} (a_{k-i} + a_{2n+1-k-i})\alpha_k,$$

где $f_k = a_{i-k} + a_{2n+1-k-i}$ при $i > n - i$ и $f_k = a_{k-i} + a_{k+i}$ при $i < n - i$.

Упражнение 2.5.9 Докажите эту формулу.

В случае $q = 2$ для выполнения экспоненцирования α^e можно применить вариант бинарного метода

$$\alpha^e = \prod_{k=1}^{n} \alpha^{2^k e_k} = \prod_{k=1}^{n} \alpha_k^{e_k},$$

где e_k —биты двоичного разложения

$$e = \sum_{k=1}^{n} e_k 2^k.$$

Последовательно умножая на α_k при $e_k = 1$, вычисляем экспоненту с помощью $\nu_2(e) - 1$ умножений на элементы базиса α_i, поэтому аддивная сложность ее вычисления равна $(\nu_2(e) - 1)(S_B - n)$, а мультипликативная не больше $(\nu_2(e) - 1)S_B$. В общем случае можно поступить подобным же образом.

Упражнение 2.5.10 Докажите, что для генератора ONB первого типа экспоненцирование можно выполнить со сложностью $O(n \log\log n \log\log\log n$ при условии бесплатного доступа к памяти объема $O(n^2)$. Почему это результат совершенно бесполезен для приложений?

В [98] отмечено также, что легко вычисляется обратный элемент для генератора ONB:

$$\alpha^{-1} = \sum_{k=1}^{n} a_k \alpha_k,$$

где $a_k = 1$ если и только если нечетно число $l = k/2$ при четном k и $l = n - (k - 1)/2$ при нечетном k.

Упражнение 2.5.11 Докажите это. Докажите также, что сложность вычисления обратного для любого элемента данного ONB равна $O(n)$.

В общем случае для нахождения обратного элемента генератора GNB k-го типа надо решить систему линейных уравнений с матрицей, в которой в каждой строке не более k ненулевых элементов.

Упражнение 2.5.12 Докажите, что обратный элемент для α^e, где $1 \le e < 2^n - 1$, а α генератор нормального базиса B в поле $GF(2^n)$, можно вычислить со сложностью $O(n - \nu_2(e))S_B$.

Указание: $\alpha^{-e} = \alpha^{2^n-1-e}$.

Результаты о порядках генераторов ONB в [98] строго не доказаны. Шпарлинский и фон цур Гатен доказали, что эти порядки не меньше $2^{\sqrt{2n}-2}$. В [98] выдвинута гипотеза о том, что если n и $2n+1$ простые, то генератор ONB в поле $GF(2^n)$ — примитивный элемент этого поля.

В [99] выдвинута гипотеза о том, что если n и $kn+1$ простые, $k \log_2 n + 1$, то генератор GNB типа k в поле $GF(2^n)$ — примитивный элемент этого поля. Существование GNB типа k легко доказать, так как порядок e двойки по модулю $kn+1$ не меньше $\log_2 p \ge \log_2 2n \ge k$, а n простое, поэтому $(nk/e, n) = 1$. В [99] также выдвинута гипотеза о том, что для любого n, не кратного 8, в поле $GF(2^n)$ существует GNB. Как уже отмечалось, при n, кратном 8, GNB не существуют. В [99] проверено, что среди всех 1179 GNB для $2 \le 527, 3 \le k \le 20\,906$ имеют примитивный генератор, и только 8 имеют порядок, меньший $(2^n - 1)/n$.

2.5.7 О сложности порождения нормальных базисов, примитивных элементов и неприводимых многочленов

В предыдущих разделах были описаны некоторые вероятностные полиномиальные по сложности алгоритмы для решения этих задач. Детерминированные полиномиальные алгоритмы известны только в предположении справедливости расширенной гипотезы Римана (ERH).

Уже отмечалось, что в [83],[77], опираясь на ERH показано, что для всех простых p и n, не кратных p, существуют в поле $GF(p^n)$ GNB типа $k = O(n^3 \log^2 np)$.

Упражнение 2.5.13 Докажите, что проверку существования в поле $GF(p^n)$ GNB типа k можно выполнить со сложностью $O(\log_2 n \log_2^3 nk)$.

Указание. Согласно 2.5.2 для проверки условия $(n, nk/d) = 1$ нет необходимости вычислять порядок d элемента p по модулю $nk+1$, а достаточно проверить, что для любого простого делителя $l|n$ неверно, что $q^{nk/l} = 1 \bmod nk + 1$.

Используя указанные факты, можно найти k, для которого существует GNB типа k, со сложностью $O(\log_2^4 n)$ при n, не кратном p (а на самом деле даже со сложностью $O(\log_2^3 n) \log_2 \log_2 \log_2 n)$. Построить сам базис можно со сложностью $O(nk \log_2^2 k \log_2^2 nk + n^3 \log_2^2 p)$. Действительно, сначала вычислим подгруппу K порядка k в группе $GF(kn+1)\{0\}$.

Упражнение 2.5.14 Докажите, что это можно сделать со сложностью $O(nk \log_2^2 kM(GF(kn+1)) = O(nk \log_2^2 k \log_2^2 nk)$.

Указание. Для нахождения примитивного корня γ степени k из единицы в поле $GF(kn+1)$ достаточно проверить, что $\gamma^k = 1$, и при любом простом делителе l числа k $\gamma^{k/l} = 1$.

Потом построим разбиение $GF(kn+1)\{0\}$ на смежные классы $K_i = Kp^i, i = 0, \ldots n-1$ и вычислим сдвиги $1 + K_i i = 0, \ldots n-1$, одновремено вычисляя цикломатические числа $c_{i,j}$, а потом элементы матрицы T.

Упражнение 2.5.15 Докажите, что матрицы T и A можно вычислить со сложностью $O(nk)M(GF(kn+1)) = O(nk) \log_2^2 kn$.

Для вычисления базисных элементов

$$\alpha_i = \sum_{a \in K_i} \zeta^a, i = 0, \ldots, n-1$$

нет необходимости находить примитивный корень ζ степени $kn+1$ из единицы в поле $GF(2^d)$. Так как

$$\alpha_0 \alpha_i = \sum_{j=0}^{n-1} t_{i,j} \alpha_j,$$

то вектор-столбец $A = \alpha_0 \alpha$, где α — вектор-столбец из компонент $\alpha_i, i = 0, \ldots, n-1$, равен $T\alpha$, поэтому линейный оператор

$$T : GF(p^n)^n \to GF(p^n)^n$$

имеет собственный вектор α, соответствующий собственному значению α_0. Собственное значение α_0 является корнем характеристического многочлена $det(T - xE)$, где E — единичный оператор. Этот многочлен имеет коэффициенты из поля $GF(p)$ и может быть вычислен со сложностью $O(n^3)M(GF(p))$, как известно из линейной алгебры. Минимальный многочлен элемента α_0 имеет степень n и корни $\alpha_i, i = 0, \ldots, n-1$. Так как характеристический многочлен делится на минимальный, а их степени равны, то они совпадают. Поэтому неприводимый многочлен, соответствующий базису $\alpha_i, i = 0, \ldots, n-1$ вычислен со сложностью $O(n^3)M(GF(p))$. Тем самым базис полностью определен. Его элементы могут быть найдены путем разложения минимального многочлена на множители над полем $GF(p^n)$, что можно сделать со сложностью $n^{7/2+\epsilon_n} p^{1/2}$ согласно [105].

Для построения нормального базиса в случае n кратного p, представим n в виде $n_1 n_2$, где $n_2 = p^k$, а n_1 не кратно p. Нормальный базис в поле $GF(p^{n_1})$ уже построен с полиномиальной сложностью в предположении ERH. Нормальный базис в поле $GF(p^{n_2})$ можно построить с полиномиальной сложностью согласно [134]. Так как $(n_1, n_2) = 1$, то, как известно, произведение генераторов этих нормальных базисов является генератором нормального базиса в поле $GF(p^n)$. Его минимальный многочлен можно определить по минимальным многочленам генераторов этих базисов с полиномиальной сложностью относительно n с помощью следующего

Упражнение 2.5.16 Если f_1, f_2 минимальные многочлены нормальных базисов в полях $GF(p^{n_1}), GF(p^{n_2})$, состоящих из элементов $\{\alpha_i, i = 0, \ldots, n_1\}$ и $\{\beta_i, i = 0, \ldots, n_2\}$ то минимальный многочлен элементов $\alpha_i \beta_j$ равен

$$\prod_{i,j} (x - \alpha_i \beta_j) = \prod_i \alpha_i^{n_2} f_2(x/\alpha_i).$$

Так как минимальный многочлен нормального базиса неприводим, то тем самым с полиномиальной сложностью построен неприводимый многочлен над полем $GF(p)$.

В предположении существования GNB с генератором, имеющим мультипликативный порядок $(2^n - 1)/n$ в [99] при n не кратном 8 указан полиномиальный по сложности алгоритм порождения примитивных элементов (и соответственно примитивных многочленов) в поле $GF(2^n)$. Допустим, что в поле $GF(2^n)$ построен GNB с генератором α порядка $e \geq (2^n - 1)/n$. Положим $d = (2^n - 1)/e$. Представим d в виде $d_1 d_2$, где $(d_2, e) = 1$, и каждый простой делитель d_1 делит e. Найдем $\beta_i \in GF(2^n)$, такие, что $\beta_1^{d_1} = \alpha, \beta_2^{d_2} = 1$. Это можно сделать детерминированно со сложностью $n^{7/2+\epsilon_n}$ согласно [105], разлагая двучлены на множители. Очевидно β_1 имеет порядок ed_1, а значит $\beta_1 \beta_2$ согласно известной лемме имеет порядок $ed_1 d_2 = 2^n - 1$, так как $(ed_1, d_2) = 1$. Поэтому $\beta_1 \beta_2$ является примитивным элементом. Вычисляя минимальный многочлен этого элемента, получаем примитивный многочлен.

Так часто оказывается $e = 3, 7, 15$ и т.д., то для практического ускорения алгоритма полезно использовать следующий факт.

Упражнение 2.5.17 Если $d = p^k - 1$, то решение уравнения $x^d = \alpha$ в поле $GF(p^n)$ сводится к решению линейной системы над $GF(p)$ с n неизвестными.

Указание. Уравнение $x^d = \alpha$ равносильно уравнению $x^{p^k} = \alpha x$, которое в нормальном базисе (да и в стандартном тоже) записывается в виде линейной системы над $GF(p)$ с n неизвестными. Если базис является GNB или вообще базисом низкой сложности и α — его элемент, то эта система имеет редкую матрицу, т.е. матрицу с $O(n)$ ненулевых элементов.

2.5.8 Редундантные базисы

Для произвольного поля $GF(q)$ рассмотрим наименьшее его расширение, содержащее корни n-й степени из единицы, другими словами поле разложения многочлена $x^n - 1$. Назовем это поле *циклотомическим* и обозначим $K(n)$. Для произвольного поля $GF(q^n)$ обозначим $K(m(n))$ наименьшее содержащее его циклотомическое поле. Это наименьшее поле, в котором можно выполнить ДПФ, не выполнимое в поле $GF(q^n)$.

Приведем таблицу значений функции $m(n)$.

n	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16
m	3	7	5	11	9	29	17	19	11	23	13	53	29	31	19

Поле $GF(q^n)$ рассматриваем как вложенное в поле $K(m(n))$. Пусть $\zeta \in K(m(n))$ примитивный корень степени $m(n)$ из единицы. Рассмотрим систему $\{\zeta^a, a = 0, \ldots, m(n) - 1\}$ и натянутое на нее подпространство в поле $K(m(n))$. Очевидно оно замкнуто относительно сложения и умножения, а значит, согласно малой теореме Ферма, и относительно инвертирования, т.е. оно образует подполе в поле $K(m(n))$. Это подполе является циклотомическим и совпадает поэтому с полем $K(m(n))$. Систему $\{\zeta^a, a = 0, \ldots, m(n) - 1\}$ назовем *редундантным базисом* в полях $GF(q^n)$ и $K(m(n))$.

В обычном смысле эти системы могут не являться базисами, так как могут быть линейно зависимыми. Введены они были в [?]. Примером редундантых базисов являются базисы вида $\{\zeta^a, a = 0, kn\}$, где $\zeta^p = 1$, $\zeta \in GF(q^d)$, где d порядок элемента q в поле $GF(p), p = kn + 1$, которые уже появлялись у нас при изучении GNB. Очевидно, что $K(p) = GF(q^d)$, а так как d кратно n, то $m(n) \leq p$. При $k = 1$ очевидно $d = n$, $m(n) = p$. Так как многочлен f_n неприводим, то $\{\zeta^a, a = 0, n - 1\}$, где $\zeta^p = 1$, линейно независимы и образуют базис в поле $GF(q^n)$. Система $\{\zeta^a, a = 0, n\}$ является редундантным базисом.

Разложение элемента поля $GF(q^n)$ или поля $K(m(n))$ по редундантному базису *неоднозначно*. Умножение в этом базисе очевидно совпадает с умножением в кольце многочленов степени $m(n)$ по модулю $x^{m(n)} - 1$. В [164] отмечается, что это умножение удобно для схемной реализации, так как совпадает с циклической сверткой.

В [100] было показано, что если существует GNB типа k в поле $GF(q^n)$, существует изоморфное отображение поля $GF(q^n)$ в кольцо $R_{f_{p-1}} = GF(q)[x]/f_{p-1}$ многочленов по модулю f_p, где $p = kn + 1, f_{p-1} = (x^p - 1)/(x - 1)$.

Фактически это мы доказали в предыдущем разделе. Но редундантный базис возникает фактически не в это кольце, а в кольце $R = GF(q)[x]/f$ многочленов по модулю $f = x^p - 1$, именно в нем умножение совпадет с циклической сверткой многочленов, причем редундантному базису соответствует стандартный базис $\{1, x, x^2, \ldots, x^{kn}\}$.

Докажем, что это кольцо содержит как подкольца кольцо $R_{f_{p-1}}$ и поле $GF(q^d)$. Действительно, круговой многочлен

$$f_p = 1 + x + \ldots, x^{p-1},$$

разлагается в произведение kn/d неприводимых многочленов степени d

$$f_p = \prod_{a \in L} m_{\zeta^a},$$

где $L \subset K$, m_{ζ^a} — минимальный аннулирующий многочлен элемента ζ^a. Значит разложение на неприводимые многочлены двучлена $x^p - 1$ над полем $GF(q)$ имеет вид

$$x^p - 1 = (x_1) \prod_{a \in L} m_{\zeta^a} = (x-1) \prod_i g_i,$$

Поэтому согласно алгебраической форме китайской теоремы об остатках кольцо R изоморфно произведению колец $R_{f_{p-1}}$ и $GF(q)$, а подкольцо кольца R , порожденное базисом $\{x, x^2, \ldots, x^{kn}\}$, изоморфно кольцу $R_{f_{p-1}}$ и может рассматриваться оносительно базиса $\{1, x, x^2, \ldots, x^{kn-1}\}$. Взаимный переход между этими базисами выполняется со сложностью $kn - 1$ и был описан в разделе об ONB.

Так как все g_i неприводимы и имеют степень d, то кольца R_{g_i} все изоморфны полю $GF(q^d)$. Очевидно, что кольцо $GF(q^d)^{nk/d}$ изоморфно произведению колец R_{g_i}, а значит и кольцу $R_{f_{p-1}}$, и может расматриваться как подкольцо кольца R. Так как диагональное отображение $x \to (x, x, \ldots, x)$ изоморфно вкладывает поле $GF(q^d)$ в кольцо $R_{f_{p-1}}$, то отсюда следует, что поле $GF(q^d)$ можно вложить (и многими разными вложениями) в подкольцо кольца R, порожденное базисом $\{x, x^2, \ldots, x^{kn}\}$, который можно отождествить с редундантным базисом $\{\zeta^a, a = 1, kn\}$, где $\zeta^p = 1$, $\zeta \in GF(q^d)$. Различным вложениям будут соответствовать разные разложения элемента поля $GF(q^d)$ по этому «базису». Нас интересуют вложения подполя $GF(q^n)$. Одно из них было указано при построении GNB. Действительно, в поле $GF(q^n)$ существует GNB типа k $\{\alpha_0, \ldots, \alpha_{n-1}\}$, где

$$\alpha_i = \sum_{a \in K_i} \zeta^a, i = 0, \ldots, n-1,$$

$K_i = q^i K, i = 0, \ldots, n-1$ —смежные классы по подгруппе K, порожденной элементом γ порядка k, и

$$\{1, \ldots, kn\} = K_1 \cup \ldots K_n.$$

Разлагая произвольный элемент a поля $GF(q^n)$ по этому базису

$$a = \sum_{i=0}^{n-1} a_i \alpha_i$$

и выражая этот базис через редундантный базис, получаем, что в последнем элемент a можно однозначно записать с соблюдением условия согласованности коэффициентов, а именно равенства всех пар коэффициентов c_i, c_j, если индексы i, j лежат в одном смежном классе по K. Переход между кооординатами в GNB и координатами в редундантном базисе делается бесплатно при выполнении условия согласованности. Можно вместо редундантного базиса взять рассматривавшийся базис $\{x, x^2, \ldots, x^{kn}\}$ в подкольце кольца R, и множество элементов этого подкольца, координаты которых в этом базисе удовлетворяют условию согласованности, образует подкольцо, изоморфное полю $GF(q^n)$. Для проверки изоморфности достаточно установить взаимнооднозначность естественного изоморфизма, и сохранение операций при нем, а это было доказано в разделе о GNB. Выполнение умножения в поле $GF(q^n)$ поэтому сводится к выполнению умножения в изоморфном ему подкольце кольца R относительно базиса $\{1, x, x^2, \ldots, x^{kn}\}$. Это умножение является циклической сверткой и выполняется весьма регулярной схемой сложности $2(kn+1)(kn+1)$ и глубины $\lceil \log_2 2kn \rceil$, как было отмечено в [164]. Если $q = 2$, то схема будет логической. Если учесть, что первую координтату можно считать нулевой, то число умножений уменьшается на $kn + 1$, и число сложений на kn. Но если учесть переход к базису $\{x, x^2, \ldots, x^{kn}\}$, то kn сложений придется добавить. Но если учесть, что среди nk координат элемента подкольца, изоморфного подполю, согласно условию согласованности только n различных (каждая из которых дублируется k раз), то число умножений в схеме уменьшается до n^2, а если учесть, что вместо kn координат достаточно

вычислить n, то и число сложений уменьшается до $n(kn-2)$, а может оказаться и меньше из-за возможной склейки слагаемых.

При $k=1$ формулы проще и их легко выписать в явном виде. При умножении $\sum\limits_{i=1}^{n} a_i x^i$ на $\sum\limits_{i=1}^{n} b_i x^i$ по модулю $x^p-1, p=n+1$, получаем многочлен $\sum\limits_{i=0}^{n} c_i x^i$, где

$$c_i = \sum_{j=1}^{n} a_j b_{i-j \bmod p} = i = 0, \ldots, n$$

где $b_0 = 0$, и каждая сумма содержит на самом деле $n-1$ слагаемое, кроме первой, в которой n слагаемых. Далее его переписываетм в виде

$$\sum_{i=1}^{n} C_i x^i, C_i = c_i + c_0 = \sum_{j=1}^{n} (a_j b_{i-j \bmod p} + a_j b_{p-j}).$$

Билинейная форма

$$C_1 = \sum_{j=2}^{n} a_j b_{1-j \bmod p} + \sum_{j=1}^{n} a_j b_{p-j}$$

содержит $2n-1$ слагаемых и остальные формы столько же. Так как базис $\zeta, \ldots, \zeta^n$ является перестановкой ONB первого типа, то полученная формула умножения эквивалентна формуле Месси-Омура для этого базиса. Но если воспользоваться предыдущей формулой, то сложность умножения уменьшается до $2n^2-1$за счет увеличения глубины на единицу. Указанная схема с точностью до перестановки переменных совпадает со схемой, полученной в [143].

Умножение многочленов степени n по модулю x^p-1 можно выполнить, используя трюк Карацубы. Пусть $n=2m$. Представляя

$$a(x) = a_0(x) + x^{m+1} a_1(x), b(x) = b_0(x) + x^{m+1} b_1(x),$$

и вычисляя произведение по модулю $x^{n+1}-1$ по формулам

$$c_0 = a_0 b_0 + a_1 b_1 x, c_1 = a_0 b_1 + a_1 b_0 = (a_0 + a_1)(b_0 + b_1) - c_0 = d_0 + d_1 x^m,$$

$$c(x) = c_0 + x^{m+1} c_1 \bmod x^{2m+1} - 1 = c_0 + d_1 + x^{m+1} d_0$$

со сложностью не более $M(m) + 2M(m+1) + 2m - 1 + m + m + 2m + 2m = M(m) + 2M(m+1) + 8m - 1$, где $M(m)$ сложность умножения многочленов степени $m-1$.

Рассмотрим случай $k=2$. В этом случае базис $\zeta, \ldots, \zeta^{kn}$ соотвествует ONB второго или третьего типа, в зависимости от порядка элемента q в поле $GF(p)$. Если порядок d равен $2n$, то получается второй тип, а если $d=n$, то третий. В обоих случаях $\gamma = -1 \bmod p$, билинейная форма

$$C_1 = \sum_{j=2}^{2n} a_j b_{2n+2-j \bmod p} + \sum_{j=1}^{2n} a_j b_{2n+1-j}$$

содержит $2n-1+2n = 4n-1$ слагаемых.

В случае базиса второго типа α_i есть перестановка $\zeta^i + \zeta^{-i} = \zeta^i + \zeta^{2n+1-i}, i = 1, \ldots, n$ поэтому из $2n$ координат произведения достаточно вычислить только ту половину, которая имеет номера от 1 до n, и в формуле для C_1 надо отождествить переменные по правилу $a_i = a_{2n+1-i}, b_i = b_{2n+1-i}$. Тогда билинейная форма принимает вид

$$C_1 = \sum_{j=2}^{n} a_j b_{j-1} + \sum_{j=1}^{n} a_{n+1-j} b_{n+2-j} + \sum_{j=1}^{n} a_j b_j + \sum_{j=1}^{n} a_j j b_j =$$

$$= \sum_{j=2}^{n} a_j b_{j-1} + \sum_{j=1}^{n} a_{n+1-j} b_{n+2-j}.$$

$$= \sum_{j=2}^{n} a_j b_{j-1} + \sum_{j=1}^{n-1} a_j b_{j+1} + a_n b_n.$$

Она состоит из $2n-1$ слагаемых. Так как выбранные координаты совпадают с координатами в ONB, то остальные билинейные формы получаются из первой циклическим сдвигом переменных. Получились те же самые формулы умножения в ONB Месси-Омура. В случае базиса третьего типа верно то же самое, но выбор номеров координат, которые надо вычислять, и правило отождествления переменных более сложны.

2.5.9 Пример схемного инвертирования с использованием редундантного базиса

Как отмечалось выше, для умножения в поле $GF(2^n)$, где 2 является примитивным элементом в поле $GF(p), p = n+1$ можно воспользоваться умножением в кольце многочленов по модулю $x^p - 1$, невзирая на неоднозначность представления в таком виде. Для перехода к однозначному представлению достаточно выполнить преобразование сложности n и глубины 1:

$$A_i = a_i + a_0, i = 1, \ldots n.$$

Умножение в этом редундантном базисе является, как отмечалось, циклической сверткой, и выполняется с мультипликативной сложностью p^2 и аддитивной сложностью $p(p-1)$. Эта сложность несколько выше сложности умножения в стандартном базисе с многочленом $x^n + \ldots + 1$ и в соответствующем ONB, однако глубина на единицу меньше, так как равна $1 + \lceil \log_2 p \rceil$. А так как возведение в квадрат в этом базисе сводится к перестановке координат и при схемной реализации выполняется бесплатно, то выполнение инвертирования в этом базисе по формуле

$$x^{-1} = x^{2^n - 2}$$

можно реализовать схемой глубины

$$d(n-1)(1 + \lceil \log_2 p \rceil) + 1,$$

где $d(n-1) = \lceil \log_2(n-1) \rceil$ — минимальная глубина аддитивной цепочки для $n-1$, у которой все шаги линейны, кроме удвоений.

Упражнение 2.5.18 Докажите это утверждение.

Например, для $n = 28$ цепочкой минимальной сложности 6 является $1, 2, 3, 6, 9, 18, 27$ но цепочка большей сложности $1, 2, 3, 4, 8, 11, 16, 27$ линейной имеет меньшую глубину 5. По ней можно построить цепочку для вычисления $2^{27} - 1$, в которой все неудваивающие шаги имеют вид

$$1, 2^2 - 1, 2^3 - 1, 2^4 - 1, 2^8 - 1, 2^{11} - 1, 2^{16} - 1, 2^{27} - 1$$

и которая также имеет глубину 5, если игнорировать удваивающие шаги.

Поэтому глубина инвертирующей схемы равна $5 \cdot 6 + 1 = 31$. Единица добавляется потому, что результат надо привести к виду в стандартном или нормальном базисе. Сложность схемы равна $(d(n-1) + \nu_2(n-1) - 2)(n+1)(2n+1) = 11571$. Заметим, однако, что для некоторого неоптимального нормального базиса можно построить для поля $GF(2^{28})$ схему глубины 27 и втрое меньшей сложности.

2.5.10 Пример схемы умножения в GNB

Рассмотрим представление поля $GF(2^{13})$ в редундантном базисе $\zeta^i, i = 1, \ldots, 52, \zeta^{53} = 1$, т.е. в виде многочлена

$$\sum_{i=1}^{52} a_i \zeta^i$$

из кольца R многочленов по модулю $x^{53}+1$ с условиями согласования коэффициентов: $a_i = a_j$, если $i \in jK$, где $K = \{1, 30, 52, 23\} = \{1, -23, -1, 23\}$ — подгруппа четвертого порядка в группе $GF(53)\{0\}$. Так как выше было найдено, что

$$K = \{1, -23, -1, 23\}, K_1 = \{2, -2, 7, -7\}, K_2 = \{4, -4, 14, -14\}, K_3 = \{8, -8, 25, -25\},$$

$$K_4 = 2^4 K = \{16, -16, 3, -3\}, K_5 = 2^5 K = \{21, -21, 6, -6\}, K_6 = 2^6 K = \{11, -11, 12, -12\},$$

$$K_7 = 2^7 K = \{22, -22, 24, -24\}, K_8 = 2^8 K = \{9, -9, 5, -5\}, K_9 = 2^9 K = \{18, -18, 10, -10\},$$

$$K_{10} = 2^{10} K = \{17, -17, 20, -20\}, K_{11} = 2^{11} K = \{19, -19, 13, -13\},$$

$$K_{12} = 2^{12} K = \{15, -15, 26, -26\}$$

есть разбиение группы $GF(53)\{0\}$ на непересекающиеся смежные классы, то

$$a_1 = a_{23} = a_{30} = a_{52}, a_2 = a_7 = a_{46} = a_{51},$$

$$a_3 = a_{16} = a_{37} = a_{50}, a_4 = a_{14} = a_{39} = a_{49},$$

$$a_5 = a_9 = a_{44} = a_{48}, a_6 = a_{21} = a_{32} = a_{47},$$

$$a_8 = a_{25} = a_{28} = a_{45}, a_{10} = a_{18} = a_{35} = a_{43},$$

$$a_{11} = a_{12} = a_{41} = a_{42}, a_{13} = a_{19} = a_{40} = a_{34},$$

$$a_{15} = a_{26} = a_{27} = a_{38}, a_{17} = a_{20} = a_{33} = a_{36},$$

$$a_{22} = a_{24} = a_{29} = a_{31}.$$

Умножим этот многочлен на подобный же многочлен $b(x)$ по модулю $x^{53} + 1$, т.е. выполним циклическую свертку по формулам

$$c_i = \sum_{j=0}^{52} a_j b_{i-j \bmod 53}, i = 0, \ldots, 52,$$

формально положив $a_0 = b_0 = 0$, и удалим свободный член, выполнив приведение по модулю $1 + x + \ldots x^{52}$. Новые коэффициенты при этом будут вычисляться по формулам

$$C_i = c_i + c_0, i = 1, \ldots, 52.$$

Непосредственно можно проверить, что они удовлетворяют условию согласованности, поэтому различных среди них будет только 13. Оставляя только их, получаем правило умножения в GNB типа 4 в поле $GF(2^{13})$. Можно непосредственно проверить, что оно совпадает с полученным ранее другим способом правилом умножения, а именно с правилом Месси-Омуры

$$z_i = (i, i \oplus 1) + (i \oplus 1, i \oplus 3) + (i \oplus 1, i \oplus 4) + (i \oplus 1, i \oplus 5) + (i \oplus 2, i \oplus 4) + (i \oplus 2, i \oplus 8) +$$

$$+(i \oplus 2, i \oplus 11) + (i \oplus 2, i \oplus 12) + (i \oplus 3, i \oplus 7) + (i \oplus 3, i \oplus 8) + (i \oplus 3, i \oplus 12) +$$

$$+(i \oplus 4, i \oplus 12) + (i \oplus 4, i \oplus 10) + (i \oplus 5, i \oplus 7) + (i \oplus 5, i \oplus 8) + (i \oplus 5, i \oplus 10) +$$

$$+(i \oplus 6, i \oplus 9) + (i \oplus 6, i \oplus 11) + (i \oplus 8, i \oplus 9) + (i \oplus 9, i \oplus 10) +$$

$$+(i \oplus 9, i \oplus 11) + (i \oplus 10, i \oplus 11) + (i \oplus 12).$$

Сложность рассматриваемого GNB, как уже отмечалось, равна числу одночленных слагаемых в билинейной форме, т.е. $22 \cdot 2 + 1 = 45$. Аддитивная сложность реализации одной координаты равна 44, а аддитивная сложность всей схемы $13 \cdot 44$. Если учесть, что мультпликативная сложность очевидно равна 13^2, то тотальная сложность схемы равна

$$13 \cdot (44 + 13) = 741.$$

Глубина схемы равна 7. Далее будет указан метод построения более простой схемы.

2.5.11 Быстрое программное инвертирование

Здесь мы рассматриваем только случай $q = 2$, т.е. имеем дело только с полями $GF(2^n)$ и базисами в них.

Для любого стандартного базиса инвертирование можно выполнять с помощью EEA — расширенного алгоритма Евклида. Например, в [?] его применяют для поля $GF(2^{163}$ и стандартного базиса, порожденного корнем пятичлена

$$x^{163} + x^7 + x^6 + x^3 + 1,$$

так как это одно из полей и один из базисов, рекомендованных стандартом $NIST$.

На каждом шаге EEA вместо деления с отстатком можно выполнять вычитание делителя, умноженного на степень переменной. Деление с остатком в общем случае сводится к таким операциям, но их использование удобнее. Во всяком случае, например согласно [107] наиболее эффективной оказалась именно такая версия EEA. Его целью является нахождение для любых данных взаимно простых многочленов $a(x), f(x)$ таких многочленов $b(x), d(x)$, что $ba + df = 1$. Тогда очевидно $ab = 1 \bmod f$, т.е. $a^{-1} = b(\mathrm{mod} f$. Во время его работы вычисляются многочлены b, d, u, v, которые после любого шага алгоритма удовлетворяют соотношениям $ba = u \bmod f, ca = v \bmod f$. На первом шаге полагаем $b = 1, c = 0, u = a, v = f$. Очевидно указанные соотношения выполнены. На очередном шаге находим $j = \deg u - \deg v$ и если $j \geq 0$, то из u вычитаем $x^j v$, т.е. присваиваем $u = u + x^j v$ и аналогично полагаем $b = b + x^j c$. Тогда будут выполнены указанные соотношения, так как по модулю f новое значение

$$ba = ba + x^j ac = u + x^j ac = u + x^j v = u \bmod f.$$

Если же $j < 0$, то меняем местами u с v и b с c, меняем знак у j, т.е. присваиваем $j = -j$, и делаем для u, v те же присваивания, что и в случае $j \geq 0$. Тогда тоже будут выполнены указанные соотношения, так как

$$ca = ba = u = v \bmod f,$$

$$ba = ca = v = u \bmod f.$$

На каждом шаге алгоритма $\max\{\deg u, \deg v\}$ уменьшается по крайней мере на 1, за исключением случая $\deg u = \deg v$, а в среднем, согласно [107], на 2. Алгоритм заканчивает работу, когда $\deg u = 0$, т.е. $u = 1$. Тогда очевидно $ba = 1, b = a^{-1} \bmod f$. Заметим, что если $\deg u > \deg v$, то на очередном шаге уменьшается $\deg u$, но не меняются ни v ни c.

Обозначим d_i степени многочленов u в те моменты, когда $\deg u < \deg v$ (после чего u и v меняются местами и опять становится $\deg u > \deg v$). Положим $d_0 = n$. В самом начале $\deg v = n > \deg a = \deg u$, поэтому $d_1 = \deg u < n$. В этот момент $c = 1, b = 0$ и на следующем шаге $\deg b = d_0 - \deg v = d_0 - d_1 > \deg c = 0$ и эта степень не меняется до тех пор, пока не поменяются местами u и v во второй раз. После этого меняются местами b, c и $\deg b = 0$, но уже на следующем шаге $\deg b = d_0 - d_1 + d_1 - d_2 = d_0 - d_2$, так как в это момент $\deg u - \deg v = d_1 - d_2$.

Продолжаяя эти рассуждения далее, видим, что всегда $\deg b \le d_0 < n$, поэтому приведение по модулю f выполнять не надо.

Единственная операция умножения, используемая в этом алгоритме, фактически означает сдвиг массива коэффициентов на j битов вправо и выполняется быстро. Еще большего ускорения можно достичь, если коэффициенты многочленов хранить не в виде массивов битов, а в виде массивов машинных слов — w-битных целых чисел ($w = 32, 64$). Но тогда операция сдвига массива будет включать в себя не только операции пересылки, но и операции выделения префикса и постфикса данной длины в машинном слове и операции побитового XOR.

В [150] был предложен «почти инверсный» алгоритм AIA, который производит впечатление более быстрого. В нем вначале находится такой многочлен $b(x)$, что $b(x)a(x) = x^k \bmod f$, а потом вычисляется a^{-1} как $b(x)g(x)$, где $g(x)x^k = 1 \bmod f$, т.е. $g(x) = x^{-k} \bmod f$. При этом всегда $k \le 2n - 1$. Для нахождения b вначале полагаем $b = 1, c = 0, u = a, v = f, k = 0$. Во время работы программы всегда

$$b(x)a(x) = u(x)x^k \bmod f, c(x)a(x) = v(x)x^k \bmod f.$$

Пока $u(x)$ делится на x выполняем присваивания $u(x) = u(x)/x, c(x) = c(x)x, k = k + 1$. Если $u = 1$ алгоритм заканчивает работу и возвращает результат — многочлен $b(x)$ и число k. В противном случае работа продолжается и если $\deg u < \deg v$, то меняются местами u и v, а потом b и c. Далее, независимо от того, выполнялась эта перестановка или нет, выполняются присваивания $u = u + v, b = b + c$ и делается переход к началу цикла деления на x, причем для ускорения работы проверка выполнения условия делимости на x не делается, так как это условие всегда выполнено. Операция прибавления 1 к k делается не более $2n - 1$ раз.

Упражнение 2.5.19 Докажите это, а также инвариантнсть соотношений

$$b(x)a(x) = x^k u \bmod f, c(x)a(x) = x^k v \bmod f$$

во время работы алгоритма.

Операция деления на x в этом алгоритме означает просто сдвиг массива влево на один бит. Как и в AEA, здесь тоже надо работать не с битами, а с машинными словами.

Алгоритм AIA похож на AEA, только удаление битов в u и v в нем выполняется, начиная с младших битов. Но еще более похож он бинарный алгоритм Евклида для чисел, описанный в [39]. Для того, чтобы в этом убедится, назовем многочлен четным, если его свободный член равен нулю, и нечетным в противном случае. Тогда деление четного многочлена на x соответствует делению четного числа на 2, умножение же любого многочлена на x соответствует умножению числа на 2.

После получения $b(x)$, такого что $b(x)a(x) = x^k \bmod f$, надо найти $g(x), \deg g < n$, такой что $g(x)x^k = b(x) \bmod f$, тогда $a^{-1} = g$. Для выполнения этого в [150] предложена следующая процедура. Определим по данному неприводимому многочлену f число $s = \min\{i \ge 1, f_i = 1\}$. Тогда $f(x) = 1 + x^s h(x)$. Например, для используемого в [107]

$$x^{163} + x^7 + x^6 + x^3 + 1$$

число $s = 3$, а для используемого в [150]

$$x^{155} + x^{62} + 1$$

число $s = 62$. Выделим в $b(x)$ s младших битов и обозначим $b'(x)$ определяемый ими многочлен, тогда $b(x) = b'(x) + x^s d(x)$. Поэтому $b'(x)f(x) + b = x^s(d(x) + b'(x)h(x)$ делится на x^s (имеет нулевые s младших битов), значит $b''(x) = (b'(x)f(x) + b)/x^s$ имеет степень меньше n и поэтому $b''(x) = b(x)x^{-s} \bmod f$, так как $b''(x)x^s = b(x) \bmod f$. Повторяя эту процедуру

$\lceil k/s \rceil$ раз (в последний раз вместо s берется $k-(\lceil k/s \rceil - 1)s$), получаем $a(x) = b(x)x^{-k} \bmod f$. Умножение $b'(x)$ на $f(x)$ быстрая процедура только если $f(x)$ — малочлен. Но именно они и используются в рассматриваемых примерах. Время вычисления $b(x)x^{-k} \bmod f$ тем меньше, чем больше s, поэтому во втором примере оно конечно меньше, чем в первом. В [150] рекомендуют выбирать $s = 32$, а не 62. Число повторений операции $b(x)x^{-s} \bmod f$ при этом возрастает иногда почти в два раза, но сама операция выполняется проще, так как

$$d(x) + b'(x)h(x) = d(x) + b'(x)x^{62-s} + b'(x)x^{155-s},$$

и хотя здесь приходится делать два сдвига, а не один, зато b' здесь представляется одним машинным словом.

Для ускорения вычисления $a(x) = b(x)x^{-k} \bmod f$ при малых s в [107] предложили включить ее непосредственно в алгоритм AIA, а именно, после того, как в цикле было выполнено деление $u(x)$ на x, выполняется деление $b(x)$ на x, а если оно невозможно, то деление $b(x)+f(x)$ на x, умножение c на x не выполняется, а переменная k в программе не используется. Тогда вместо соотношений

$$b(x)a(x) = x^k u \bmod f, c(x)a(x) = x^k v \bmod f$$

инвариантыми будут соотношения

$$b(x)a(x) = u \bmod f, c(x)a(x) = v \bmod f$$

Упражнение 2.5.20 Докажите это.

В [107] проведено сравнение этих алгоритмов, и выяснилось, что при малых s действительно модифицированный AIA работает немного быстрее, чем AIA, но для многочлена

$$x^{233} + x^{74} + 1$$

AIA, как и ожидалось, работает чуть быстрее.

Удивительно то, что согласно [107] AEA во всех трех рассмотренных там примерах работал чуть быстрее, чем AIA. В [150] подобное сравнение не проводилось, однако там подчеркивается важность применения для ускорения работы программы нескольких чисто программистских трюков. Среди них следующие: использование для хранения машинных слов, определяющих многочлены u, v, b, c не массивов, а индивидуальных переменных; вместо обмена значениями u, v b, c требующего 6 операций пересылки, пишется вторая копия кода с копиями соответствующих переменных, и когда требуется сделать указанный обмен, просто делается прыжок в эту копию; так как число машинных слов, требующихся для хранения многочленов u, v убывает, а для многочленов b, c возрастает, то для выполнения операций с ними пишутся несколько копий кода со всеми возможными вариантами числа используемых переменных для хранения машинных слов, и в тот момент, когда появляется необходимость в использовании новой переменной, делается переключение на выполнение соответствующей копии кода, причем эти копии пишутся без использования циклов.

Некоторого ускорения можно добится используя GNB при малых k. Выше было показано, что инвертирование в GNB сводится к инвертированию в кольце R, т.е. к инвертированию по модулю многочлена $f = x^p - 1$, которое можно выполнить как с помощью EEA, так и с помощью AIA. В последнем случае D.Dahm заметил, что при использовании ONB (т.е. GNB типа $k = 1, 2$) вычисление $a(x) = b(x)x^{-k} \bmod f$ делается просто с помощью циклического сдвига, так как $x^{-k} \bmod f = x^{p-k} \bmod f$, поэтому

$$a(x) = b(x)x^{-k} = b(x)x^{p-k} \bmod f = \sum_{i=0}^{p-1} b_i x^{i+p-k \bmod p} \bmod f = \sum_{i=0}^{p-1} b_i x^{i-k \bmod p} \bmod f =$$

$$= \sum_{i=0}^{p-1} b_{i+k \bmod p} x^i \bmod f.$$

В случае $k = 1$ несколько позднее то же самое было отмечено в [155]. Мы покажем, что это трюк можно применять для любого GNB. Для этого заметим, что инвертирование любого элемента a в базисе GNB типа k при записи его в виде многочлена $a(x)$ степени nk по модулю $x^p - 1$, как отмечалось выше, начинается с его отображения в подкольцо кольца R, изморфное рассматриваемому полю $GF(q^n)$. Это подкольцо состоит из всех многочленов степени nk с нулевым свободным членом, коэффициенты которых удовлетворяют условию согласованности; переход от элемента a к многочлену $a(x)$ и обратно выполняется бесплатно (для обратного перехода это верно только при выполнении условия согласованности). Ранее было показано, что многочлен $a(x)$ будет обратимым в этом кольце, значит будет взаимнопростым с $f(x)$. Обратный многочлен $a^{-1}(x)$ при этом определяется однозначно и его можно найти с помощью AIA и указанного выше циклического сдвига. Многочлен, соотвествующий обратному элементу a^{-1}, удовлетворяющий условию согласованности коэффициентов и имеющий нулевой свободный член, очевидно будет обратен многочлену $a(x)$ в кольце R, согласно отмеченному выше изоморфизму, и значит совпадет с многочленом $a^{-1}(x)$ согласно отмеченной выше единственности. Поэтому многочлен $a^{-1}(x)$ удовлетворяет условию согласованности коэффициентов и имеет нулевой свободный член, что не было очевидно заранее. Выполняя бесплатный переход из кольца R в поле $GF(q^n)$, мы получаем искомый обратный элемент a^{-1} в этом поле.

В поле $GF(2^{163})$ существут GNB типа 4. Возможно, его применение вместе с AIA ускорит инвертирование в сравнении с описаннымм выше применением стандартного базиса, однако надо иметь в виду, что степени используемых в этом алгоритме многочленов могут возрасти вплоть до четырех раз.

Более эффективно с этой целью использование ONB второго типа для поля $GF(2^{191}$. В этом случае степени используемых многочленов возрастают только в два раза. Но, как было показано выше, умножение в этом базисе можно свести к умножению по модулю 15-члена

$$f(x) = (1 + x^{2^7})h + x^{2^6}, h = 1 + x^{2^5} + x^{2^5+2^4} + x^{2^5+2^4+2^3} + \ldots + x^{2^6-1}$$

степени 191 и может быть выполнено почти с той же скоростью, что и умножение по модулю $x^{191} + \ldots + 1$.

Зато возведение в квадрат в ONB делается быстро, а возведение в степень 32 еще быстрее, так как требует только циклического сдвига машинных слов.

Ранее было также показано, что частное от деления x^{383} на f равно

$$r(x) = (1 + x^{2^7})h + x^{192}$$

и приведение по модулю f сводится к умножению на 15-член r (с вычислением только старших коэффициентов начиная с члена x^{383}). Умножение многочленов можно выполнить, применив однократную или двукратную итерацию Карацубы и перемножив многочлены малой степени, например методом, изложенным в разделе Умножение на h требует 6 сдвигов (один из них на 32, поэтому он очень простой) и 7 покомпонентных сложений векторов коэффициентов. Избавиться от лишних пересылок при сдвигах можно, представив $h = 1 + x^{2^5} g$, сначала выполнив умножение на g, потом сдвиг на 32 позиции (означающий просто сдвиг массива машинных слов на единицу), и потом выполнив сложение векторов, состоящих из машинных слов. Умножение на $1 + x^{2^7}$ сводится к сдвигу на величину, кратную $w = 32$, что делается просто, и к сложению. Умножение на x^{192} сводится к сдвигу коэффициентов на 192, после чего делается последнее сложение, в котором надо складывать только биты с номерами большими 382

Другой вариант организации умножения в поле $GF(2^{191})$ основан на описанном выше сведении к умножению в кольце R по модулю $x^{383} - 1$. Степени перемножаемых многочленов

здесь вдвое больше, что при использовании даже метода Карацубы увеличивает время работы втрое, зато сводятся почти к нулю накладные расходы и упрощается программирование. Как отмечалось выше, некоторое количество сложений в методе Карацубы экономится благодаря совместимости его с циклической сверткой. Возведение в квадрат в этом кольце сводится к перестановке координат, но выполнять его все же лучше обычным путем возведения в квадрат с использованием предвычисленной таблицы и выполнения сдвига на 383 позиции и сложения. Эти операции тоже программируются проще, чем возведение в квадрат по модулю пятичлена и даже трехчлена, но из-за двойного увеличения степени и соответственно числа используемых машинных слов, возможно, все же эта программа будет работать медленнее, чем возведение в квадрат по модулю трехчлена.

Заметим еще наконец, что при использовании ONB в поле $GF(2^{191})$ можно и для инвертирования воспользоваться переходом к стандартному базису с неприводимым 15-членом многочленом

$$f(x) = (1 + x^{2^7})h + x^{2^6}, h = 1 + x^{2^5} + x^{2^5+2^4} + x^{2^5+2^4+2^3} + \ldots + x^{2^6-1}.$$

Тогда вычисление $x^{-k} \bmod f$ не сводится к циклическому сдвигу, но все же делается довольно быстро. Действительно, при $k \leq 2^5$

$$x^{-k} \bmod f = (f(x) - 1)/x^k = x^{2^5-k} + x^{2^5+2^4-k} + x^{2^5+2^4+2^3-k} + \ldots + x^{2^6-1-k} + x^{2^6-k} + x^{2^7-k}h,$$

$$h = 1 + x^{2^5} + x^{2^5+2^4} + x^{2^5+2^4+2^3} + \ldots + x^{2^6-1}.$$

Поэтому умножение на $x^{-k} \bmod f$ сводится к нескольким сдвигам битового массива коэффициентов и нескольким сложениям, подобному тому, как выполнялось подобное умножение выше; приведение результата по модулю f делается также. В случае $383 \geq k > 2^5$ умножение на x^{-k} по модулю f сводится не более чем к 12-катному выполнению операции умножения на x^{-32}; эту операцию можно в несколько раз ускорить, если заранее вычислить при $k = 1, 2, 3$

$$x^{-32 \cdot 2^k} \bmod f = 1 + x^{2^{4+k}} + x^{2^{4+k}+2^{3+k}} + \ldots + x^{2^{5+k}-2^k} + x^{2^{5+k}} + x^{3 \cdot 2^{5+k}} h^{2^k} \bmod f,$$

$$h^{2^k} = 1 + x^{2^{5+k}} + x^{2^{5+k}+2^{4+k}} + x^{2^{5+k}+2^{4+k}+2^{3+k}} + \ldots + x^{2^{6+k}-2^k} \bmod f.$$

2.5.12 Деление с помощью алгоритма Евклида

Удивительно, но только в 2000 году одновременно несколькими авторами (например, [151]) было замечено, что с помощью алгоритма Евклида можно сразу выполнять деление, а не только инвертирование, а потом умножение. Действительно, если мы хотим вычислить $e/b \bmod f$, например в начале работы обычного расширенного алгоритма Евклида на первом шаге полагаем $b = e$ вместо $b = 1$. Тогда вычисляемые во время его работы многочлены b, d, u, v, которые после любого шага алгоритма вместо соотношений $ba = u \bmod f, ca = v \bmod f$ будут удовлетворять соотношениям $ba = ue \bmod f, ca = ve \bmod f$. Останавливая этот вариант алгоритма, как обычно, когда $u = 1$, получаем, что $ba = e \bmod f$, откуда $b = e/a \bmod f$. Подобным же образом модифицируются и оба варианта бинарного алгоритма Евклида. Например, в почти инверсном варианте бинарного алгоритма вначале тоже полагаем $b = e$ вместо $b = 1$ и тогда во время работы программы всегда

$$b(x)a(x) = e(x)u(x)x^k \bmod f, c(x)a(x) = e(x)v(x)x^k \bmod f,$$

и заканчивается она, как обычно, когда $u = 1$, тогда $b(x)a(x) = e(x)x^k \bmod f$, откуда $b(x) = x^k e(x)/a(x) \bmod f$. После этого, как обычно, надо вычислить откуда $b(x)x^{-k} \bmod f$.

Разумеется, подобным же образом можно модифицировать и соответствующие алгоритмы для чисел.

Так как в криптографических приложениях обычно инвертирование входит в состав деления, то указанные модификации дают экономию времени, тем большую, чем меньше соотношение между между временем чистого инвертирования и чистого умножения. Впрочем, в противовес распространенному мнению что инвертирование раза в три медленее умножения, в [95] опять утверждается, что инвертирование в шесть, а то и в десять раз медленнее умножения, и и обычный расширенный алгоритм Евклида даже для поля $GF(2^{233})$ с базисом, порожденным трехчленом, чуть быстрее почти инверсного алгоритма на Pentium III (800 MHz), и лишь на SPARC (500MHz) чуть медленнее. Последнее обстоятельство объясняется [95] тем, что удобная для быстрого вычисления степени многочлена в расширенном алгоритме ассемблерная команда Pentium III для нахождения наибольшего значащего бита в машинном слове на процессоре SPARC заменена командой, которая сканирует биты не слева, а справа. В [95] также отмечается, что модификации для деления бинарных вариантов алгоритма работают медленнее, так как длина массива для хранения многочлена $b(x)$ сразу предполагается большой, а не малой, но растущей, как раньше.

2.5.13 Усовершенствованное умножение методом Месси-Омура

Рассмотрим в поле $GF(2^3$ оптимальный нормальный базис второго типа $\{\alpha, \alpha^2, \alpha^4\}$, где $1 + \alpha^2 + \alpha^3 = 0$ То, что это такой базис, можно проверить непосредственно. Заметим, что

$$\alpha + \alpha^2 + \alpha^4 + 1 = (1 + \alpha^2 + \alpha^3)(1 + \alpha) = 0,$$

откуда можно вывести линейную независимость системы $\{\alpha, \alpha^2, \alpha^4\}$, опираясь на линейную независимость системы $\{1, \alpha, \alpha^2\}$ и наоборот. Заметим еще

$$\alpha^7 + 1 = (\alpha + 1)(1 + \alpha + \alpha^2 + \alpha^3 + \alpha^3 + \alpha^4 + \alpha^5 + \alpha^6 =$$

$$(\alpha + 1)(1 + \alpha^2 + \alpha^3)(1 + \alpha + \alpha^3) = 0, \alpha^7 = 1, \alpha^8 = \alpha,$$

$$\alpha^3 = 1 + \alpha^2 = \alpha^4 + \alpha, \alpha^6 = (\alpha^3)^2 = \alpha^8 + \alpha^2 = \alpha + \alpha^2, \alpha^5 = \alpha^3\alpha^2 = (1 + \alpha^2)\alpha^2 = \alpha^2 + \alpha^4,$$

откуда непосредственно выводится формула умножения в этом базисе

$$(x_0\alpha + x_1\alpha^2 + x_2\alpha^4)(y_0\alpha + y_1\alpha^2 + y_2\alpha^4) = ((x_0y_1 + x_1y_0) + (x_1y_2 + x_2y_1) + x_2y_2)\alpha +$$

$$+((x_0y_2 + x_2y_0) + (x_1y_2 + x_2y_1) + x_0y_0)\alpha^2 + +((x_0y_1 + x_1y_0) + (x_0y_2 + x_2y_0) + x_1y_1)\alpha^4.$$

Эти формулы состоят из трех квадратичных форм, получающихся друг из друга циклическим сдвигом переменных. Общее число слагаемых в каждой из них равно $5 = 2 \cdot 3 - 1$, поэтому указанный базис является оптимальным.

Так как каждая из скобок с двумя слагаемыми встречается по два раза, нужно реализовать только три такие скобки со сложностью 9 и еще три отдельных произведения со сложностью 3, а потом выполнить еще 6 сложений по модулю два. Поэтому сложность схемы умножения равна 18, а глубина равна 4. Непосредственное применение схемы Месси-Омура дает оценку лишь $3 \cdot (3 + 3 + 2) = 33$. В обоих схемах число умножений равно 9.

Пользуясь симметрией матриц A рассматриваемых квадратичных форм, число умножений можно уменьшить до 6, не меняя общей сложности и глубины. Для этого введем обозначения

$$[i, j] = (x_i + x_j)(y_i + y_j), (i, j) = (x_iy_j + x_jy_i), (i) = x_iy_i,$$

тогда очевидно

$$[i, j] = (i, j) + (i) + (j),$$

откуда

$$((x_0y_1 + x_1y_0) + (x_1y_2 + x_2y_1) + x_2y_2) = (0,1) + (1,2) + (2) = [0,1] + [1,2] + (0),$$

$$((x_0y_2 + x_2y_0) + (x_1y_2 + x_2y_1) + x_0y_0) = (0,2) + (1,2) + (0) = [0,2] + [1,2] + (1),$$

$$((x_0y_1 + x_1y_0) + (x_0y_2 + x_2y_0) + x_1y_1) = (0,1) + (0,2) + (1) = [0,1] + [0,2] + (2)$$

(последние два тождества можно не проверять, так как они следуют из первого после соответствующего циклического сдвига координат). У этой схемы очевидно сложность и глубина будет такой же, но число умножений будет равно 6, так как вычисление $[i,j]$ в отличие от вычисления (i,j) требует одного умножения и двух сложений, а не наоборот.

Рассмотрим в поле $GF(2^{3n}$ оптимальный нормальный базис $\{\alpha, \alpha^2, \alpha^4\}$. Для сложности и глубины получаются оценки (при n не кратном трем)

$$L(M(3n)) \leq 6L(M(n)) + 12n, D(M(3n)) \leq D(M(n)) + 3.$$

Рассмотрим в поле $GF(2^4$ оптимальный нормальный базис первого типа $\{\alpha, \alpha^2, \alpha^4, \alpha^8\}$, где α корень многочлена $1 + x + x^2 + x^3 + x^4$. То, что это такой базис, можно проверить непосредственно. Заметим, что

$$\alpha^5 + 1 = (1 + \alpha + \alpha^2 + \alpha^3 + \alpha^4)(1 + \alpha) = 0,$$

$$\alpha^8 = \alpha^3\alpha^5 = \alpha^3,$$

откуда следует, что система $\{\alpha, \alpha^2, \alpha^4, \alpha^8, \}$, является перестановкой базиса $\{\alpha, \alpha^2, \alpha^3, \alpha^4\}$. Заметим еще

$$\alpha\alpha^2 = \alpha^3 = \alpha^8,$$

$$\alpha\alpha^4 = \alpha^5 = 1 = \alpha^4 + \alpha^3 + \alpha^2\alpha = \alpha + \alpha^2 + \alpha^4 + \alpha^8,$$

$$\alpha\alpha^8 = \alpha^9 = \alpha^4\alpha^5 = \alpha^4,$$

откуда непосредственно выводится формула Месси-Омура умножения в этом базисе:

$$(a_0\alpha + a_1\alpha^2 + a_2\alpha^4 + a_3\alpha^8)(b_0\alpha + b_1\alpha^2 + b_2\alpha^4 + b_3\alpha^8) =$$

$$(a_1b_2 + a_2b_1 + a_0b_2 + a_2b_0 + a_1b_3 + a_3b_1 + a_3b_3)\alpha +$$

$$(a_2b_3 + a_3b_2 + a_1b_3 + a_3b_1 + a_2b_0 + a_0b_2 + a_0b_0)\alpha^2 +$$

$$(a_3b_0 + a_0b_3 + a_2b_0 + a_0b_2 + a_3b_1 + a_1b_3 + a_1b_1)\alpha^4 +$$

$$(a_0b_1 + a_1b_0 + a_3b_1 + a_1b_3 + a_0b_2 + a_2b_0 + a_2b_2)\alpha^8$$

Эта формула состоят из четырех квадратичных форм, получающихся друг из друга циклическим сдвигом переменных. Общее число слагаемых в каждой из них равно $7 = 2 \cdot 4 - 1$, поэтому указанный базис является оптимальным.

Для более краткой записи этих форм используем обозначения

$$[i,j] = (x_i + x_j)(y_i + y_j), (i,j) = (x_iy_j + x_jy_i), (i) = x_iy_i,$$

тогда очевидно

$$(a_1b_2 + a_2b_1 + a_0b_2 + a_2b_0 + a_1b_3 + a_3b_1 + a_3b_3) =$$

$$= (1,2) + (0,2) + (1,3) + (3)[1,2] + [0,2] + [1,3] + (0),$$

$$(a_2b_3 + a_3b_2 + a_1b_3 + a_3b_1 + a_2b_0 + a_0b_2 + a_1b_1) =$$

$$= (2,3) + ((1,3) + (2,0)) + (1)[2,3] + ([1,3] + [2,0]) + (1),$$

$$(a_3b_0 + a_0b_3 + a_2b_0 + a_0b_2 + a_3b_1 + a_1b_3 + a_2b_2) =$$

$$= (3,0) + ((2,0) + (3,1)) + (2)[3,0] + ([2,0] + [3,1]) + (2),$$
$$(a_0b_1 + a_1b_0 + a_3b_1 + a_1b_3 + a_0b_2 + a_2b_0 + a_3b_3) =$$
$$= (0,1) + ((3,1) + (0,2)) + (3)[0,1] + ([3,1] + [0,2]) + (3).$$

Эти формулы дают две разные схемы умножения одинаковой сложности 31 и глубины 4 так как сложность вычисления $[i,j]$ или (i,j) равна 3, а глубина равна 2, и сумму $(0,2)+(1,3)$ или сумму $[0,2]+[1,3]$ достаточно вычислить один раз. Но число умножений у схемы, построенной по формулам, содержащим $[i,j]$ будет меньше, а именно равно 10, так как вычисление $[i,j]$ в отличие от вычисления (i,j) требует одного умножения и двух сложений, а не наоборот.

Рассмотрим в поле $GF(2^{4n}$ оптимальный нормальный базис $\{\alpha, \alpha^2, \alpha^4, \alpha^8\}$, где α корень многочлена $1 + x + x^2 + x^3 + x^4$ в его подполе $GF(2^4)$.

Для сложности и глубины получаются оценки (при нечетном n)

$$L(M(4n)) \leq 10L(M(n)) + 21n, D(M(4n)) \leq D(M(n)) + 3.$$

Как было показано выше, в общем случае ONB первого типа после некоторой перестановки координат первая билинейная форма в формуле умножения Месси-Омура имеет вид

$$z_1 = \sum_{j=2}^{n} a_j b_{n+2-j} + \sum_{j=1}^{n} a_j b_{n+1-j}.$$

Чтобы ее можно было сравнить с полученными выше формулами, надо все индексы в ней уменьшить на единицу и привести к интервалу от 0 до $n-1$. Тогда она записывается в виде

$$z_0 = \sum_{j=1}^{n-1} a_j b_{n-j} + \sum_{j=0}^{n-1} a_j b_{n-1-j}.$$

В симметричной матрице этой формы очевидно в каждой строке стоят две единицы, кроме строки с номером $n/2$, в которой единица стоит только на диагонали.

Например, при $n = 4$ получим билинейную форму

$$z_0 = a_1b_3 + a_2b_2 + a_3b_1 + a_0b_3 + a_1b_2 + a_2b_1 + a_3b_0.$$

Очевидно, это форма получается из только что полученной формы

$$(a_1b_2 + a_2b_1 + a_0b_2 + a_2b_0 + a_1b_3 + a_3b_1 + a_3b_3)$$

транспозицией индексов 2 и 3.

В общем случае матрица A для ONB первого типа тоже в каждой строке имеет по две единицы, за исключением одной строки, где единица стоит на диагонали, и в силу симметрии то же верно и для столбцов. Далее будет показано, что тоже самое верно и для любого ONB.

Рассмотрим в поле $GF(2^5)$ оптимальный нормальный базис второго типа $\{\alpha, \alpha^2, \alpha^4, \alpha^8, \alpha^{16}\}$. где

$$1 + \alpha + \alpha^2 + \alpha^4 + \alpha^5 = 0.$$

Заметим, что (проверку для краткости опускаем)

$$\alpha\alpha^2 = \alpha^8 + \alpha, \alpha\alpha^8 = \alpha^4 + \alpha^2, \alpha\alpha^4 = \alpha^8 + \alpha^{16}, \alpha\alpha^{16} = \alpha^4 + \alpha^{16},$$

откуда непосредственно выводится формула Месси-Омура умножения в этом базисе:

$$(a_0\alpha + a_1\alpha^2 + a_2\alpha^4 + a_3\alpha^8 + a_4\alpha^{16})(b_0\alpha + b_1\alpha^2 + b_2\alpha^4 + b_3\alpha^8 + b_4\alpha^{16}) =$$
$$(c_0\alpha + c_1\alpha^2 + c_2\alpha^4 + c_3\alpha^8 + c_4\alpha^{16}),$$

$$c_0 = (0,1) + (1,3) + (2,3) + (2,4) + (4) = [0,1] + [1,3] + [2,3] + [2,4] + (0),$$
$$c_1 = (1,2) + (2,4) + (3,4) + (3,0) + (0) = [1,2] + [2,4] + [3,4] + [3,0] + (1),$$
$$c_2 = (2,3) + (3,0) + (4,0) + (4,1) + (1) = [2,3] + [3,0] + [4,0] + [4,1] + (2),$$
$$c_3 = (3,4) + (4,1) + (0,1) + (0,2) + (2) = [3,4] + [4,1] + [0,1] + [0,2] + (3),$$
$$c_4 = (4,0) + (0,2) + (1,2) + (1,3) + (3) = [4,0] + [0,2] + [1,2] + [1,3] + (4),$$

где

$$[i,j] = (x_i + x_j)(y_i + y_j), (i,j) = (x_i y_j + x_j y_i), (i) = x_i y_i.$$

Формулы состоят из пяти квадратичных форм, получающихся друг из друга циклическим сдвигом переменных. Общее число слагаемых в каждой из них равно $9 = 2 \cdot 5 - 1$, поэтому указанный базис является оптимальным.

Эти формулы дают две разные схемы умножения одинаковой сложности 55 и глубины 5 так как сложность вычисления $[i,j]$ или (i,j) равна 3, а глубина равна 2, и различных слагаемых в формулах ровно 15, так как каждое слагаемое вида (i,j) встречается по два раза, и всего их 10, а слагаемые вида (i) встречаются по одному разу и всего их 5. Но число умножений у схемы, построенной по формулам, содержащим $[i,j]$ будет меньше, а именно равно 15, так как вычисление $[i,j]$ в отличие от вычисления (i,j) требует одного умножения и двух сложений, а не наоборот.

Рассмотрим в поле $GF(2^{5n})$ оптимальный нормальный базис $\{\alpha, \alpha^2, \alpha^4, \alpha^8, \alpha^{16}\}$. Для сложности и глубины получаются оценки (при n не кратном 5)

$$L(M(5n)) \le 15L(M(n)) + 40n, D(M(5n)) \le D(M(n)) + 4.$$

Рассмотрим оптимальный нормальный базис второго типа в поле $GF(2^6)$

$$\{\alpha, \alpha^2, \alpha^4, \alpha^8, \alpha^{16}, \alpha^{32}\}.$$

Его минимальный многочлен f_6 вычисляется по рекуррентной формуле $f_0 = 0, f_1 = x + 1, f_{k+1} = xf_k(x) + f_{k-1}(x)$, откуда имеем

$$x^6 + x^5 + x^4 + x + 1.$$

Заметим, что (проверку для краткости опускаем)

$$\alpha\alpha^2 = \alpha^{16} + \alpha, \alpha\alpha^8 = \alpha^4 + \alpha^{32}, \alpha\alpha^4 = \alpha^8 + \alpha^{16}, \alpha\alpha^{16} = \alpha^4 + \alpha^2, \alpha\alpha^{32} = \alpha^8 + \alpha^{32},$$

откуда непосредственно выводится формула Месси-Омура умножения в этом базисе:

$$(a_0\alpha + a_1\alpha^2 + a_2\alpha^4 + a_3\alpha^8 + a_4\alpha^{16} + a_5\alpha^{32})(b_0\alpha + b_1\alpha^2 + b_2\alpha^4 + b_3\alpha^8 + b_4\alpha^{16} + b_5\alpha^{32}) =$$
$$(c_0\alpha + c_1\alpha^2 + c_2\alpha^4 + c_3\alpha^8 + c_4\alpha^{16} + c_5\alpha^{32}),$$
$$c_i = (i, i\oplus 1) + (i\oplus 1, i\oplus 4) + (i\oplus 2, i\oplus 3) + (i\oplus 2, i\oplus 4) + (i\oplus 3, i\oplus 5) + (i\oplus 5) =$$
$$[i, i\oplus 1] + [i\oplus 1, i\oplus 4] + [i\oplus 2, i\oplus 3] + [i\oplus 2, i\oplus 4] + [i\oplus 3, i\oplus 5] + (i),$$

где

$$[i,j] = (x_i + x_j)(y_i + y_j), (i,j) = (x_i y_j + x_j y_i), (i) = x_i y_i.$$

Формулы состоят из шести квадратичных форм, получающихся друг из друга циклическим сдвигом переменных. Общее число слагаемых в каждой из них равно $11 = 2 \cdot 6 - 1$, поэтому указанный базис является оптимальным.

Так как среди форм

$$(i, i\oplus 1), (i\oplus 1, i\oplus 4), (i\oplus 2, i\oplus 3), (i\oplus 2, i\oplus 4), (i\oplus 3, i\oplus 5),$$

составляющих формы c_i, на самом деле только $5 \cdot 6/2 = (n-1)n/2$ разных, то их суммарная сложность равна $3n(n-1)/2$, а с учетом сложности формул вида (i) она равна

$$3n(n-1)/2 + n = n(3n-1)/2 = 51.$$

Поэтому для вычисления системы форм c_i достаточно еще $n(C_B - 1)/2$ сложений, где C_B — число слагаемых в каждой из форм (равное по определению сложности рассматриваемого базиса B), ведь слагаемых вида (i, j) в каждой форме $(C_B - 1)/2$. Окончательная сложность указанной схемы

$$n(3n-1)/2 + n(C_B - 1)/2 = \frac{n}{2}(C_B + 3n - 2).$$

Эти формулы (одна с круглыми, а другая с прямыми скобками) дают на самом деле две разные схемы умножения одинаковой сложности $\frac{n}{2}(C_B+3n-2) = 81$ и глубины $1+\lceil \log_2(C_B+1)\rceil = 5$ так как сложность вычисления $[i, j]$ или (i, j) равна 3, а глубина равна 2.

Но число умножений у схемы, построенной по формулам, содержащим $[i, j]$ будет меньше, а именно равно $n(n+1)/2 = 21$, так как вычисление $[i, j]$ в отличие от вычисления (i, j) требует одного умножения и двух сложений, а не наоборот.

Для сложности и глубины получаются оценки (при n не кратном 6)

$$L(M(6n)) \leq 21L(M(n)) + 60n, D(M(6n)) \leq D(M(n)) + 4.$$

Для неприводимого многочлена

$$x^6 + x^5 + x^4 + x^2 + 1$$

получается еще один нормальный базис, уже не оптимальный. Этот тот самый базис, который получается перемножением оптимальных нормальных базисов порядков 2 и 3, т.е.

$$\{\alpha\beta, \alpha\beta^2, \alpha\beta^4, \alpha^2\beta, \alpha^2\beta^2, \alpha^2\beta^4\},$$

где

$$\alpha^2 + \alpha = 1, \beta^3 + \beta^2 = 1.$$

Точнее, он является перестановкой нормального базиса

$$\{, \gamma, \gamma^2, \gamma^4, \gamma^8, \gamma^{16}, \gamma^{32}\}, \gamma = \alpha\beta,$$

так как

$$\gamma^2 = \alpha^2\beta^2, \gamma^4 = \alpha^4\beta^4 = \alpha\beta^4, \gamma^8 = \alpha^8\beta^8 = \alpha^2\beta, \gamma^{16} = \alpha^4\beta^2 = \alpha\beta^2,$$
$$\gamma^{32} = \alpha^2\beta^4.$$

Для перехода к этому базису достаточно сделать перестановку координат

$$(a_0, a_4, a_2, a_3, a_1, a_5).$$

Минимальный многочлен элемента γ имеет корнями все элементы этого базиса, поэтому он равен

$$((x + \alpha\beta)(x + \alpha\beta^2)(x + \alpha\beta^4))((x + \alpha^2\beta)(x + \alpha^2\beta^2)(x + \alpha^2\beta^4)) =$$
$$((\alpha^2 x)^3 + (\alpha^2 x)^2 + 1)((\alpha x)^3 + (\alpha x)^2 + 1) =$$
$$x^6 + x^5 + x^4 + x + 1$$

так как в силу $\alpha^3 = 1$

$$((x + \alpha\beta)(x + \alpha\beta^2)(x + \alpha\beta^4)) = (\alpha^2 x + \beta)(\alpha^2 x + \beta^2)(\alpha^2 x + \beta^4) = (\alpha^2 x)^3 + (\alpha^2 x)^2 + 1)$$

ведь

$$(x+\beta)(x+\beta^2)(x+\beta^4) = x^3 + x^2 + 1.$$

Умножение в этом базисе по формуле Месси-Омура задается формулами

$$(a_0\gamma + a_1\gamma^2 + a_2\gamma^4 + a_3\gamma^8 + a_4\gamma^{16} + a_5\gamma^{32})(b_0\gamma + b_1\gamma^2 + b_2\gamma^4 + b_3\gamma^8 + b_4\gamma^{16} + b_5\gamma^{32}) =$$

$$(c_0\gamma + c_1\gamma^2 + c_2\gamma^4 + c_3\gamma^8 + c_4\gamma^{16} + c_5\gamma^{32}),$$

Формулу для билинейной формы c_0 удобнее получить другим способом, не вычисляя матрицу T коэффициентов разложения $\gamma\gamma^{2^k}$. Для этого воспользуемся разложениями по базису

$$\{\alpha\beta, \alpha\beta^2, \alpha\beta^4, \alpha^2\beta, \alpha^2\beta^2, \alpha^2\beta^4\},$$

именно,

$$(a_0\gamma + a_1\gamma^2 + a_2\gamma^4 + a_3\gamma^8 + a_4\gamma^{16} + a_5\gamma^{32}) =$$
$$= \alpha(a_{00}\beta + a_{01}\beta^2 + a_{02}\beta^4)\alpha^2(a_{10}\beta + a_{11}\beta^2 + a_{12}\beta^4),$$
$$(b_0\gamma + b_1\gamma^2 + b_2\gamma^4 + b_3\gamma^8 + b_4\gamma^{16} + b_5\gamma^{32}) =$$
$$= \alpha(b_{00}\beta + b_{01}\beta^2 + b_{02}\beta^4)\alpha^2(b_{10}\beta + b_{11}\beta^2 + b_{12}\beta^4),$$

и формулами умножения в базисах

$$\{\alpha, \alpha^2\}, \{\beta, \beta^2, \beta^4\},$$

полученных в предыдущих текстах. Тогда в результате умножения

$$(c_0\gamma + c_1\gamma^2 + c_2\gamma^4 + c_3\gamma^8 + c_4\gamma^{16} + c_5\gamma^{32}) =$$
$$= \alpha(c_{00}\beta + c_{01}\beta^2 + c_{02}\beta^4)\alpha^2(c_{10}\beta + c_{11}\beta^2 + c_{12}\beta^4),$$

вектор

$$C_0 = c_{00}\beta + c_{01}\beta^2 + c_{02}\beta^4$$

выражается в виде

$$C_0 = A_1B_1 + A_0B_1 + A_1B_0, A_i = a_{i0}\beta + a_{i1}\beta^2 + a_{i2}\beta^4, B_i = b_{i0}\beta + b_{i1}\beta^2 + b_{i2}\beta^4,$$

а так как умножение в базисе

$$\{\beta, \beta^2, \beta^4\}$$

определяется билинейной формой

$$(1,2) + (0,1) + (2),$$

где

$$(i,j) = (a_ib_j + a_jb_i), (i) = a_ib_i,$$

поэтому c_{00} выражается в виде билинейной формы

$$c_{00} = (a_{11}b_{12} + a_{12}b_{11}) + (a_{10}b_{11} + a_{11}b_{10}) + a_{12}b_{12} +$$
$$+(a_{01}b_{12} + a_{02}b_{11}) + (a_{00}b_{11} + a_{01}b_{10}) + a_{02}b_{12} +$$
$$(a_{11}b_{02} + a_{12}b_{01}) + (a_{10}b_{01} + a_{11}b_{00}) + a_{12}b_{02}.$$

В общем случае, если внешняя форма имеет вид

$$C'(a,b) = \sum_{i,j=0}^{m-1} C'_{i,j}a_ib_j,$$

а внутренняя форма имеет вид

$$C''(a,b)=\sum_{i,j=0}^{n-1}C''_{i,j}a_ib_j,$$

то их «суперпозиция» имеет вид

$$C(a,b)=\sum_{i,j=0}^{m-1}C'_{i,j}\sum_{k,l=0}^{n-1}C''_{k,l}a_{i,k}b_{j,l}=$$

$$=\sum_{i,j=0,k,l=0}^{m-1,n-1}C'_{i,j}C''_{k,l}a_{i,k}b_{j,l},$$

откуда видно, что матрица этой формы $C_{i,j,k,l}$ является кронекеровым произведением матриц C',C'', так как $C_{i,j,k,l}=C'_{i,j}C''_{k,l}$, а ее сложность(равная сумме ее элемеентов и, что равносильно, числу слагаемых в форме) равна

$$\sum_{i,j=0,k,l=0}^{m-1,n-1}C'_{i,j}C''_{k,l}=\left(\sum_{i,j=0}^{m-1}C'_{i,j}\right)\left(\sum_{k,l=0}^{n-1}C''_{k,l}\right)$$

произведению сложностей своих сомножителей. В частности, в рассматриваемом примере сложность нормального базиса, порожденного γ, равна $3\cdot5=15$, что видно и непосредственно. Если теперь в форме

$$c_{00}=(a_{11}b_{12}+a_{12}b_{11})+(a_{10}b_{11}+a_{11}b_{10})+a_{12}b_{12}+$$

$$+(a_{01}b_{12}+a_{02}b_{11})+(a_{00}b_{11}+a_{01}b_{10})+a_{02}b_{12}+$$

$$(a_{11}b_{02}+a_{12}b_{01})+(a_{10}b_{01}+a_{11}b_{00})+a_{12}b_{02}.$$

перейти к координатам базиса, порожденного γ, по формулам

$$a_{0,0}=a_0,a_{0,1}=a_4,a_{0,2}=a_2,a_{1,0}=a_3,a_{1,1}=a_1,a_{1,2}=a_5,$$

и применить использованные раньше сокращенные обозначения

$$(i,j)=(a_ib_j+a_jb_i),(i)=a_ib_i,$$

то получим (после соответствующих перестановок) форму

$$c_0=(1,5)+(1,3)+(5)+(4,5)+(0,1)+(3,4)+(2,5)+(1,2)=$$

$$=(1,5)+(1,3)+(4,5)+(0,1)+(3,4)+(2,5)+(1,2)+(5).$$

Используя указанный выше метод построения схемы умноженния для нормального базиса, получаем для этого базиса схему сложности

$$\frac{n}{2}(C_B+3n-2)=3(15+18-2)=93$$

и глубины

$$1+\lceil\log_2(C_B+1)\rceil=5.$$

Метод этот совершенно общий и фактически совпадает с предложенным в [143]. Там приведено хотя и короткое, но непросто воспринимаемое доказательство, и ничего не говорится о глубине.

На самом деле для его обоснования достаточно заметить, что для каждого k в ней найдется член вида $(i, i \oplus k)$, причем при $k = 0$ — ровно один. Для этого в свою очередь достаточно воспользоваться связью между симметрической матрицей коэффициентов $(a_{i,j})$ этой формы и матрицей $t_{i,j}$ коэффициентов разложения

$$\alpha\alpha^{2^k} = \sum_{i=0}^{n-1} t_{k,i}\alpha^{2^i},$$

которая имеет вид $a_{i,j} = t_{i-j \bmod n, -j \bmod n}$, и заметить, что среди коэффициентов $t_{k,j}$ при $k = 0$ единица встречается только при $j = 1$, значит $a_{n-1,n-1} = 1$, а остальные диагональные элементы нули. Поэтому в форме c_0 слагаемое (i) встречается только при $i = n - 1$.

Упражнение 2.5.21 Проверьте, что в формуле c_0 каждый индекс, кроме нуля, встречается четное число раз, а нуль — — нечетное число раз. Указание. Среди коэффициентов $t_{k,j}$ при любом фиксированном $j \neq 0$

единиц четное количество, а при $j = 0$ — нечетное количество, как было доказано в теореме о нижней оценке сложности нормальных базисов, а индекс i встречается в формуле c_0 столько раз, сколько единиц стоит в i-м столбце матрицы A, а их столько же, сколько единиц в $n - i \bmod n$-м столбце матрицы T, т.е. нечетное количество при $i = 0$ и четное ненулевое количество в остальных случаях.

Из предыдущего утверждения следует, что если вместо форм (i, j) подставить формы $[i, j]$, как в рассмотренных выше примерах, то и в случае произвольного нормального базиса схема преобразуется в схему той же тотальной сложности, но меньшей мультипликативной сложности, равной $n(n + 1)/2$ независимо от этого базиса.

Упражнение 2.5.22 Докажите это.

Упражнение 2.5.23 Проверьте, что если базис является ONB, то в его матрице ровно $2n-1$ единица, поэтому в нулевом столбце матрицы ровно одна единица, а в остальных столбцах ровно две, значит в билинейной форме c_0 ненулевой индекс встречается два раза, а нулевой индекс — один раз.

Заметим, что для ONB первого типа найдется единственное $k = n/2$ такое, что

$$\alpha\alpha^{2^k} = 1 = \sum_{i=0}^{n-1} t_{k,i}\alpha^{2^i},$$

так как $2^{n/2} = -1 \bmod p$, поэтому в $n/2$-й строке матрицы T стоят одни единицы, а в остальных строках, очевидно только по одной единице. Поэтому в случае ONB первого типа в линейной форме c_0 слагаемые вида $(i, i \oplus n/2)$ встречаются n раз, а слагаемые вида $(i, i \oplus k)$ при любом другом фиксированном k — ровно один раз.

Упражнение 2.5.24 Проверьте, что для ONB второго и третьего типа в каждой строке, кроме первой, стоят по две единицы (это видно из доказательства соответствующих теорем) и в линейной форме c_0 слагаемые вида $(i, i \oplus k)$ при любом фиксированном $k \neq 0$ встречаются ровно два раза, а слагаемое вида (i) — ровно один раз.

Поэтому в случае ONB первого типа для реализации всей системы линейных форм, получающихся из c_0 циклическим сдвигом переменных, достаточно реализовать по одному разу все формы (i, j) при $i - j \neq n/2 \bmod n$, а также по одному разу все формы (i, j) при

$i-j=n/2 \bmod n$, потом их сложить и для реализации каждой c_i нужно к этой сумме прибавлять по $n-1$ форм вида (i,j) при $i-j \neq n/2 \bmod n$. Для реализации всех форм вида (i,j) нужно $n(n+1)/2$ конъюнкторов и $n(n-1)/2$ элементов сложения по модулю два, и дополнительно нужно $n(n-1)+n-1=n^2-1$ элементов сложения по модулю два. Тотальная сложность схемы равна $2n^2-1$.

Оценка сложности для произвольного нормального базиса $\frac{n}{2}(C_B+3n-2)$ иногда может быть улучшена. Например, заметим, что есть более простая схема умножения в рассмотренном выше базисе в поле $GF(2^6)$, имеющая ту же глубину. Она строится по формуле

$$c_0=(1,5)+(1,3)+(4,5)+(0,1)+(3,4)+(2,5)+(1,2)+(5)=$$

$$=((1,5)+(4,5)+(1,3)+(3,4))+((0,1)+(1,2))+(2,5)+(5)=$$

$$((a_1+a_4)(b_5+b_3)+(b_1+b_4)(a_5+a_3))+((a_0+a_2)b_1+(b_0+b_2)a_1)+$$

$$+(a_2+a_5)(b_2+b_5)+a_2b_2,$$

которую легко усмотреть из покрытия (по модулю два) матрицы этой формы прямоугольниками

$$\{1,4\}\times\{3,5\},\{3,5\}\times\{1,4\},\{0,2\}\times\{1\},\{1\}\times\{0,2\},\{2,5\}\times\{2,5\},\{2\}\times\{2\}.$$

Очевидно, что глубина этой формулы и всех остальных пяти формул, полученных из нее циклической перестановкой переменных, равна 5. Если заранее вычислить все 12 сумм вида

$$a_i+a_{i\oplus 2}, b_i+b_{i\oplus 2},$$

и все 6 сумм вида

$$a_i+a_{i\oplus 3}, b_i+b_{i\oplus 3}$$

со сложностью 18, то каждая из формул c_i будет вычисляться с помощью 6 умножений и 5 сложений, откуда мультипликативная сложность указанной схемы равна 36, а аддитивная — $5\cdot 6+18=48$, а полная сложность равна 84. Но если заметить, что произведений вида

$$(a_i+a_{i\oplus 3})(b_i+b_{i\oplus 3})$$

на самом деле не 6 а 3, то мультипликативная сложность уменьшается до 33, а полная —- до 81 и оказывается равной сложности умножения в оптимальном базисе, а так как XOR более медленный и более сложный элемент, то эта схема даже лучше.

В заключение в качестве примера покажем, как можно построить для рассмотренного выше GNB 4-го типа более простую схему, чем при непосредственном применении формулы Месси-Омура. Напомним, что они в данном случае имеют вид

$$z_i=(i,i\oplus 1)+(i\oplus 1,i\oplus 3)+(i\oplus 1,i\oplus 4)+(i\oplus 1,i\oplus 5)+(i\oplus 2,i\oplus 4)+(i\oplus 2,i\oplus 8)+$$

$$+(i\oplus 2,i\oplus 11)+(i\oplus 2,i\oplus 12)+(i\oplus 3,i\oplus 7)+(i\oplus 3,i\oplus 8)+(i\oplus 3,i\oplus 12)+$$

$$+(i\oplus 4,i\oplus 12)+(i\oplus 4,i\oplus 10)+(i\oplus 5,i\oplus 7)+(i\oplus 5,i\oplus 8)+(i\oplus 5,i\oplus 10)+$$

$$+(i\oplus 6,i\oplus 9)+(i\oplus 6,i\oplus 11)+(i\oplus 8,i\oplus 9)+(i\oplus 9,i\oplus 10)+$$

$$+(i\oplus 9,i\oplus 11)+(i\oplus 10,i\oplus 11)+(i\oplus 12).$$

Применяя указанный выше метод, можно построить для произвольного GNB четвертого типа схему, состоящую из n умножений, $n(n-1)/2$ форм вида (i,j) и $n(C-1)/2=n(2n-4)$ сложений. Если вместо форм вида (i,j) использовать формы вида $[i,j]$, то аддитивная сложность схемы будет равна $n(C-1)/2+n(n-1)=n(C+2n-3)/2=n(3n-5)$, а мультипликативная

сложность равна $n(n-1)/2+n = n(n+1)/2$. Тотальная сложность равна $3.5n^2 - 4.5n = 533$. Воспользуемся формулами

$$z_i = ((i\oplus 1, i\oplus 4) + (i\oplus 1, i\oplus 5) + (i\oplus 4, i\oplus 10) + (i\oplus 5, i\oplus 10))+$$
$$+((i\oplus 3, i\oplus 7) + (i\oplus 5, i\oplus 7) + (i\oplus 5, i\oplus 8) + (i\oplus 3, i\oplus 8))+$$
$$+((i\oplus 6, i\oplus 9) + (i\oplus 6, i\oplus 11) + (i\oplus 9, i\oplus 10) + (i\oplus 10, i\oplus 11))$$
$$((i\oplus 2, i\oplus 8) + (i\oplus 2, i\oplus 11) + (i\oplus 8, i\oplus 9) + (i\oplus 9, i\oplus 11)))$$
$$+(i, i\oplus 1) + (i\oplus 1, i\oplus 3) + (i\oplus 2, i\oplus 4) + (i\oplus 2, i\oplus 12)+$$
$$+(i\oplus 3, i\oplus 12) + (i\oplus 4, i\oplus 12) + +(i\oplus 12) =$$
$$(a_{i\oplus 4} + a_{i\oplus 5})(b_{i\oplus 1} + b_{i\oplus 10}) + (b_{i\oplus 4} + b_{i\oplus 5})(a_{i\oplus 1} + a_{i\oplus 10})+$$
$$(a_{i\oplus 3} + a_{i\oplus 5})(b_{i\oplus 7} + b_{i\oplus 8}) + (b_{i\oplus 3} + b_{i\oplus 5})(a_{i\oplus 7} + a_{i\oplus 8})+$$
$$(a_{i\oplus 6} + a_{i\oplus 10})(b_{i\oplus 9} + b_{i\oplus 11}) + (b_{i\oplus 6} + b_{i\oplus 10})(a_{i\oplus 9} + a_{i\oplus 11})+$$
$$(a_{i\oplus 2} + a_{i\oplus 9})(b_{i\oplus 8} + b_{i\oplus 11}) + (b_{i\oplus 2} + b_{i\oplus 9})(a_{i\oplus 8} + a_{i\oplus 11})+$$
$$+(a_{i\oplus 1})(b_i + b_{i\oplus 3}) + (b_{i\oplus 1})(a_i + a_{i\oplus 3}) + (a_{i\oplus 2})(b_{i\oplus 4} + b_{i\oplus 12}) + (b_{i\oplus 2})(a_{i\oplus 4} + a_{i\oplus 12})+$$
$$(a_{i\oplus 12})(b_{i\oplus 4} + b_{i\oplus 3} + b_{i\oplus 12}) + (b_{i\oplus 12})(a_{i\oplus 4} + a_{i\oplus 3} + a_{i\oplus 12}).$$

Мультипликативная сложность построенной по ней схемы равна $14 \cdot 13$, так как в каждой формуле 14 умножений, а аддитивная сложность $(14+13)\cdot 13$, так как в каждой формуле 13 сложений плюс еще заранее вычисленные суммы

$$a_i + a_{i\oplus 1}, a_i + a_{i\oplus 9}, a_i + a_{i\oplus 2}, a_i + a_{i\oplus 7}, a_i + a_{i\oplus 3}, a_i + a_{i\oplus 8}, a_i + a_{i\oplus 8} + a_{i\oplus 12},$$

и такие же суммы для переменных b_i. Тотальная сложность равна 533. Но если переписать

$$(a_{i\oplus 2})(b_{i\oplus 4} + b_{i\oplus 12}) + (b_{i\oplus 2})(a_{i\oplus 4} + a_{i\oplus 12})+$$
$$(a_{i\oplus 12})(b_{i\oplus 4} + b_{i\oplus 3} + b_{i\oplus 12}) + (b_{i\oplus 12})(a_{i\oplus 4} + a_{i\oplus 3} + a_{i\oplus 12})$$

в виде

$$(a_{i\oplus 2} + a_{i\oplus 12})(b_{i\oplus 4} + b_{i\oplus 12}) + (b_{i\oplus 2} + b_{i\oplus 12})(a_{i\oplus 4} + a_{i\oplus 12})+$$
$$a_{i\oplus 12}b_{i\oplus 3} + b_{i\oplus 12}a_{i\oplus 3},$$

то можно обойтись без сумм $a_i + a_{i\oplus 8} + a_{i\oplus 12}$ и аналогичных сумм для переменных b_i, поэтому тотальная сложность уменьшается до 507. Глубина схемы равна 6. Для сравнения заметим, что наилучшая схема умножения в стандартном базисе имеет мультипликативную сложность 169, аддитивную сложность 202, но глубину 7.

Глава 3

Операции в $GF(2^n)$ в полиномиальном базисе

3.1 Классический алгоритм умножения в $GF(2)[X]$

3.1.1 Операция умножения в кольце $GF(2)[X]$

Рассматриваемые в данном разделе операции *сложения* и *умножения*$p_1(X) = \sum_{i=0}^{n-1} a_i X^i$ и $p_2(X) = \sum_{i=0}^{n-1} b_i X^i$ многочленов над $GF(2)$ (то есть операции сложения и умножения в кольце $GF(2)[X]$) определяются следующим образом.

$$p_1(X) + p_2(X) = \sum_{i+0}^{n-1} (a_i + b_i) X^i. \tag{3.1}$$

$$p_1(X) \times p_2(X) = \sum_{k=0}^{2n-2} \sum_{i+j=k} a_i b_j X^k, \tag{3.2}$$

Рассматриваются классический метод умножения и его модификации.

При реализации соответствующих алгоритмов применяются программные эвристики, учитывающие, что возможности одновременного выполнения однотипных поразрядных операций ограничиваются пределами машинного слова, но в то же время достаточно большой доступный объем оперативной памяти позволяет расширить параллелизм за счет табличной реализации групповых операций.

Если многочлен $f(X) = \sum_{i=0}^{n-1} a_i X^i$ представлен последовательностью коэффициентов

$$a_0, a_1, \ldots, a_n,$$

то его можно представить последовательностью

$$A^{(0)}, A^{(1)}, \ldots, A^{(k-1)}$$

из $k = \lceil \frac{n}{s} \rceil$ s-разрядных машинных слов

$$A^{(0)} = a_0, a_1, \ldots, a_{s-1},$$

$$A^{(1)} = a_s, a_{s+1}, \ldots, a_{2s-1},$$

$$\ldots$$

$$A^{(k-1)} = a_{(k-1)s}, a_{(k-1)s+1}, \ldots, a_{ns-1}.$$

В случае, когда n не делится на s, старшие $ks - n$ бит слова $A^{(k-1)}$ дополняются нулями.

В данном и следующих разделах векторы коэффициентов, размещаемые в одном машинном слове, называются «длинными целыми числами» (им соответствует, например, тип данных long). Машинные слова могут разбиваться на более мелкие составные части, например, 32-разрядное «длинное целое число» A представляется четырьмя байтами

$$A[0], A[1], A[2], A[3].$$

Использование данных одновременно и как машинных слов и как байт обеспечивается, например, типом данных "union,"а представление многочленов как последовательностей машинных данных осуществляется в виде классов.

Эти особенности предопределяют структуру декомпозиции операции как функции над «длинными» операндами, сводящей ее к операциям над машинными словами, на которые разбиваются эти операнды. В свою очередь, операции над машинными словами могут разлагаться в последовательность операций над их частями, допускающих, как правило, табличную реализацию.

Простейшим случаем такой декомпозиции является разложение операции сложения многочленов степени n на $k =]\frac{n}{s}[$ операций поразрядного сложения содержимого машинных слов длины s бит.

Декомпозиция умножения требует более тонкого анализа. Умножение над $GF(2)$ только в простейших случаях сводится к распараллеливанию стандартного метода умножения <столбиком>. Один из возможных вариантов такого рода оказывается эффективным при достаточно малом количестве ненулевых коэффициентов одного из сомножителей.

Целью данного раздела является, таким образом, изучение эффективных имплементаций операции умножения многочленов над $GF(2)$, классическим методом с выбором модификации метода в зависимости от сложности операндов.

3.1.2 Элементарные многочлены. Таблица умножения

Степень многочлена $P(X) \in GF(2)[X]$ обозначается $\deg P(X)$. Многочлены степени не выше $s - 1$ будем называть *элементарными*. Число s является параметром и используется в дальнейшем для оптимизации времени работы алгоритмов. Возможные значения для s: 1,2,4,8,16,32.

Тогда многочлен $P(X)$ задается суммой

$$\sum_{i=0}^{t} Q_i(X) X^{is},$$

где $Q_i(X)$ - элементарные многочлены. Наибольшее i, что $Q_i(X) \neq 0$, обозначим $\deg_s P(X)$.

Для разных алгоритмов предполагается использовать вообще говоря различные значения параметра s. Так для вычисления суммы $P_1(x) + P_2(X)$ многочленов $P_1(X)$ и $P_2(x)$ разумно положить $s = 32$. Тогда элементарный многочлен определяется в памяти ПК машинным словом, а операция сложения многочленов сводится к логическому суммированию соответствующих элементарных многочленов.

Произведение двух многочленов P_1 P_2 вычисляется путем нахождения элементарных многочленов $Z_i(X)$, являющихся коэффициентами при X^{ik}, $1 = 0, 1 \ldots, \deg_s P_1 X + \deg_s P_2(x)$.

Перемножая два элементарных многочлена, получаем многочлен степени не выше $2s-2$, который задается суммой

$$Q_1(X) + Q_2(X)X^s,$$

где $Q_i(X)$ - элементарный многочлен, $i = 1, 2$.

Таким образом, произведение элементарных многочленов $P_1(X)P_2(X)$ представляется парой элементарных многочленов и может быть вычислено, например, по методу Берлекемпа-Петерсона: умножать элементарные многочлены $P_1(X)$ и $P_2(X)$,

$$P_2(X) = \sum_{i=0}^{s-1} a_i X^i,$$

предлагается путем вычисления произведения $P_1(X)X^i$ для каждого $i, \quad i = 0, 1, \ldots, s-1$, для которого $a_i \neq 0$, и их суммирования.

Элементарные многочлены $Q_1(X)$ и $Q_2(X)$ такие, что

$$P_1(X)x^i = Q_1(X) + Q_2(x)X^{s-1}$$

определяются формулами

$$Q_1(X) = P_1(x)X^i \bmod X^k,$$
$$Q_2(x) = Q_1(X) + P_1(X)).$$

Пример. Пусть $P_1(X) = 1 + X + X^2,\ k = 3,\ i = 2.$

$$P_1(X)X^i = (1 + X + X^2)X^2 = X^2 + X^3 + X^4 = Q_1(x) + Q_2(X)X^2,$$

где

$$Q_1(X) = P_1(X)X^2 \bmod X^3 = X^2,$$
$$Q_2(X) = Q_1(X) + P_1(X) = X + X^2.$$

Рассмотрим алгоритм умножения элементарного многочлена $P_(X)$ на элементарный многочлен $P_2(X)$.

Даны векторы U и V длины k коэффициентов элементарных многочленов $P_1(X)$ и $P_2(X)$ в порядке возрастания степеней соответствующих термов многочленов, для формирования результата используется вектор Z длины $2k$ он образуется двумя векторами Z_1 и Z_2 длины k (Z_1 соответствует младшим, а Z_2 – старшим разрядам вектора Z)
Требуется вычислить произведение $Z = U \cdot V$.

1. $Z = 0$,
2. Выполнить k раз
Если $]V = 1$, то $Z_2 = Z_2 + U$,
$Z = Z[\leftarrow], V = V[\leftarrow]$.

$]V$ означает младший разряд вектора V, $[\leftarrow]$ – операция сдвига в сторону младших разрядов. Элементы векторов Z_1 и Z_2 определяют коэффициенты элементарных многочленов $Q_1(X)$ и $Q_2(X)$.

При небольших значения s (например, $s \leq 8$) операция умножения элементарных многочленов может выполняться с помощью заранее составленной таблицы T_M умножения. Её строки и столбцы соответствуют различным элементарным многочленам, а элементы – произведениям элементарных многочленов (наборы бинарных коэффициентов записываются в порядке возрастания степеней соответствующих термов), что позволяет отказаться от поразрядных операций и работать с байтами как с минимальными неделимыми блоками данных.

Пример. Таблица умножения при $s = 2$ имеет вид:

	00	10	01	11
00	0000	0000	0000	0000
10	0000	1000	0100	1100
01	0000	0100	0010	0110
11	0000	1100	0110	1010

Примечание. При $s = 8$ таблица умножения занимает $2^8 \times 2^8 \times 2^1 = 2^{17}$ байт, или $2^7 = 128$ Кбайт.

3.1.3 Умножение многочленов с использованием таблицы умножения

Пусть

$$P_1(X) = \sum_{i=0}^{t} U_i(X)X^{ik},\ P_2(X) = \sum_{j=0}^{p} V_j(X)X^{jk}.$$

$$P(X)\sum_{r=0}^{t+p} Z_r(X)X^{rk} = P_1(X) \times P_2(X).$$

Рассмотрим алгоритм вычисления "коэффициентов"$Z_r(X)$ произведения $P(X)$ многочленов $P_1(X)$ и $P_2(X)$.

Пусть $\deg_s P_1(X) = t$ и $\deg_s P_2 = p$. "Коэффициенты"$U_i(X)$ и $V_j(X)$ будем записывать в виде бинарных векторов длины s, образующих элементы массивов $U[t+1]$ и $V[p+1]$, "коэффициенты"$Z_r(X)$ произведения $P(X)$ будем вычислять как элементы массива $Z[t+p+1]$. Будем обозначать $\{U[i] \times V[j]\}_1$ s младших разрядов вектора, представляющего произведение элементарных многочленов U_i и V_j, а $\{U[i] \times V[j]\}_2$ – s старших разрядов этого произведения (эти векторы можно брать непосредственно из заранее составленной таблицы умножения). Алгоритм умножения можно описать теперь следующим образом

1. $Z = 0$;
2. Для $i = 0, s$
   Для $j = 0, p$
   $Z_{i+j} = Z_{i+j} + \{U[i] \times V[j]\}_1$;
   $Z_{i+j+1} = Z_{i+j+1} + \{U[i] \times V[j]\}_2$

Пример 3.1.1 Пусть $s = 8$ и нужно перемножить многочлены

$$P_1(X) = (1 + X^2)X^8 + (X^3 + 1) \text{ и } P_2(X) = (1 + X + X^7)X^{16} + X^6.$$

Тогда

$$P_1(X)P_2(X) =$$

$$= (1 + X^2)(1 + X + X^7)X^{24} + (X^3 + 1)(1 + X + X^7)X^{16} + (1 + X^2)X^6X^8 + (X^3 + 1)X^6.$$

Обращаясь к заранее вычисленной таблице произведений элементарных многочленов, получаем

$$(1 + X^2)(1 + X + X^7) = XX^8 + (1 + X + X^2 + X^3 + X^7),$$

$$(X^3+1)(1+X+X^7) = X^2X^8 + (1+X+X^3+X^4+X^7),$$
$$(1+X^2)X^6 = X^8+X^6,\ (X^3+1)X^6 = XX^8+X^6,$$

Откуда

$$P_1(X)P_2(X) = XX^{32} + (1+X+X^2+X^3+X^7)X^{24} + X^2X^{24} +$$
$$+(1+X+X^3+X^4+X^7)X^{16} + X^{16} + X^6X^8 + XX^8 + X^6 =$$
$$= XX^{32} + (1+X+X^3+X^7)X^{24} + (X+X^3+X^4+X^7)X^{16} + (X+X^6)X^8 + X^6.$$

Преимущества такого алгоритма состоят в следующем. Оценим объем памяти, используемой для хранения матрицы T при $k=8$. Таблица T содержит 2^{16} элементов, каждый из которых занимает 2 байта. Поэтому для хранения матрицы T нужно 2^{17} байт или 128 Kb, что составляет вполне разумный объем памяти для современных вычислительных средств.

С другой стороны, реализация умножения двух многочленов степени не выше $2^8-1=255$ обычным "школьным"алгоритмом приведет к не более чем $(\frac{2^8}{k})^2 = 2^{10}$ обращениям к таблице T и не более чем 2^{10} байтовым логическим сложениям, в то время, как число операций без использования матрицы T оценивается как $(2^8)^2 = 2^{16}$.

3.1.4 Модификация классического алгоритма и гибридный алгоритм умножения

Рассматриваемый в настоящем параграфе алгоритм играет вспомогательную роль в построенной нами схеме вычислений. Он используется лишь когда один из перемножаемых многочленов содержит небольшое число (не более трети) единичных коэффициентов.

Сдвигом вектора $(c_{n'-1}, c_{n'-2}, \ldots, c_1, c_0)$ будем называть вектор $(c_{n'-1}, c_{n'-2}, \ldots, c_1, c_0, 0)$.

Рассмотрим два многочлена $f(X)$ и $g(X)$ в стандартном (полиномиальном) базисе:

$$f(X) = \sum_{i=0}^{n-1} a_i X^i, \qquad g(X) = \sum_{i=0}^{n-1} b_i X^i.$$

Пусть векторы

$$A = (a_{n-1}, a_{n-2}, \ldots, a_1, a_0) \text{ и } B = (b_{n-1}, b_{n-2}, \ldots, b_1, b_0)$$

их коэффициентов заданы последовательностями машинных слов

$$C^{(k-1)}, C^{(k-2)}, \ldots, C^{(1)}, C^{(0)} \text{ и } D^{(k-1)}, D^{(k-2)}, \ldots, D^{(1)}, D^{(0)}$$

длины s соответственно.

Определим последовательность векторов $V_i, \quad i = 0,1,\ldots,s-1$ так, что $V_0 = A$, и для любого $i, \quad i = 1,2,\ldots,n-1$, вектор V_i получается сдвигом вектора V_{i-1}. Из вектора V_i добавлением слева $(k+1)s-n-i$ нулей получим вектор V_i', с $(k+1)s$ компонентами, $i = 0,1,\ldots,s-1$. V_i' может быть задан последовательностью $E_i = E_i^{(k)}, E_i^{(k-1)}, \ldots, E_i^{(1)}, E_i^{(0)}$ из $k+1$ машинного слова.

Пусть нужно перемножить многочлены $f(X)$ и $g(X)$. Сначала построим последовательности E_i машинных слов $i = 0,1,\ldots,s-1$. Далее определим индексы $t_0, t_1, \ldots, t_m$ компонент вектора B, равных единице, $0 \le t_0 < t_1 < \ldots < t_m \le n-1$. Для каждого $j, \quad j = 0,1,\ldots,m$ определим частное q_j и остаток r_j от деления числа t_j на s.

Последовательность из $2n$ нулевых машинных слов обозначим P. Таким образом, $P = L_0, L_1, \ldots, L_{2n-1}$ и $L_i = 0 \quad i = 0, 1, \ldots, 2n-1$. Далее для каждого $j, \quad j = 0, 1, \ldots, m$ положим

$$L_{q_j+i} := L_{q_j+i} + E^{(i)}_{r_j}, \quad i = 0, 1, \ldots, k.$$

Полученная в результате описанных выше действий последовательность P машинных слов задает вектор коэффициентов произведения многочленов $f(X)$ и $g(X)$.

Теперь поясним, как ускорить поиск индексов единичных элементов вектора B. Предварительно (до начала выполнения алгоритма умножения многочленов) составим одномерную таблицу T, с помощью которой по заданному байту b определяются места единичных бит в этом байте. Перемножая многочлены $f(X)$ и $g(X)$, для каждого j-го байта машинного слова $D^{(i)}$ из последовательности $D^{(k-1)}, D^{(k-2)}, \ldots, D^{(1)}, D^{(0)}$ по таблице T определим список номеров мест единичных бит. По этим номерам и числам i, j определяется очередная порция индексов единичных элементов вектора B. Так, если ω – один из номеров в списке единичных бит j-го байта машинного слова $D^{(i)}$, то ему соответствует индекс $si + 8j + \omega$.

Пример. Пусть требуется перемножить многочлены $f(X) = X^{10} + X^8 + X^7 + X^5 + X^3 + X^2 + 1$ и $g(X) = X^8 + X^7 + X^5 + X$. В демонстрационных целях предположим, что $s = 4$. Тогда многочлены $f(X)$ и $g(X)$ задаются векторами

$$A = (1, 0, 1, 1, 0, 1, 0, 1, 1, 0, 1), \; B = (0, 0, 1, 1, 0, 1, 0, 0, 0, 1, 0),$$

соответственно, или последовательностями машинных слов

$$C = 5, 10, 13, \; D = 1, 10, 2.$$

Сначала определим последовательность векторов V_0, V_1, V_2, V_3, получаемых сдвигами из вектора A, и далее дополняем слева эти векторы нулями так, чтобы получаемые вектора состояли из 16 компонент. (16 – наименьшее число, делящееся на 4 и не меньшее количества компонент в векторе V_3) :

$$V'_0 = (0, 0, 0, 0, 0, 1, 0, 1, 1, 0, 1, 0, 1, 1, 0, 1),$$

$$V'_1 = (0, 0, 0, 0, 1, 0, 1, 1, 0, 1, 0, 1, 1, 0, 1, 0),$$

$$V'_2 = (0, 0, 0, 1, 0, 1, 1, 0, 1, 0, 1, 1, 0, 1, 0, 0),$$

$$V'_3 = (0, 0, 1, 0, 1, 1, 0, 1, 0, 1, 1, 0, 1, 0, 0, 0),$$

Полученные векторы задают соответственно следующие последовательности машинных слов:

$$E_0 = 0, 5, 10, 13, \quad E_1 = 0, 11, 5, 10,$$
$$E_2 = 1, 6, 11, 4, \quad E_3 = 2, 13, 6, 8.$$

Эти последовательности задают таблицу сдвигов.

В векторе B единицы расположены на местах с номерами 1, 5, 7, 8. Деля эти числа на 4 с остатком, получаем:

$$q_0 = 0, \quad r_0 = 1, \quad q_1 = 1, \quad r_1 = 1,$$
$$q_2 = 1, \quad r_2 = 3, \quad q_3 = 2, \quad r_3 = 0.$$

В заключение вычисляем результат умножения с использованием таблицы сдвигов:

$$\begin{array}{cccccccc} & 0 & 0 & 0 & 11 & 5 & 10 \\ \bigoplus & 0 & 0 & 11 & 5 & 10 & 0 \\ & 0 & 2 & 13 & 6 & 8 & 0 \\ & 0 & 5 & 10 & 13 & 0 & 0 \\ \hline & 0 & 7 & 12 & 5 & 7 & 10 \end{array}$$

Последовательность машинных слов, записанная под чертой, соответствует вектору коэффициентов

(111.1100.0101.0111.1010)

и задает многочлен

$$X^{18} + X^{17} + X^{16} + X^{15} + X^{14} + X^{10} + X^8 + X^6 + X^5 + X^4 + X^3 + X,$$

который и является произведением рассматриваемых в настоящем примере многочленов $f(X)$ и $g(X)$.

При перемножении некоторых заданных многочленов $f(X)$ и $g(X)$ сначала выбирается один из рассмотренных алгоритмов. Если среди перемножаемых многочленов есть содержащий небольшое число единичных коэффициентов (менее некоторого порога P), то быстрее будет работать модифицированный классический алгоритм. В противном случае следует использовать другие методы, в частности, оригинальную схему, рассматриваемую в следующем разделе. Конкретное значение порога P определяется экспериментально при многократной апробации обоих рассматриваемых алгоритмов.

Далее приводится способ быстрого подсчета числа единичных коэффициентов многочлена $f(X)$.

Заранее строится одномерный массив M с $2^{16} = 64768$ элементами: $M[0], M[1], \ldots, M[64767]$ такой, что $M[i]$ равен числу единиц в двоичном разложении числа i. Например,

$$M[1234] = M[01001011001_2] = 5.$$

Пусть многочлен $f(X)$ задан последовательностью $A_0, A_1, \ldots,$
A_k двухбайтовых чисел. Тогда сумма $M[A_0] + M[A_1] + \ldots + M[A_k]$ равна числу единичных коэффициентов многочлена $f(X)$.

3.2 Оптимизация умножения многочленов

3.2.1 Введение

Данный раздел посвящен изучению методов выполнения умножения многочленов над полем $GF(2)$, основанных на декомпозиции по методу Карацубы (Карацуба А.А., Оффман Ю.П. Умножение многозначных чисел на автоматах. ДАН СССР, 1962, т. 145). Показана эффективность декомпозиционного подхода к реализации этой операции при степенях полиномов в диапазоне от 100 до 1000, характерном для криптосистем с открытым ключом.

Эффективность выполнения операций умножения достигается за счет применения оригинальной декомпозиционной схемы умножения по методу Карацубы и учета специфики операндов при модификации классического алгоритма.

3.2.2 Умножение многочленов по методу Карацубы

В 1962 году Карацуба А.А. (ныне известный специалист по теории чисел, профессор МГУ) предложил метод умножения многозначных чисел, более эффективный, чем известный классический метод умножения <столбиком>.

Умножение двух $2n$-значных чисел легко сводится к четырем умножениям двух n-значных чисел. По методу Карацубы достаточно трех таких умножений.

Запишем $2n$-значное число в виде

$$A_n + 10^n \cdot B_n.$$

Легко проверяется тождество

$$(A_n + 10^n \cdot B_n) \cdot (C_n + 10^n \cdot D_n) =$$

$$A_n \cdot C_n(10^n + 1) + (D_n - C_n) \cdot (A_n - B_n) \cdot 10^n + B_n \cdot D_n \cdot (10^{2n} + 10^n).$$

Подобным образом можно поступить и при умножении двух многочленов степени $2n$ в кольце полиномов над полем Галуа $GF(2)$.

Пусть многочлены $p(x)$ и q(x) степени 2n-1 представлены в виде

$$p(x) = p_1(x) + x^n \times p_2(x),$$

$$q(x) = q_1(x) + x^n \times q_2(x),$$

где p_1, p_2, q_1, q_2 – многочлены степени $n - 1$.

Тогда произведение

$$p(x) \times q(x)$$

можно представить в форме

$$p(x) \times q(x) =$$

$$= p_1(x) \times q_1(x) +$$

$$+(p_1(x) + p_2(x)) \times (q_1(x) + q_2(x))x^n +$$

$$+p_2(x) \times q_2(x)x^{2n} +$$

$$+(p_1(x) \times q_1(x))x^n +$$

$$(p_2(x) \times q_2(x))x^n.$$

Как видно, использовано только три умножения вместо четырех умножений многочленов степени n. Систематическое применение этого приема соответствует рекурсивной схеме, <развертkой> которой при различных базисах рекурсии можно получить явные схемы умножения, которые рассматриваются в следующем параграфе.

Оценим сложность $T(n)$ алгоритма умножения n-разрядных чисел (полиномов степени $n - 1$). В соответствии с описанием алгоритма Карацубы

$$T(n) = \begin{cases} k, & \text{при} \quad n = 1, \\ sT(n/2) + kn & \text{при n>1} \end{cases} \tag{3.3}$$

Здесь k – константа, отражающая сложения и сдвиги. Решение уравнения 2.1 ограничено сверху функцией

$$3kn^{\log\ 3} \approx 3kn^{\log\ 1.59}.$$

Действительно, индукцией по k (Полагая, что n есть степень числа 2, $n = 2^s$), можно показать, что

$$T(n) = 3kn^{\log\ 3} - 2kn. \tag{3.4}$$

Базис, $s = 1,\ n = 1$ тривиален. Далее, если предположить, что функция (2.2) удовлетворяет (2.1) при $n = 2^s$, То

$$T(2n) = 3T(n) + 2kn = 3(skn^{\log\ 3} - 2kn) + 2kn = 3k(2n)^{\log\ 3} - 2k(2n),$$

то есть эта функция удовлетворяет (2.1) и при $m = 2^{s+1} = 2n$.

Отсюда следует, что $T(n) \leq 3kn^{\log\ 3}$.

3.2.3 Оптимизация операций умножения многочленов. Итерации метода Карацубы.

Будем рассматривать многочлены в виде $\sum_{i=0}^{s} Q_i(x)x^{ik}$, где $Q_i(x)$ - элементарные многочлены степени не выше $k-1$. Рассмотрим матрицу T с 2^k строками и 2^k столбцами, в которой элемент t_{ij}, находящийся на пересечении i-й строки и j-го столбца определяется следующим образом. Пусть $a_0a_1\dots a_{k-1}$ - двоичное разложение числа $i-1$, т.е. $i-1=\sum_{r=0}^{k-1} a_r2^r$, а $b_0b_1\dots b_{k-1}$ - двоичное разложение числа $j-1$. Тогда произведение элементарных многочленов $Q^{(1)}, \quad Q^{(1)}=\sum_{r=0}^{k-1} a_rx^r$ и $Q^{(2)}=\sum_{r=0}^{k-1} b_rx^r$ равно $P(x)=Q^{(3)}x^k+Q^{(4)}$ для некоторых элементарных многочленов $Q^{(3)}$ и $Q^{(4)}$. Полагаем $t_{ij}=P(x)$.

Если вычислить матрицу T заранее, то умножение элементарных многочленов сводится к отысканию нужного элемента матрицы T, что ускоряет процедуру умножения многочленов.

Пример. Пусть $k=8$ и нужно перемножить многочлены $P_1(x)=(1+x^2)x^8+(x^3+1)$ и $P_2(x)=(1+x+x^7)x^{16}+x^6$. Тогда $P_1(x)P_2(x)=(1+x^2)(1+x+x^7)x^{24}+(x^3+1)(1+x+x^7)x^{16}+(1+x^2)x^6x^8+(x^3+1)x^6$. Обращаясь к заранее вычисленной таблице произведений элементарных многочленов, получаем

$$(1+x^2)(1+x+x^7)=xx^8+(1+x+x^2+x^3+x^7),$$
$$(x^3+1)(1+x+x^7)=x^2x^8+(1+x+x^3+x^4+x^7),$$
$$(1+x^2)x^6=x^8+x^6,$$
$$(x^3+1)x^6=xx^8+x^6,$$

Откуда $P_1(x)P_2(x)=xx^{32}+(1+x+x^2+x^3+x^7)x^{24}+x^2x^{24}+(1+x+x^3+x^4+x^7)x^{16}+x^{16}+x^6x^8+xx^8+x^6=xx^{32}+(1+x+x^3+x^7)x^{24}+(x+x^3+x^4+x^7)x^{16}+(x+x^6)x^8+x^6$.

Оценим объем памяти, используемой для хранения матрицы T при $k=8$. Таблица T содержит 2^{16} элементов, каждый из которых занимает 2 байта. Поэтому для хранения матрицы T нужно 2^{17} байт или 128 Kb, что составляет вполне разумный объем памяти для современных вычислительных средств. С другой стороны, реализация умножения двух многочленов степени не выше $2^8-1=255$ обычным "школьным"алгоритмом приведет к не более чем $(\frac{2^8}{k})^2=2^{10}$ обращениям к таблице T и не более чем 2^{10} байтовым логическим сложениям, в то время, как число операций без использования матрицы T оценивается как $(2^8)^2=2^{16}$.

С использованием предварительных вычислений можно уменьшить время работы алгоритма деления с остатком многочлена $P_1(x)$ на $P_2(x)$ следующим образом. Через d_i обозначим $\deg P_i(x), \quad i=1,2$. Пусть $d_1\geq d_2$. Найдем элементарный многочлен $Q(x)$ такой, что $\deg(P_1(x)+Q(x)P_2(x))x^{n_1-n_2-k+1}\leq d_1-k$ при $d_1-d_2\geq k$ или $\deg(P_1(x)+Q(x)P_2(x))<d_2$, $d_1-d_2<k$. Для этого в случае $d_1-d_2\geq k$ (при $d_i\geq k$) представим многочлен $P_i(x)$ в виде $Q^{(i)}(x)x^{d_i-k+1}+P^{(i)}(x)$, $\deg P^{(i)}(x)\leq d_i-k$ или (при $d_i<k$) положим $Q^{(i)}(x)=P_i(x) \quad i=1,2$. В случае $d_1-d_2<k$ представим многочлен $P_1(x)$ в виде $Q_1(x)x^{d_2}+P^{(1)}(x)$ с $\deg P^{(1)}(x)<d_2$ и $P_2(x)$ в виде

$Q_2(x)x^{2d_2-d_1} + P^{(2)}(x) \quad \deg P^{(2)}(x) < 2d_2 - d_1$ (при $2d_2 - d_1 \geq 0$) или положим $Q_2(x) = P_2(x)$ (при $2d_2 - d_1 < 0$). Ясно, что многочлен $Q(x)$ однозначно определяется по паре элементарных многочленов $Q^{(i)}(x), \quad i = 1, 2$. Таким образом, можно заранее вычислить таблицу такого соответствия и использовать ее при реализации алгоритма деления многочленов с остатком.

Многочлен $P(x), \quad P(x) = \sum_{i=0}^{r} a_i(x)x^{ik}$, где a_i -элементарные многочлены в дальнейшем будем обозначать через $\sum_{i=0}^{r} a_i y^i$, используя вместо x^k переменную y. Пусть заданы два многочлена $P_i(y), \quad i = 1, 2$ степени (по переменной y) не большей $2t + 1$. Тогда найдутся многочлены $P_i^{(j)}(y), \quad j = 1, 2, \quad i = 1, 2$, степень каждого из которых не превосходит t и $P_i(y) = P_i^{(1)}(y) + y^{t+1}P_i^{(2)}(y)$. Пусть нужно вычислить произведение $P_1(y)P_2(y)$. Алгоритм Карацубы состоит в следующем. Вычислим три произведения: $A(y) = P_1^{(1)}(y)P_2^{(1)}(y), \quad B(y) = P_1^{(2)}(y)P_2^{(2)}(y), \quad C(y) = (P_1^{(1)} + P_1^{(2)})(P_2^{(1)} + P_2^{(2)})$, а затем воспользуемся тождеством $P_1(y)P_2(y) = A(y) + (C(y) + A(y) + B(y))y^{t+1} + B(y)y^{2t+2}$. Таким образом, мы свели умножение двух многочленов к трем умножениям многочленов, имеющих "почти"в два раза меньшую степень. Применим эту процедуру далее к полученным трем произведениям, пока не сведем задачу к умножению элементарных многочленов. Сложность по времени алгоритма Карацубы составляет $O(n^{\log_2 3})$, что по порядку меньше, чем сложность "школьного"алгоритма умножения ($O(n^2)$).

Пример. Проиллюстрируем алгоритм Карацубы на примере многочленов переменной x. Пусть нужно перемножить многочлены $P_1(x) = 1 + x + x^3$ и $P_2(x) = 1 + x^2 + x^3$. Положим $P_1^{(1)}(x) = 1 + x, \quad P_1^{(2)}(x) = x, \quad P_2^{(1)}(x) = 1, \quad P_2^{(2)}(x) = 1 + x$. Тогда $P_i(x) = P_i^{(1)}(x) + x^2P_i^{(2)}(x), \quad i = 1, 2$, Поэтому $P_1(x)P_2(x) = (1+x)(1)+((1+x+x)(1+1+x)+(1+x)(1)+x(1+x))x^2+x(1+x)x^4$. Из трех полученных произведений $(1 + x)1, \quad 1x, \quad x(1 + x)$ в дальнейшем рассмотрении нуждается только $x(1 + x)$, которое по алгоритму Карацубы представляется в виде $0 \cdot 1 + ((0 + 1)(1 + 1) + 0 \cdot 1 + 1 \cdot 1)x + 1 \cdot 1x^2$.

3.2.4 Умножение <длинных> целых чисел

Рассмотрим способ умножения <длинных> целых чисел, который предусматривает разбиение сомножителей на байты и использование таблиц умножения байт.

Пусть <длинное> целое число L состоит из 32-х бит, т.е. из четырех байт. На этапе, предшествующем выполнению умножений <длинных> целых чисел, строится таблица T, состоящая из $2^8 = 256$ строк и такого же количества столбцов. Элемент t_{ij} таблицы T, находящийся в i-ой строке и j-ом столбце этой таблицы, $i = 0, 1, \ldots, 255, \quad j = 0, 1, \ldots, 255$, представляет собой два байта, определяемых следующим образом. Пусть $i = \sum_{k=0}^{7} a_k 2^k$ и $j = \sum_{k=0}^{7} b_k 2^k$, $a_k, b_k \in \{0, 1\}$, двоичные разложения чисел i и j соответственно.

Пусть $\sum_{k=0}^{15} c_k x^k$ – результат умножения многочлена $\sum_{k=0}^{7} a_k x^k$ на многочлен

$\sum_{k=0}^{7} b_k x^k$. Тогда пара байт $C_0 = c_0 c_1 \dots c_7$ и $C_1 = c_8 c_9 \dots c_{15}$ составляет элемент t_{ij}.

Пусть заданы два <длинных> целых числа A и B, каждое из которых состоит из четырех байт:

$$A = A[0]A[1]A[2]A[3],\ B = B[0]B[1]B[2]B[3].$$

Используя таблицу T, построенную нами для умножения байт, сначала найдем следующие числа:

$$\begin{gathered} O_i = A[i] \times B[i], \quad i = 0,1,2,3, \\ T_{i,1} = (A[2i] + A[2i+1]) \times (B[2i] + B[2i+1]),\ i = 0,1, \\ T_{i,2} = (A[i] + A[i+2]) \times (B[i] + B[i+2]),\ i = 0,1, \\ F = (A[0] + A[1] + A[2] + A[3]) \times \\ \times (B[0] + B[1] + B[2] + B[3]). \end{gathered} \tag{3.5}$$

Каждое из чисел $O_i, \quad i = 0,1,2,3, \quad T_{i,1}, T_{i,2}, \quad i = 0,1, \quad F$ задается парой байт. Таким образом, приведенные формулы определяют значения байт

$$O_i[0], O_i[1] \quad i = 0,1,2,3,$$

$$T_{i,1}[0], T_{i,1}[1],\ T_{i,2}[0], T_{i,2}[1] \quad i = 0,1,$$

$$F[0], F[1].$$

Используя эти числа, находим следующие величины:

$$\begin{gathered} D[1] = O_1[0] + O_0[1],\ S[1] = O_0[0] + D[1], \\ D[i] = O_{i((\mathrm{mod}4)}[0] + O_{i-1((\mathrm{mod}4)}[1],\ S[i] = S[i-1] + D[i], \\ i = 2,3,\dots,6. \end{gathered} \tag{3.6}$$

Если последовательность байт

$$P[0]P[1]P[2]P[3]P[4]P[5]P[6]P[7]$$

задает искомое произведение P многочленов, то

$$\begin{gathered} P[0] = O_0[0], \\ P[1] = S[1] + T_{0,1}[0], \\ P[2] = S[2] + T_{0,1}[1] + T_{0,2}[0], \\ P[3] = S[3] + \\ + T_{0,1}[0] + T_{1,1}[0] + T_{0,2}[0] + T_{0,2}[1] + T_{1,2}[0] + F[0], \\ P[4] = S[4] + \\ + T_{0,1}[1] + T_{1,1}[1] + T_{0,2}[1] + T_{1,2}[0] + T_{1,2}[1] + F[1], \\ P[5] = S[5] + T_{1,1}[0] + T_{1,2}[1], \\ P[6] = S[6] + T_{1,1}[1], \\ P[7] = O_3[1]. \end{gathered} \tag{3.7}$$

3.2.5 Декомпозиционная схема умножения многочленов

Рассмотрим способ построения схем умножения, с учетом размерностей сомножителей при этом будем использовать функцию умножения <длинных> (s-разрядных) целых чисел, реализация которой описана в предыдущем параграфе. Таким образом, предполагаем, что для любых двух <длинных> целых чисел A и B определен результат их умножения $S = A \times B$, коэффициенты которого задаются последовательностью из двух <длинных> целых чисел $S^{(0)}, S^{(1)}$.

На вход алгоритма поступают две последовательности, каждая из которых состоит из k <длинных> целых чисел, количество разрядов s которых совпадает с разрядностью используемого для вычислений процессора.

Схема вычислений по рассматриваемому алгоритму содержит два вспомогательных уровня и уровень вычисления результата.

На первом вспомогательном уровне вычисляются некоторые суммы <длинных> целых чисел, поступающих на вход алгоритма. Получаются новые <длинные> целые числа, которые перемножаются на этом уровне определенным образом. Каждое из полученных чисел многократно используется при дальнейших вычислениях.

Результаты умножений поступают на вход следующего уровня схемы, где путем рациональных суммирований находятся новые <длинные> целые числа. На последнем уровне выполняются также только сложения таких чисел.

Таким образом, декомпозиция операции умножения имеет вид $\Sigma\Sigma(\Pi\Sigma)$, то есть сначала выполняются предварительные сложения, потом – умножения длинных цепочек, далее – сложения с целью оптимизации вычислений с многократно используемыми числами и, наконец, – вычисления результата, использующие только сложения.

Пример. Случай $n = 6$ и $s = 32$.

Рассматриваемый при выбранных значениях n и s вариант схемы умножения позволяет перемножить многочлены степени не большей $n \cdot s - 1 = 191$.

Пусть на вход алгоритма подаются две последовательности из шести целых чисел:

$$A^{(0)}, A^{(1)}, A^{(2)}, A^{(3)}, A^{(4)}, A^{(5)},\ B^{(0)}, B^{(1)}, B^{(2)}, B^{(3)}, B^{(4)}, B^{(5)},$$

задающих соответственно многочлены $f(x)$ и $g(x)$. Сначала определим следу-

ющие произведения целых чисел (этапы $\Pi\Sigma$ в структурной схеме $\Sigma\Sigma(\Pi\Sigma)$)

$$\begin{gathered}
O_i = A^{(i)} * B^{(i)}, \quad i = 0, 1, \ldots, 5, \\
T_{1,i} = \left(A^{(2i)} + A^{(2i+1)}\right) * \left(B^{(2i)} + B^{(2i+1)}\right), \quad i = 0, 1, 2 \\
T_{2,i} = \left(A^{(i)} + A^{(i+2)}\right) * \left(B^{(i)} + B^{(i+2)}\right), \quad i = 0, 1 \\
T_{3,i} = \left(A^{(i)} + A^{(i+4)}\right) * \left(B^{(i)} + B^{(i+4)}\right), \quad i = 0, 1, \\
F_{1,0} = \left(A^{(0)} + A^{(1)} + A^{(2)} + A^{(3)}\right) * \\
\left(B^{(0)} + B^{(1)} + B^{(2)} + B^{(3)}\right), \\
F_{2,0} = \left(A^{(0)} + A^{(1)} + A^{(4)} + A^{(5)}\right) * \\
\left(B^{(0)} + B^{(1)} + B^{(4)} + B^{(5)}\right), \\
F_{3,i} = \left(A^{(i)} + A^{(i+2)} + A^{(i+4)}\right) * * \\
\left(B^{(i)} + B^{(i+2)} + B^{(i+4)}\right), \quad i = 0, 1, \\
E = \left(A^{(0)} + A^{(1)} + A^{(2)} + A^{(3)} + A^{(4)} + A^{(5)}\right) * \\
\left(B^{(0)} + B^{(1)} + B^{(2)} + B^{(3)} + B^{(4)} + B^{(5)}\right).
\end{gathered} \tag{4.1}$$

Так как каждое из чисел

$$O_i,\ T_{1,i},\ T_{2,i},\ T_{3,i},\ F_{1,0},\ F_{2,0},\ F_{3,i},\ E$$

задается последовательностью из двух целых s-разрядных чисел, то определены целые числа

$$O_i^{(0)}, \quad O_i^{(1)}, \quad i = 0, 1, 2, \ldots, 5,$$

$$T_{1,i}^{(0)}, \quad T_{1,i}^{(1)}, \quad i = 0, 1, 2,$$

$$T_{j,i}^{(0)}, \quad T_{j,i}^{(1)}, \quad j = 2, 3, \quad i = 0, 1,$$

$$F_{j,0}^{(0)}, \quad F_{j,0}^{(1)}, \quad j = 1, 2,$$

$$F_{3,i}^{(0)}, \quad F_{3,i}^{(1)}, \quad i = 0, 1,$$

$$E^{(0)}, \quad E^{(1)}.$$

Используя эти числа, находим следующие величины:

$$\begin{gathered}
S^{(1)} = O_0^{(0)} + O_1^{(0)} + O_0^{(1)}, \\
S^{(i)} = S^{(i-1)} + O_i^{(0)} + O_{i-1}^{(1)}, \quad i = 2, 3, \ldots, 5, \\
S^{(6)} = S^{(5)} + O_2^{(0)} + O_4^{(0)} + O_5^{(1)}, \\
S^{(7)} = S^{(6)} + O_3^{(0)} + O_5^{(0)} + O_2^{(1)} + O_4^{(1)}, \\
S^{(8)} = O_0^{(1)} + O_1^{(1)} + O_1^{(0)} + O_4^{(0)}, \\
S^{(9)} = O_1^{(1)} + O_4^{(1)} + O_4^{(0)} + O_5^{(0)}, \\
S^{(10)} = O_4^{(1)} + O_5^{(1)} + O_5^{(0)},
\end{gathered} \tag{4.2}$$

$$\begin{aligned} R^{(3)} &= T_{1,0}^{(0)} + T_{1,1}^{(0)}, \\ R^{(4)} &= T_{1,0}^{(1)} + T_{1,1}^{(1)}, \\ R^{(5)} &= R^{(3)} + T_{1,2}^{(0)}, \\ R^{(6)} &= R^{(4)} + T_{1,2}^{(1)}, \end{aligned} \tag{4.3}$$

$$\begin{aligned} K^{(3)} = K^{(7)} &= T_{2,0}^{(1)} + T_{2,0}^{(0)} + T_{2,1}^{(0)}, \\ K^{(4)} = K^{(8)} &= T_{2,0}^{(1)} + T_{2,1}^{(0)} + T_{2,1}^{(1)}, \end{aligned} \tag{4.4}$$

$$\begin{aligned} C^{(7)} &= F_{1,0}^{(0)} + F_{2,0}^{(0)} + F_{3,0}^{(0)} + F_{3,1}^{(0)} + F_{3,0}^{(1)}, \\ C^{(8)} &= F_{1,0}^{(1)} + F_{2,0}^{(1)} + F_{3,0}^{(1)} + F_{3,1}^{(1)} + F_{3,1}^{(0)}, \end{aligned} \tag{4.5}$$

$$\begin{aligned} W^{(5)} &= T_{3,0}^{(0)} + T_{3,0}^{(1)} + T_{3,1}^{(0)}, \\ W^{(6)} &= W^{(5)} + T_{3,1}^{(1)}, \\ W^{(7)} &= W^{(6)}, \\ W^{(8)} &= W^{(7)} + T_{3,0}^{(0)}, \end{aligned} \tag{4.6}$$

Наконец, определяем последовательность $P^{(0)}, P^{(1)}, P^{(2)}, \ldots, P^{(11)}$ целых чисел, задающую произведение многочленов $f(x)$ и $g(x)$.

$$\begin{aligned} P^{(0)} &= O_0^{(0)}, \\ P^{(1)} &= S^{(1)} + T_{1,0}^{(0)}, \\ P^{(2)} &= S^{(2)} + T_{1,0}^{(1)} + T_{2,0}^{(0)}, \\ P^{(3)} &= S^{(3)} + R^{(3)} + K^{(3)} + F_{1,0}^{(0)}, \\ P^{(4)} &= S^{(4)} + R^{(4)} + K^{(4)} + F_{1,0}^{(1)} + T_{3,0}^{(0)}, \\ P^{(5)} &= S^{(5)} + R^{(5)} + T_{2,1}^{(1)} + F_{2,0}^{(0)} + W^{(5)}, \\ P^{(6)} &= S^{(6)} + R^{(6)} + T_{2,0}^{(0)} + F_{2,0}^{(1)} + F_{3,0}^{(0)} + W^{(6)}, \\ P^{(7)} &= S^{(7)} + T_{1,0}^{(0)} + K^{(7)} + C^{(7)} + W^{(7)} + E^{(0)}, \\ P^{(8)} &= S^{(8)} + T_{1,0}^{(1)} + K^{(8)} + C^{(8)} + W^{(8)} + E^{(1)}, \\ P^{(9)} &= S^{(9)} + T_{1,2}^{(0)} + T_{2,1}^{(1)} + T_{3,1}^{(1)} + F_{3,1}^{(1)}, \\ P^{(10)} &= S^{(10)} + T_{1,2}^{(1)}, \\ P^{(11)} &= O_5^{(1)}, \end{aligned} \tag{4.7}$$

Таким образом, нами построена схема умножения многочленов при $n = 6, \quad s = 32$.

Как видим, известная схема умножения многочленов по методу Карацубы представлена нами не в рекуррентном, а в явном виде. При этом разработанная схема вычислений, как уже было указано выше, имеет структуру $\Sigma\Sigma(\Pi\Sigma)$ и содержит программные эвристики.

Приведенный способ умножения многочленов допускает, очевидно, распараллеливание вычислений как внутри уровней, так и в целом.

Подобные явные схемы процедурно строятся и при бо̀льших параметрах операндов.

3.2.6 Результаты экспериментов

Осуществлена тестовая проверка следующих способов умножения многочленов в стандартном базисе:

С – классический метод умножения на уровне отдельных коэффициентов;

СТ – метод умножения на уровне элементарных многочленов степени 7 ($k = 8$) с использованием заранее вычисляемой таблицы умножения элементарных многочленов;

СМ – описанная в предыдущем параграфе модификация классического метода, минимизирующая число операций сдвига и сложения.

КМ – умножение многочленов по методу Карацубы (декомпозиционная схема не используется).

КС – умножение многочленов по декомпозиционной схеме.

Для получения временных экспериментальных оценок производились по 10 000 циклов умножения (с приведением результата по модулю пятичлена) с определением минимального, максимального и среднего по 10 таким испытаниям времени выполнения этих циклов. В каждом цикле осуществлялось приведение результата умножения к стандартному базису поля Галуа путем нахождения остатка от деления на неприводимый пятичлен соответствующей степени.

Результаты испытаний с использованием процессора Pentium MMX, 233 Мгц приведены в таблице (время указано в секундах). Приведенные данные получены с использованием экспериментальной библиотеки арифметических операций в конечных полях, описанной ниже в параграфе 4.3.1.

173			191			239			
Мин.	Ср.	Макс.	Мин.	Ср.	Макс.	Мин.	Ср.	Макс.	Метод
1.26	1.40	1.56	1.42	1.56	1.71	1.76	1.97	2.11	С
0.53	0.62	0.68	0.61	0.69	0.78	0.81	0.95	1.05	СТ
0.71	0.74	0.76	0.73	0.76	0.78	0.80	0.83	0.88	СМ
0.45	0.46	0.48	0.47	0.49	0.51	0.50	0.52	0.55	К
0.17	0.18	0.18	0.22	0.23	026	0.36	0.37	0.39	КС

Таким образом, выявленные сравнением теоретических оценок сложности преимущества умножения по предложенной в данной работе схеме перед классическим методом и базовым методом Карацубы подтверждаются экспериментально.

Отметим, что вычисления в соответствии с такими схемами можно осуществить также посредством логических схем. Так на Рис.2.1 дано блочное представление схемы умножения <длинных> целых чисел, рассмотренной в параграфе 4.4.

Вычисления в блоках Σ^1 и П реализуются по формулам (2.3) В блоке Σ^2, выделенном жирными линиями, вычисления имеют итеративный характер в соответствии с формулами (2.4). В блоке Σ^3 осуществляются заключительные суммирования по формулам (2.5).

Умножение многочленов степени 191 можно осуществить посредством логической схемы, блочное представление которой дано на Рис.2.2.

Блоки Σ^1 и П реализуют вычисления по формулам (2.6). При этом умножения в блоке П могут быть осуществлены по схемам умножения <длинных> целых чисел на Рис.2.1. Вычисления в блоках Σ^2, Σ^3 и Σ^5, выделенных жирными линиями, в соответствии с формулами (2.7), (2.8) и (2.11) имеют итеративный характер. Суммирование в блоках Σ^4, Σ^6 осуществляется по формулам (2.9) и (2.10). В блоке Σ^7 формируется окончательный результат P по формулам (2.12.)

3.3 Еще две модификации алгоритма умножения

После опубликования первого издания нашей брошюры авторам стала известна публикация J.Lopez,R.Dahab в AsiaCrypt-2000, в которой предлагается два варианта алгоритма умножения, первый из которых не требует дополнительной памяти а второй требует обращения к предвычисленной таблице меньшего размера, чем используемая в изложенном выше алгоритме. Далее излагаются оба этих метода, следуя статье авторов с некоторыми нашими комментариями.

Второй алгоритм является разивитием первого, поэтому начинаем с него. Как отмечают авторы, он является модификацией алгоритма [124] Напомним, что этот алгоритм заключается в следующем: показатель степени записывается 2^k-ичной системе

$$a = \sum_{i=0}^{s-1} A_i 2^{ki},$$

где A_i имеет двоичную запись $(a_{ik+k-1} \dots a_{ik})_2$, степени базы g, записываемые далее для удобства в аддитивной нотации, предвычисляются в порядке $2^{ki}g, 0 \le i < s$, потом для каждого $u = (u_{s-1} \dots u_0)_2, 0 \le u < 2^s$ вычисляются в порядке возрастания чисел $u_{s-1} + \dots + u_0$) суммы

$$P_u = \sum_{i=0}^{s-1} u_i 2^{ki} g,$$

а потом вычисляется степень $a \cdot g$ по формуле

$$a \cdot g = \sum_{j=0}^{k-1} 2^j \left(\sum_{i=0}^{s-1} a_{ik+j} 2^{ki} \cdot g \right) = \sum_{j=0}^{k-1} 2^j P_{I_j},$$

где $I_j = (a_{(s-1)k+j} \dots a_{k+j} a_j)_2$. Преимущество этой формулы перед стандартной формулой

$$a \cdot g = \sum_{j=0}^{j=s-1} 2^{kj} \left(\sum_{i=0}^{k-1} a_{jk+i} 2^i \cdot g \right)$$

в том, что при условии предвычисления внутренних сумм упрощается вычисление степени $a \cdot g$ за счет уменьшения числа возведений в квадрат (удвоений

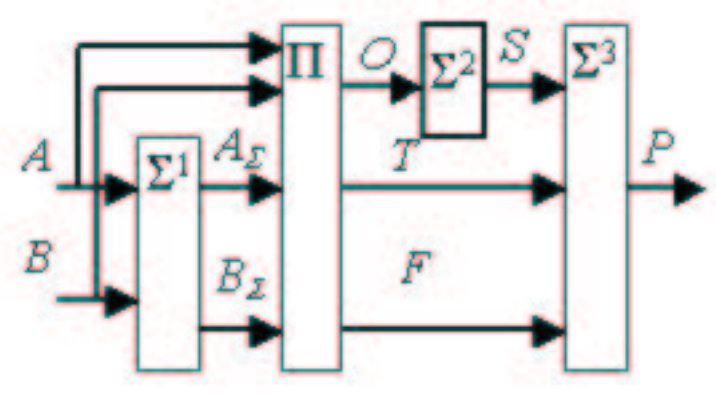

Рис. 3.1: Базовая декомпозиционная схема.

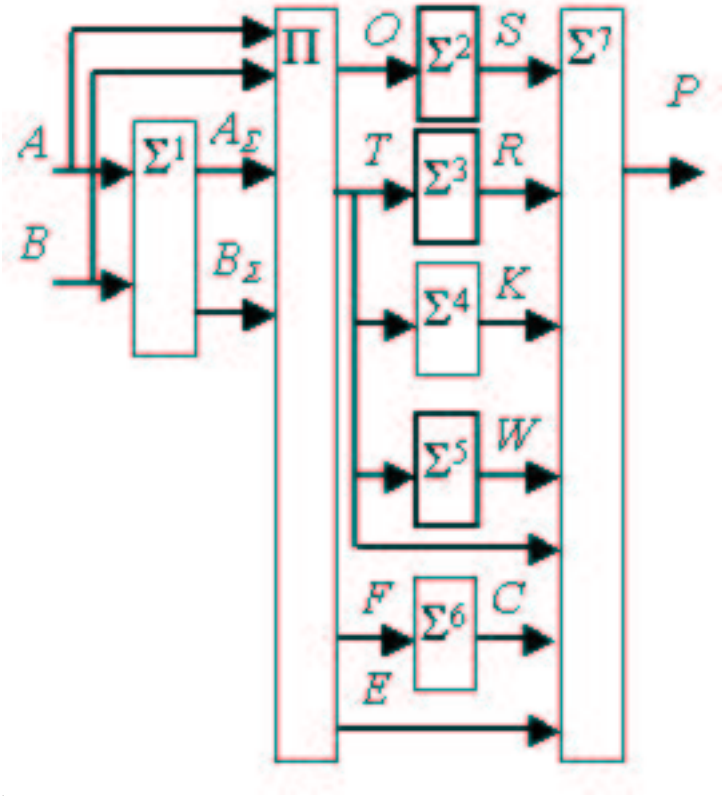

Рис. 3.2: Главная декомпозиционная схема.

в аддитивной нотации), которых становится только k вместо $(s-1)k$ во второй формуле. Однако само предвычисление становится немного более сложным.

Алгоритмы экспоненцирования в аддитивной нотации легко могут быть модифицированы в алгоритмы умножения как чисел, так и многочленов. Например, известный еще древним индусам бинарный алгоритм экспоненцирования в 19 веке широко использовался в России для умножения чисел, и Д.Кнут называет его в своей книге «русским крестьянским методом». Не удивительно, что и указанный выше алгоритм можно превратить в алгоритм умножения многочленов. Для этого будем представлять любой двоичный многочлен $a(X) = \sum_{i=0}^{n-1} a_i X^i$, как и раньше, в виде

$$a(X) = \sum_{i=0}^{s-1} A_i(X) X^{ki}, A_i(X) = \sum_{j=0}^{k-1} a_{ik+j} X^j,$$

и хранить в машинной памяти в виде массива чисел $A_0, \ldots, A_{s-1}$, где A_i имеет двоичную запись $(a_{ik+k-1} \ldots a_{ik})_2$, а размер k можно выбрать равным величине машинного слова, и положить $k = 32$, а не 8, как было у нас раньше. Тогда приведенная выше формула для экспоненцирования может быть переписана в виде следующей формулы умножения многочленов

$$a(X)g(X) = \sum_{j=0}^{k-1} X^j \left(\sum_{i=0}^{s-1} a_{ik+j} X^{ki} \cdot g \right)$$

Произведение $b(X) = a(X)g(X)$ имеет степень $2n-2$ и представляется массивом коэффициентов $B_{2s-1}, \ldots, B_0$. В начале он заполняется нулями. Потом выполняется главный цикл, в котором j меняется от $k-1$ до нуля, и на каждом его шаге полученный ранее многочлен вида

$$Q_{j-1} = \sum_{l=j+1}^{l=k-1} X^{l-j} \left(\sum_{i=0}^{s-1} a_{ik+l} X^{ki} \cdot g \right) = \sum_{r=0}^{2s-1} B_r(X) X^{kr}$$

соответственно схеме Горнера умножается на X и и складывается с многочленом

$$T_j = \sum_{i=0}^{s-1} a_{ik+j} X^{ki} \cdot g = \sum_{i=0}^{2s-1} B_i(X) X^{ki} = \sum_{r=0}^{2s-1} B'_r(X) X^{kr}.$$

Умножение на X сводится к сдвигу каждого из чисел B_i в сторону старших разрядов, что требует одной операции над каждым словом, но предварительно надо сделать с каждым словом сдвиг на 31 бит в сторону младщих разрядов, чтобы не пропал при первом сдвиге старший бит, который потом надо будет прибавлять к числу B_{i+1}, для чего нужна будет еще одна операция XOR или покомпонентная дизъюнкция. Всего для выполнения умножения на X нужно $2s$ операций XOR и $4s$ операций шифт (так будем называть сдвиги битов в машинных словах). Для прибавления к Q_{j-1} очередного многочлена

$$T_j = \sum_{i=0}^{s-1} a_{ik+j} X^{ki} \cdot g = \sum_{r=0}^{2s-1} B_r(X) X^{kr}$$

выполняем внутренний цикл по i от нуля до $s-1$, на каждом шаге которого вычисленный ранее многочлен

$$Q_{j-1}+T_{j,i-1}=Q_{j-1}+\sum_{l=0}^{i-1}a_{lk+j}X^{kl}\cdot g=\sum_{r=0}^{s+i-2}B_r(X)X^{kr}$$

складываем с многочленом

$$R_{j,i}=a_{(i-1)k+j}X^{ki}\cdot g=\sum_{r=0}^{s+i-1}B'_r(X)X^{kr}.$$

Преимущество указанной формулы над формулой с другим порядком суммирования в том, что умножение на $X^{k(i-1)}$ делается в этом цикле «бесплатно», так сводится только к «шифту» самого массива коэффициентов B_i, а последующее сложение многочленов сводится к s операциям XOR над машинными словами: $B_{l+i-1}=B_{l+i-1}\oplus G'_l, l=0,\ldots,s-1$, где G_i — массив, задающий многочлен $g(X)$, и выполняется оно только, если $a_{(i-1)k+j}$ —- j-й бит числа A_{i-1}, равен 1.

Общее число операций, выполняемых в этом алгоритме равно $s^2k+2s(k-1)=n^2/k+2n-2s$ операций XOR и $4s(k-1)=4n-4s$ операций шифт (предполагаем, что $n=sk$).

Опишем теперь второй алгоритм, который является ускорением первого за счет предвычисления произведений вида

$$P_u=(u_{m-1}X^{m-1}+\ldots u_0)g(X), u=(u_{m-1}\ldots u_0)_2, 0\le u<2^m, m<k.$$

На самом деле это не совсем предвычисление, так как оно делается подпрограммой нашей программы, и его сложность надо учитывать при вычислении сложности всего алгоритма умножения. Далее можно считать, что $m=4$, но почему такой выбор оптимален, будет обяснено позднее.

Для вычисления всех произведений $P_u\cdot g$ нужно вначале вычислить все произведения $X^jg(X), 0\le j<m$, для чего потребуется $sm=nm/k$ операций XOR и $2sm=2nm/k$ операций шифт. После этого для вычисления всех $P_u\cdot g$, представленных в виде сумм

$$P_u\cdot g=u_{m-1}X^{m-1}g(X)+\ldots u_0g(X), u=(u_{m-1}\ldots u_0)_2,$$

достаточно выполнить 2^m-2 операции сложения многочленов, если вычислять эти суммы в порядке роста величин $u_{m-1}+\ldots u_0$, т.е достаточно $(s+1)(2^m-2)$ операций XOR с машинными словами.

Далее для умножения можно воспользоваться формулой

$$a(X)g(X)=\sum_{j=0}^{k-1}X^j\left(\sum_{i=0}^{s-1}a_{ik+j}X^{ki}\cdot g\right)$$

$$= \sum_{j=0}^{k/m-1} X^{mj} \left(\sum_{i=0}^{s-1} X^{ki} \sum_{l=0}^{m-1} a_{ik+mj+l} X^l \cdot g \right)$$

$$= \sum_{j=0}^{k/m-1} X^{mj} \left(\sum_{i=0}^{s-1} X^{ki} P_{u_{i,j}} \cdot g \right), u_{i,j} = (a_{ik+mj+m-1} \dots a_{ik+mj})_2$$

Алгоритм после выполнения предвычисления $P_u \cdot g$ работает подобно первому алгоритму. Так как все многочлены $P_u \cdot g$ уже вычислены, нам для вычисления произведения достаточно обратиться $s^2k/m = n^2/km$ раз к оперативной памяти для извлечения коэффициентов многочленов $P_{u_{i,j}} \cdot g$, умножение этих многочленов на X^{ki} делается бесплатно, для вычисления всех сумм

$$\sum_{i=0}^{s-1} X^{ki} P_{u_{i,j}} \cdot g, j = 0, \dots, k/m$$

нужно $s^2k/m = n^2/km$ операций XOR, и для окончательного вычисления

$$= \sum_{j=0}^{k/m-1} X^{mj} \left(\sum_{i=0}^{s-1} X^{ki} P_{u_{i,j}} \cdot g \right), u_{i,j} = (a_{ik+mj+m-1} \dots a_{ik+mj})_2$$

нужно еще $2s(k/m-1) = 2n/m - 2s$ операций XOR и $4s(k/m-1) = 4n/m - 4s$ операций шифт. Общая сложность алгоритма равна $n^2/km + 2n/m - 2n/k + nm/k + (n/k+1)(2^m-2)$ операций XOR и $4n/m - 4n/k + 2nm/k$ операций шифт, а также n^2/km операций обращения к памяти, объем которой равен $2^m(s+1)$ машинных слов, или $2^m(n/k+1)k/8 = 2^m(n/8+4)$ байт. Для минимизации этой суммы надо минимизировать $n/m + 2^m$ выбрать m так, чтобы $2^m \le n/m$, тогда сумма не превзойдет $2n^2/km + 2n/m - 4n/k - 2 + nm/k$, но будет не меньше $n^2/km + 2n/m - 4n/k - 2 + nm/k$, при этом для ее минимизации надо выбрать m максимальным, удовлетворяющим неравенству $2^m m \le n$. Таким m будет $\log_2 n - \log_2 \log_2 n + O(1)$, т.е. при $n \le 256$ $m \le 5$.

Например, при $160 \le n \le 192, k = 32$ если взять $m = 4$, то число операций XOR будет равно $n^2/128 + n/2 - n/16 + n/8 + (n/32+1)14$ т.е. приблизительно $n(1 + n/128)$. Если же взять $m = 5$, то число операций XOR будет приблизительно равно $n^2/160 + 2n/5 - n/16 + 5n/32 + (n/32+1)30$ т.е. приблизительно $n(1 + (n + 65)/160$, что примерно на 5-8 процентов больше, хотя число остальных операций при $m = 5$ несколько уменьшится. Удобнее взять все же $m = 4$, так как оно делит нацело $k = 32$.

Для сравнения оценим число операций в стандартном алгоритме, описанном нами ранее, в котором однако мы брали $k = 8$, и использовали предвычисляемую «таблицу умножения » объемом 2^{16} байт. В предыдущем алгоритме объем используемой таблицы был $2^m(n/8+4) = 2n+64$ байт, что существенно меньше. Число обращений к таблице в стандартном алгоритме было $(n/k)^2 = n^2/64 \le 3n$ в рассматриваемом диапазоне изменения n, что вдвое больше числа обращений к памяти $n^2/km = n^2/(32 \cdot 4) = n^2/128$ в приведенном выше алгоритме. Число

операций XOR в стандартном алгоритме чуть меньше указанного выше числа умножений с помощью таблицы, а число операций XOR в приведенном выше алгоритме равно приблизительно $n(1+n/128)$, т.е. почти такое же, как и в стандартном алгоритме. В целом же указанный алгоритм лучше стандартного по числу операций примерно раза в два, и требует существенно меньше оперативной памяти. Но его превосходство над описанным выше алгоритмом Карацубы исчезает, за исключением превосходства в обьеме используемой памяти, о чем свидетельствует и полученная авторами алгоритма оценка скорости его работы на процессоре Pentium 233 MHz: операции в поле размерности 163 выполнялись за 10.2 миллионные доли секунды. Так как оценка сложности алгоритма Карацубы по порядку равна $n^{\log_2 3}$ и растет медленнее, чем оценка сложности $n^2/\log n$ указанного алгоритма, естественно ожидать, что с ростом n надлежаще запрограммированный алгоритм Карацубы будет работать быстрее. Заметим однако, что для ускорения работы алгоритма Карацубы, после выполнения определенного числа его итераций при достаточно малом n мы начинали пользоваться стандартным алгоритмом. Видимо целесообразно заменить его на этом этапе указанным выше алгоритмом, и время работы описанной выше версии алгоритма Карацубы уменьшится.

Отметим также, что при возможности использовать большой объем оперативной памяти стандартный алгоритм легко ускорить, выбирая большие значения k, например $k = 12$, объем требуемой памяти будет порядка 30 мегабайт. Можно перейти к использованию и значений k в плоть до 16, но использовать «таблицу умножения» уже становится практически невозможно. Однако вместо нее можно использовать таблицу логарифмов в поле $GF(2^k$ относительно какого-нибудь примитивного элемента, объем которой равен 2^k, и подобную же таблицу антилогарифмов. Умножение в этом поле сводится к трехкратному обращению к таблицам и сложению чисел по модулю $2^k - 1$. Однако полученные коэффициенты произведения многочленов будут вычислены по модулю некоторого многочлена степени k, определяющего указанное поле. Нам же надо вычислять их точно, для чего все операции над многочленами степени не выше $2k - 2$ надо выполнять точно. Поэтому придется продублировать умножение многочленов, используя второй экземпляр поля $GF(2^k$ с другим многочленом в качестве модуля, а потом для восстановления точного результата применить китайскую теорему об остатках. Сложность выполнения последних операций незначительна в сравнении со сложностью операций умножения. Если операции обращения к 2^k байтам памяти выполняются быстрее арифметических операций, то основное время будет уходить на сложение логарифмов по модулю $2^k - 1$, которое можно свести к последовательно выполняемому обычному сложению, выделению k-го бита, его обнулению и прибавлению к результату на месте младшего бита. Кроме того, перед выполнением операции сложения по модулю 2^k-1, над проверить не является ли нулем один из сомножителей, тогда произведение полагается равным нулю без использования обращения к таблице логарифмов. Указанный прием будет эффективнее, если его применять не

к стандартному методу умножения, а к методу Карацубы, так как в нем число умножений меньше числа сложений. Если же объем используемой памяти слишком велик и обращение к ней производится медленнее выполнения операции сложения по модулю $2^k - 1$, то можно коэффициенты перемножаемых многочленов хранит в виде логарифмов, тогда операция умножения коэффициентов не требует обращения к памяти, но для сложения требуется операции сложения и вычитания по модулю $2^k - 1$ и операция обращения к таблице логарифмов Гаусса-Зеха согласно формуле

$$\alpha^X + \alpha^y = \alpha^X(1 + \alpha^{y-X}) = \alpha^{X+z}, z = \log_\alpha(1 + \alpha^{y-X}) = glog_\alpha(y - X).$$

Заметим еще, что в отличие от метода Карацубы, применимого также для построения логических схем для умножения, приведенный выше метод является чисто программным трюком. Однако схему для умножения произвольного многочлена на фиксированный многочлен сложности $n^2/logn$ таким методом все же построить можно, причем эта схема фактически совпадет со схемой умножения двоичного вектора на фиксированную двоичную же матрицу, которую можно построить, применяя алгоритм «четырех русских » или, что равносильно, метод Лупанова синтеза вентильных схем, о котором уже упоминалось выше по другому поводу.

3.4 Умножение многочленов в поле $GF(2^n)$

3.4.1 Деление и приведение многочленов по модулю неприводимого многочлена

3.4.2 «Школьный» алгоритм деления многочленов в стандартном базисе.

С использованием операций сложения и умножения многочленов реализуется алгоритм нахождения частного и остатка при делении одного многочлена $(p_1(x))$ на другой $(p_2(x))$, т.е. нахождение таких многочленов $q(x)$ и $r(x)$, что

$$p_1(x) = q(x)p_2(x) + r(x)$$

и

$$\deg r(x) < \deg p_2(x).$$

Пример 3.4.1 Пусть

$$p_1(x) = x^5 + x^2 + x + 1 \text{ и } \ p_2(x) = x^3 + x^2.$$

Тогда

$$p_1(x) = x^2p_2(x) + x^4 + x^2 + x + 1 = x^2p_2(x) + xp_2(x) + x^3 + x^2 + x + 1 =$$
$$= x^2p_2(x) + xp_2(x) + p_2(x) + x + 1 = (x^2 + x + 1)p_2(x) + x + 1.$$

Поэтому в настоящем примере частное и остаток от деления многочлена $p_1(x)$ на многочлен $p_2(x)$ равны соответственно $q(x) = x^2 + x + 1$ и $r(x) = x + 1$.

Рассмотрим один из алгоритмов деления многочлена $p_1(x)$ на многочлен $p_2(x)$.

Даны векторы U длины p и V длины $s, s \leq p$, коэффициентов многочленов $p_1(x)$ степени $\deg p_1(x) \leq p-1$ и $p_2(x)$ степени $s-1$ в порядке возрастания степеней соответствующих термов многочленов, вектор U образуется двумя векторами U_1 длины $p-s$ (соответствует младшим разрядам и U_2 длины s; для формирования частного используется вектор Z длины $p-s+1$ он образуется двумя векторами Z_1 длины 1 и Z_2 длины $p-s$ (Z_1 соответствует младшему, а Z_2 - старшим разрядам вектора Z);

Требуется вычислить частное и остаток от деления многочлена $p_1(x)$ на многочлен $p_2(x)$.

Алгоритм (так называемый "школьный"алгоритм деления) описывается следующим образом.

ВХОД: вектор $U = U_1||U_2$ длины v коэффициентов
многочлена-делимого $p_1(x)$ степени $\deg p_1 \leq v$.
вектор $V = V_1||V_2$ длины s коэффициентов
могочлена-делителя $p_2(x)$ степени $\deg p_2 \leq s$.
ВЫХОД: вектор $Z = Z_1||Z_2$ длины $v-s$ коэффициентов
многочлена-частного $q(x)$, (длина Z_1 равна 1),
вектор U_2 длины $v-s$ коэффициентов
многочлена-остатка $r(x)$,
1. Принять $Z = 0$,
Если $[U = 1$, то $Z_1 = 1, U_2 = U_2 + V$;
2. Выполнить $p-s-1$ раз
$U = U[\rightarrow], Z = Z[\rightarrow]$,
Если $[U = 1$ то $Z_1 = 1$, $U_2 = U_2 + V$.

Здесь $[U$ – старший элемент вектора U; $[\rightarrow]$ – операция сдвига на одну позицию в сторону старших разрядов (умножение на 2, или на полином x). Элементы вектора U_2 определяют коэффициенты многочлена-остатка; элементы вектора Z являются коэффициентами многочлена-частного.

Пример 3.4.2 Пусть $p = 8,\ s = 5$,

$$p_1(x) = 1 + x + x^2 + x^5,$$

$$p_2(x) = x^2 + x^3.$$

Очевидно (см. выше), что

$$S(x) = 1 + x + x^2;\ R(x) = 1 + x.$$

Вычисления представлены в следующей таблице

шаг	U_1	U_2	V	Z_1	Z_2
1	11	1001	0011	0	00
		0011		1	00
	11	**1010**		**1**	**00**
2^1	01	1101		0	10
		0011		1	10
	01	**1110**		**1**	**10**
2^2	00	1111		0	11
		0011		1	11
	00	1100		**1**	**11**

Жирным шрифтом указаны результаты каждого шага.

Сложность этого алгоритма такая же, как сложность классического алгоритма умножения.

3.4.3 Приведение многочленов по неприводимому «малочлену»

Пусть многочлен $p_1(x) = \sum_{i=0}^{n-1} u_i \times x^i$ необходимо разделить на многочлен $p_2(x) = \sum_{j=0}^{m-1} v_i \times x^i$. Частное представим многочленом $s(x) = \sum_{i=0}^{n'-1} z_i \times x^i$, а остаток – многочленом $r(x) = \sum_{i=0}^{n''-1} u'_i \times x^i$. Полагаем, что многочлен p_2 имеет малое число ненулевых коэффициентов (является «малочленом»), причем коэффициенты $v_{\deg\ p_2-1}, \ldots, v_{\deg\ p_2-s+1}$– нулевые.

Примечание. Если степень многочлена-делимого $p_1(x)$ не превышает $2\deg p_2(x) - 2$, то нулевыми должны быть коэффициенты $v_{\deg\ p_2-1}, \ldots, v_{\deg\ p_2-s+3}$.

Образуем список $L = [r_1, r_2, \ldots, r_l]$ степеней ненулевых коэффициентов многочлена p_2, кроме старшей.

Обозначим

$$p = \deg_s p_1,\ q = \deg_s p_2,$$

$$l_{r_i} = \left[\frac{sp - \deg\ p_2 + r_i}{s}\right], i = 1, \ldots l,$$

$$t_{r_i} = \mathrm{rem}(sp - \deg\ p_2 + r_i, s), i = 1, \ldots l.$$

Предполагается, что эти значения для каждого элемента списка L вычисляются заранее.

Двоичные последовательности коэффициентов многочленов $p_1(x)$, $p_2(x)$, $s(x)$ и $r(x)$ будем обозначать U, V, Z и U' соответственно, а их части, соответствующие разбиению на машинные слова («длинные» целые числа) – $U^{(i)}, i = 0, \ldots, p$ $V^{(i)}, i = 0, \ldots, q$ $Z^{(i)}, i = 0, \ldots, p - q$ $U'^{(i)},\ i = 0 \ldots q$.

Алгоритм вычисления многочлена-остатка $r(x) = \mathrm{rem}(p_1(x), p_2(x)$ описывается следующим образом.

1.Пока $p > q$

а) Если $U^{(p)} \neq 0$, то для каждого элемента r_i списка L выполнить следующее: принять $U^{(l_{r_i})} = U^{(l_{r_i})} + U^{(p_0)}$, $U^{(l_{r_i}+1)} = U^{(l_{r_i}+1)} + U^{(p_1)}$, где U^{p_0} получается из $U^{(p)}$ удалением t_{r_i} старших разрядов и добавлением t_{r_i} нулевых младших разрядов (сдвигом в сторону старших разрядов). U^{p_1} получается из $U^{(p)}$ удалением $s - t_{r_i}$ младших разрядов и добавлением $s - t_{r_i}$ нулевых старших разрядов (сдвигом в сторону младших разрядов).

Формально это действие а) описывается следующим образом:

$$U^{(l_{r_i})} || U^{(l_{r_i}+1)} = U^{(p)} x^{t_{r_i}}, i = 1, \ldots l.$$

б) Положить $p = p-1$ и для каждого элемента r_i списка L принять $l_{r_i} = l_{r_i} - 1$.

2. Если $p = q$ и $\deg U^{(p)} \geq \deg V^{(q)}$

для каждого элемента r_i списка L выполнить:

из вектора $U^{(p)}$ образовать вектор W, путем "обнуления"$\deg V^{(q)}$ младших разрядов, выполнить действия, подобные описанным в п.1,а) с использованием вектора W вместо вектора $U^{(p)}$ с тем отличием, что в случае $l_{r_i} < 0$ изменяется только вектор $U^{(l_{r_i}+1)}$;

"обнулить"старшие $s - \deg V^{(p)}$ разрядов вектора $U^{(p)}$.

Формально это действие б) описывается следующим образом:

$$U^{(l_{r_i})} || U^{(l_{r_i}+1)} = \text{rem } (U^{(p)}, x^{\deg V^{(p)}}) x^{t_{r_i}}, i = 2, \ldots l.$$

$$U^{(1)} = \text{rem } (U^{(p)}, x^{\deg V^{(p)}}).$$

Рассматриваемые при каждой итерации «длинные» целые числа $U^{(p)}$, $p > q$, являются числами $Z^{(p-q)}$, составляющими вектор коэффициентов частного, при этом число $Z^{(0)}$ последнего равно вектору W, получающемуся при выполнении п. 2 алгоритма. «Длинные» целые числа $U^{(j)}, j = 0, \ldots, q$ по окончании алгоритма образуют последовательность U' коэффициентов многочлена-остатка $r(x)$.

Данный вариант деления реализуется наилучшим способом при s, равном длине машинного слова $s = 32, 64$ или 128. При этом одна итерация алгоритма оказывается эквивалентной s итерациям «школьного» алгоритма, то есть их одновременному параллельному выполнению.

Если же $t_{r_l} < \deg p_2 - s + 3$ в случае $\deg p_1(x) \leq 2 \deg p_2(x)$, то удобно использовать виртуальные машинные слова длины $\tilde{s} = sq + 1$, представляя многочлен $p_1(x)$ двумя такими виртуальными словами $\tilde{U}^{(0)}, \tilde{U}^{(1)}$, а многочлен $p_2(x)$ – одним виртуальным словом $\tilde{V}^{(0)}$. При этом алгоритм работает с виртуальными словами, а не с машинными словами. Этот случай имеет место, если делитель $p_2(x)$ степени n является трехчленом вида $1 + x + x^n$.

Для многочленов-делителей более общего вида, таких что $t_{r_l} < \frac{\deg\ p_2(x)}{2}$ удобно использовать виртуальные машинные слова длины $\tilde{s} = \frac{s(q+1)}{2}$, представляя многочлен-делимое $p_1(x)$ четырьмя такими словами

$$\tilde{U}^{(0)}, \tilde{U}^{(1)}, \tilde{U}^{(2)}, \tilde{U}^{(3)},$$

а многочлен-делитель $p_2(x)$ – двумя.

$$\tilde{V}^{(0)}, \tilde{V}^{(1)}.$$

Примечание. В обоих случаях старшее виртуальное слово может быть не полным.

Пример 3.4.3 Пусть все коэффициенты многочлена-делимого $p_1(x)$ степени 319 равны 1:

$U^{(0)}$	$U^{(1)}$	$U^{(2)}$	$U^{(3)}$	$U^{(4)}$	$U^{(5)}$	$U^{(6)}$	$U^{(7)}$	$U^{(8)}$	$U^{(9)}$
1^{32}	1^{32}	1^{32}	1^{32}	1^{32}	1^{32}	1^{32}	1^{32}	1^{32}	1^{32}

а многочлен-делитель является пятичленом

$$p_2(x) = 1 + x^3 + x^6 + x^7 + x^{163}.$$

Длина s машинного слова – 32 двоичных разряда. При делении "школьным"алгоритмом потребуется выполнить около тысячи сложений и сдвигов 32 - разрядных машинных слов.

Вариант А. При реализации алгоритма приведения с использованием реальных машинных слов 32 итерации «школьного» алгоритма будут выполняться параллельно.

Вычислим

$$p = \deg_s p_1 = 9, \quad q = \deg_s p_2 = 5.\ L = [r_1, r_2, r_3, r_4] = [0, 3, 6, 7].$$

$$\begin{aligned}
l_{r_1} &= \lfloor \tfrac{32\cdot 9-163+0}{32} \rfloor = 3;\\
l_{r_2} &= \lfloor \tfrac{32\cdot 9-163+3}{32} \rfloor = 4;\\
l_{r_3} &= \lfloor \tfrac{32\cdot 9-163+6}{32} \rfloor = 4;\\
l_{r_4} &= \lfloor \tfrac{32\cdot 9-163+7}{32} \rfloor = 4;\\
t_{r_1} &= \mathrm{rem}(32\cdot 9 - 163 + 0, 32) = 29;\\
t_{r_2} &= \mathrm{rem}(32\cdot 9 - 163 + 3, 32) = 0;\\
t_{r_3} &= \mathrm{rem}(32\cdot 9 - 163 + 6, 32) = 3;\\
t_{r_4} &= \mathrm{rem}(32\cdot 9 - 163 + 7, 32) = 4.
\end{aligned}$$

Выполним первую итерацию алгоритма

1^1. $p = 9 > q = 5$.

а) $U^{(p)} = U^{(9)} = 1^{32} \neq 0$. Для каждого t_{r_i}, вычислим

i	t_{r_i}	$U^{(p_0)}$	$U^{(p_1)}$	$U^{(3)}$	$U^{(4)}$	$U^{(5)}$
0				1^{32}	1^{32}	1^{32}
1	29	$0^{29}1^3$	$1^{29}0^3$	$1^{32} + U^{p_0} = 1^{29}0^3$	$1^{32} + U^{(p_1)} = 0^{29}1^3$	1^{32}
2	0	1^{32}	0^{32}	$1^{29}0^3$	$U^{(4)} + U^{p_0} = 1^{29}0^3$	$U^{(5)} + U^{(p_1)} = 1^{32}$;
3	3	0^31^{29}	1^30^{29}	$1^{29}0^3$	$U^{(4)} + U^{p_0} = 1^30^{26}1^3$	$U^{(5)} + U^{(p_1)} = 0^31^{29}$
4	4	0^41^{28}	1^40^{28}	$1^{29}0^3$	$U^{(4)} + U^{p_0} = 1^30^11^{25}0^3$	$U^{(5)} + U^{(p_1)} = 1^30^11^{25}0^3$

б) $U^{(9)} = 0,\ p = p - 1 = 9 - 1 = 8.$

$$\begin{aligned}
l_{r_1} &= 3 - 1 = 2;\\
l_{r_2} &= 4 - 1 = 3;\\
l_{r_3} &= 4 - 1 = 3;\\
l_{r_4} &= 4 - 1 = 3.
\end{aligned}$$

Выполняя итерации $1^2, 1^3, 1^4$, аналогичные первой итерации, получим

$U^{(0)}$	$U^{(1)}$	$U^{(2)}$	$U^{(3)}$	$U^{(4)}$	l_{r_1}	l_{r_2}	l_{r_3}	l_{r_4}
		$1^{29}0^3$	$1^30^11^{28}$	$1^{29}0^3$	1	2	2	2
	$1^{29}0^3$	$1^30^11^{28}$	1^{32}		0	1	1	1
$1^{29}0^3$	$1^30^11^{28}$	1^{32}			-1	0	0	0

Осталось выполнить пятую итерацию (действие 1^5 не выполняется)

2^5. $p = q = 5$, $\deg U^{(p)} = 31 > \deg V^{(p)} = 2$.

Из вектора $U^{(p)} = U^{(5)} = U^{(5)}$ «обнулением» трех младших разрядов образуем вектор $W = 0^41^{28}$. Выполним вычисления для каждого $t_{r_i},\ i = 1, 2, 3, 4$.

i	t_{r_i}	$U^{(p_0)}$	$U^{(p_1)}$	$U^{(0)}$	$U^{(1)}$
1	29	0^{32}	$0^11^{28}0^3$	$U^{(0)} + U^{(p_1)} = 1^10^{31}$;	
2	0	0^41^{28}	0^{32}	$U^{(0)} + U^{(p_0)} = 1^10^31^{28}$	$U^{(1)} + U^{p_1} = 1^30^11^{28}$;
3	3	0^71^{25}	1^30^{29}	$U^{(0)} + U^{(p_0)} = 1^10^31^30^{25}$	$U^{(1)} + U^{p_1} = 0^40^11^{28}$;
4	4	0^81^{24}	1^40^{28}	$U^{(0)} + U^{(p_0)} = 1^10^31^30^11^{24}$	$U^{(1)} + U^{p_1} = 1^{32}$.

После "обнуления"$32-3 = 29$ старших разрядов вектора $U^{(5)}$ получаем окончательный результат rem $(p_1(x), p_2(x))$ приведения многочлена $p_1(x)$ по модулю $p_2(x)$:

$U^{(0)}$	$U^{(1)}$	$U^{(2)}$	$U^{(3)}$	$U^{(4)}$	$U^{(5)}$
$1^10^31^30^11^{24}$	1^{32}	1^{32}	1^{32}	$1^{29}0^3$	1^30^{29}

Вариант В.При реализации алгоритма приведения с использованием виртуальных машинных слов длины $\frac{32(q+1)}{2} = \frac{32\cdot 6}{2} = 96$ параллельно будут выполняться 96 последовательных итераций «школьного» алгоритма.

По построению,

$$\tilde{s} = \frac{32 \cdot 5 + 1}{2} = 96,\ \tilde{p} = \deg_{\tilde{s}} p_1(x) = 3,\ \tilde{q} = \deg_{\tilde{s}} p_2(x) = 1.$$

Список $L = [r_1, r_2, r_3, r_4] = [0, 3, 6, 7]$.

Многочлен-делимое представляется четырьмя виртуальными словами:

$\tilde{U}^{(0)} = U^{(0)}U^{(1)}U^{(2)}$	$\tilde{U}^{(1)} = U^{(3)}U^{(4)}U^{(5)}$	$\tilde{U}^{(2)} = U^{(6)}U^{(7)}U^{(8)}$	$\tilde{U}^{(3)} = U^{(9)}$
1^{96}	1^{96}	1^{96}	1^{32}

Вычислим

$$\tilde{l}_{r_1} = \lfloor \tfrac{96\cdot 3-163+0}{96} \rfloor = 1;$$
$$\tilde{l}_{r_2} = \lfloor \tfrac{96\cdot 3-163+3}{96} \rfloor = 1;$$
$$\tilde{l}_{r_3} = \lfloor \tfrac{96\cdot 3-163+6}{96} \rfloor = 1;$$
$$\tilde{l}_{r_4} = \lfloor \tfrac{96\cdot 3-163+7}{96} \rfloor = 1;$$
$$\tilde{t}_{r_1} = \mathrm{rem}(96\cdot 3-163+0, 96) = 29;$$
$$\tilde{t}_{r_1} = \mathrm{rem}(96\cdot 3-163+3, 96) = 32;$$
$$\tilde{t}_{r_1} = \mathrm{rem}(96\cdot 3-163+6, 96) = 35;$$
$$\tilde{t}_{r_1} = \mathrm{rem}(96\cdot 3-163+7, 96) = 36.$$

Вычисления первой итерации:

1^1. $\tilde{p} = 3 > \tilde{q} = 1$.

а) $\tilde{U}^{(p)} = \tilde{U}^{(3)} = 1^{32}0^{64} \neq 0$, для каждого $r_i,\ i = 1, 2, 3, 4$, вычислим:

i	t_{r_1}	$\tilde{U}^{(p_0)}$	$\tilde{U}^{(p_1)}$	$\tilde{U}^{(1)}$	$\tilde{U}^{(2)}$
0				1^{96}	$1^{32}(0^{64})$
1	29	$0^{29}1^{32}(0^{35})$	0^{96}	$1^{96} + \tilde{U}^{(p_0)} = 1^{29}0^{32}1^{35}$	$1^{96} + \tilde{U}^{(p_1)} = 1^{96}$
2	32	$0^{32}1^{32}0^{32}$	0^{96}	$U^{(1)} + \tilde{U}^{(p_0)} = 1^{29}0^{3}1^{29}0^{3}1^{32}$	$U^{2} + \tilde{U}^{(p_1)} = 1^{96}$
3	35	$0^{35}1^{32}0^{29}$	0^{95}	$U^{(1)} + \tilde{U}^{(p_0)} = 1^{29}0^{3}1^{3}0^{26}1^{3}0^{3}1^{29}$	$U^{(2)} + \tilde{U}^{(p_1)} = 1^{96}$
4	36	$0^{36}1^{32}0^{28}$	0^{96}	$U^{(1)} + \tilde{U}^{(p_0)} = 1^{29}0^{3}1^{3}0^{1}1^{25}0^{3}1^{3}0^{1}1^{28}$	$U^{(2)} + \tilde{U}^{(p_1)} = 1^{96}$

б) $\tilde{U}^{(2)} = 0,\ p = p - 1 = 3 - 1 = 2.$

$$l_{r_1} = 1 - 1 = 0;$$
$$l_{r_2} = 1 - 1 = 0;$$
$$l_{r_3} = 1 - 1 = 0;$$
$$l_{r_4} = 1 - 1 = 0.$$

Вычисления второй итерации:

1^2. $p = 2 > q = 1$.

а) $\tilde{U}^{(p)} = \tilde{U}^{(2)} = 1^{96} \neq 0$.

i	t_{r_1}	$\tilde{U}^{(p_0)}$	$\tilde{U}^{p_1}$	$\tilde{U}^{(0)}$	$\tilde{U}^{(1)}$
0				$\tilde{U}^{0} = 1^{96}$	$\tilde{U}^{(1)} = 1^{29}0^{3}1^{3}0^{1}1^{25}0^{3}1^{3}0^{1}1^{28}$
1	29	$0^{29}1^{67}$	$1^{29}0^{67}$	$1^{96} + \tilde{U}^{(p_0)} = 1^{29}0^{67}$	$\tilde{U}^{(1)} + \tilde{U}^{p_1} = 0^{32}1^{3}0^{1}1^{25}0^{3}1^{3}0^{1}1^{28}$
2	32	$0^{32}1^{64}$	$1^{32}0^{64}$	$\tilde{U}^{(0)} + \tilde{U}^{(p_0)} = 1^{29}0^{3}1^{64}$	$\tilde{U}^{(1)} + \tilde{U}^{p_1} = 1^{35}0^{1}1^{295}0^{3}1^{3}0^{1}1^{28}$
3	35	$0^{35}1^{61}$	$1^{35}0^{61}$	$\tilde{U}^{(0)} + \tilde{U}^{(p_0)} = 1^{29}0^{3}1^{3}0^{61}$	$\tilde{U}^{(1)} + \tilde{U}^{p_1} = 0^{36}1^{25}0^{3}1^{3}0^{1}1^{28}$
4	36	$0^{36}1^{60}$	$1^{36}0^{60}$	$\tilde{U}^{(0)} + \tilde{U}^{(p_0)} = 1^{29}0^{3}1^{3}0^{1}1^{60}$	$\tilde{U}^{(1)} + \tilde{U}^{p_1} = 1^{61}0^{3}1^{3}0^{1}1^{28}$

б) $U^{(2)} = 0,\ p = p - 1 = 2 - 1 = 1.$

$$l_{r_1} = 0 - 1 = -1;$$
$$l_{r_2} = 0 - 1 = -1;$$
$$l_{r_3} = 0 - 1 = -1;$$
$$l_{r_4} = 0 - 1 = -1.$$

Вычисления третьей итерации (действие 1^3 не выполняется):
2^3. $p = 1 = q = 1$, $\deg \tilde{U}^{(1)} = 95 > \deg \tilde{V}^{(1)} = 67$.

Из вектора $\tilde{U}^{p_1} = 1^{61}0^31^30^11^{28}$ «обнулением» 67 младших разрядов образуем вектор $W = 0^{68}1^{28}$. Выполним вычисления для каждого $t_{r_1},\ i = 1, 2, 3, 4$.

i	t_{r_i}	$\tilde{U}^{(p_0)}$	$\tilde{U}^{(p_1)}$	$\tilde{U}^{(0)}$
0				$\tilde{U}^{(p_0)} = 1^{29}0^31^30^11^{60}$
1	29	0^{96}	$0^11^{28}0^{67}$	$\tilde{U}^{(0)} + \tilde{U}^{(p_1)} = 1^10^{31}1^30^11^{60}$
2	32	0^{96}	$0^41^{28}0^{64}$	$\tilde{U}^{(0)} + \tilde{U}^{(p_1)} = 1^10^31^{31}0^11^{60}$
3	35	0^{96}	$0^71^{28}0^{61}$	$\tilde{U}^{(0)} + \tilde{U}^{(p_1)} = 1^10^31^30^{29}1^{60}$
4	36	0^{96}	$0^81^{28}0^{60}$	$\tilde{U}^{(0)} + \tilde{U}^{(p_1)} = 1^10^31^30^11^{88}$

После "обнуления"$96 - 67 = 29$ старших разрядов вектора $\tilde{U}^{(1)} = 1^{61}0^31^30^11^{28}$ получаем окончательный результат rem $(p_1(x), p_2(x))$ приведения многочлена $p_1(x)$ по модулю $p_2(x)$:

$\tilde{U}^{(0)} = U^{(0)}U^{(1)}U^{(2)}$	$\tilde{U}^{(1)} = U^{(3)}U^{(4)}U^{(5)}$
$1^10^31^30^11^{88} = 1^10^31^30^11^{24}\|\|1^{32}\|\|1^{32}$	$1^{61}0^31^30^{29} = 1^{32}\|\|1^{29}0^{(3)}\|\|1^30^{29}$

Для «малочлена» более общего случая (допускается один ненулевой коэффициент среди $v_{\deg\ p_2-1}, \ldots, v_{\deg\ p_2-s+1}$) при выполнении алгоритма вместо вектора $U^{(p)}$ необходимо использовать целую часть $Z^{(p)}$ от деления многочлена $U^{(p)}(x) \cdot x^{\deg\ p_2-t}$ (при $p > q$) или $U^{(p)\prime}(x) \cdot x^{\deg\ p_2-t}$ (при $p = q$) на двучлен $x^{\deg\ p_2-t} + 1$, где $t = t_{r_l}$ степень второго после старшего ненулевого слагаемого «малочлена», $U^{(p)}(x)$ – многочлен с последовательностью коэффициентов $U^{(p)}$, $U^{(p)\prime}(x)$ получается из $U^{(p)}(x)$ обнулением $\deg V^{(p)}$ младших коэффициентов. В частности, при $t = \deg p_2 - 1$ используется двучлен $1 + x$.

Примечание. Если степень многочлена-делимого $p_1(x)$ не превышает $2 \deg p_2(x) - 2$, то единственный ненулевой коэффициент допускается среди коэффициентов $v_{\deg\ p_2(x)-1}, \ldots, v_{\deg\ p_2-s+3}$.

Пример 3.4.4 Приведем многочлен степени 30, заданный вектором коэффициентов (в прядке возрастания степеней переменной)

$$U = U^{(0)}U^{(1)}U^{(2)}U^{(3)} = (00101000\ 10101000\ 10001010\ 10001010),$$

по модулю многочлена

$$1 + x^{21} + x^{22} = (10000000\ 00000000\ 0000011).$$

Пусть $s = 8$, тогда $p = 3$, $q = 2$, список $L = [r_1, r_2] = [0, 21]$. Вычислим

$$l_{r_1} = \lfloor \tfrac{8\cdot 3-22+0}{8} \rfloor = 0,$$
$$l_{r_2} = \lfloor \tfrac{8\cdot 3-22+21}{8} \rfloor = 2,$$
$$t_{r_1} = \mathrm{rem}(8 \cdot 3 - 22 + 0, 8) = 2,$$
$$t_{r_1} = \mathrm{rem}(8 \cdot 3 - 22 + 21, 8) = 7$$

Вычисления первой итерации алгоритма:

1^1. $p = 3 > q = 2$. Вычислим частное от деления многочлена $U^{(3)}x^{22-21} = U^{(3)}x =$ (01000101) на двучлен $1 + x$, получим

$$Z^{(3)} = (10000110).$$

Далее получим а) Для каждого $t_{r_i},\ i = 1, 2$

i 0	t_{r_i}	$U^{(p_0)}$	U^{p_1}	$U^{(0)} =$ (00101000)	$U^{(1)} =$ (10101000)	$U^{(2)} =$ (10001010)
1	2	(00100001)	(10000000)	$U^{(0)} + U^{(p_0)} =$ (00001001)	$U^{(1)} + U^{(p_1)} =$ (00101000)	
2	7	(00000001)	(00001100)			$U^{(2)} + U^{(p_0)} =$ (10001011)

После первой итерации получаем частичный остаток

$$(000010010010100010001011).$$

б)Модифицируем значения $l_{r_1} = 0 - 1 = -1,\ l_{r_2} = 2 - 1 = 1,\ p = 3 - 1 = 2$.

Вычисления второй итерации:

1^2. $p = q = 2$, $\deg U^{(2)} = 7 > \deg V^{(2)} = 6$. Образуем многочлен $W = (00000011)$ обнулением $\deg V^{(2)} = 6$ младших разрядов вектора $U^{(2)} = (10001011)$. Вычислим частное $Z^{(2)}$ от деления многочлена $Wx = (000000011)$ на двучлен $1 + x$, получим

$$Z^{(2)} = (00000001).$$

Далее выполним вычисления для $t_{r_i},\ i = 1, 2:$

i 0	t_{r_1}	$U^{(p_0)}$	U^{p_1}	$U^{(0)} =$ (00001001)	$U^{(1)} =$ (00101000)	$U^{(2)} =$ (10001011)
1	2	(00000000)	(01000000)	$U^{(0)} + U^{(p_1)} =$ (01001001)		
2	7	(00000000)	(00000010)		$U^{(1)} + U^{(p_0)} =$ (00101000)	$U^{(2)} + U^{(p_1)} =$ (10001011)

После второй итерации и обнуления 8-6= 2 старших разрядов получаем остаток – результат модулярного приведения:

$$(01001001\ 00101000\ 10001000).$$

Пример 3.4.5 Приведем многочлен степени 22, заданный вектором коэффициентов (в прядке возрастания степеней переменной)

$$(00101000\ 10101000\ 10001010),$$

по модулю многочлена

$$1 + x^{11} + x^{15} = (10000000\ 00010001).$$

Пусть $s = 8$, тогда $p = 2$, $q = 1$, список $L = [r_1, r_2] = [0, 11]$. Вычислим

$$\begin{gathered} l_{r_1} = \lfloor \tfrac{8\cdot 2-15+0}{8} \rfloor = 0, \\ l_{r_2} = \lfloor \tfrac{8\cdot 2-15+11}{8} \rfloor = 1, \\ t_{r_1} = \mathrm{rem}(8 \cdot 2 - 15 + 0, 8) = 1, \\ t_{r_1} = \mathrm{rem}(8 \cdot 2 - 15 + 11, 8) = 4 \end{gathered}$$

вычисления первой итерации:

1^1. $p = 2 > q = 1$. Вычислим частное $Z^{(2)}$ от деления многочлена $U^{(2)}x^{15-11} = U^{(2)}x^4 =$ (000010001010) на двучлен $1 + x^4$, получим

(00101010).

Далее вычислим для $t_{r_i},\ i = 1, 2$:

i	t_{r_i}	$U^{(p_0)}$	$U^{(p_1)}$	$U^{(0)} =$ (00101000)	$U^{(1)} =$ (10101000)
1	1	(00010101)	(00000000)	$U^{(0)} + U^{(p_0)} =$ (00111101)	$U^{(1)} + U^{(p_1)} =$ (10101000)
2	4	(00000010)	(10100000)		$U^{(1)} + U^{(p_0)} =$ (10101010)

После первой итерации получаем частичный остаток

(00111101 10101010).

Модифицируем значения $l_{r_1} = 0 - 1 = -1,\ l_{r_2} = 2 - 1 = 0,\ p = 2 - 1 = 1.$

Вторая итерация (действие 1^2 не выполняется):

2^2. $p = q = 1$, $\deg U^{(2)} = 6 < \deg V^{(2)} = 7$. После второй итерации имеем результат приведения:

(00111101 10101010).

Данный вариант алгоритма можно ускорить, вычислив заранее, бинарную таблицу T содержащую 2^{16} строк, определяемых двухбайтовыми двоичными наборами

$$(a_0, ..., a_7, a_8, ..., a_{15})$$

коэффициентов многочлена

$$\sum_{i=0}^{i=15} a_i x^i$$

и образуемых коэффициентами

$$(b_0, ..., b_7, b_8, ..., b_{14}, b_{15})$$

частного от деления многочлена

$$x \cdot \sum_{i=0}^{i=15} a_i x^i$$

на двучлен $1 + x$:

$$T(a_0, ..., a_7, a_8, ..., a_{15}) = (b_0, ..., b_7, b_8, ..., b_{14}, b_{15}).$$

Тогда набор коэффициентов

$$(b_0^0, \ldots, b_{15}^0, b_0^1, \ldots, b_{15}^1, \ldots, b_0^{t-1}, \ldots, b_{15}^{t-1})$$

частного от деления многочлена

$$x \cdot \sum_{j=0}^{t-1} x^{16j} (\sum_{i=0}^{i=15} a_i^j x^i),$$

определяемого набором двоичных коэффициентов

$$(a_0^0, \ldots, a_{15}^0, a_0^1, \ldots, a_{15}^1, \ldots, a_0^{t-1}, \ldots, a_{15}^{t-1}),$$

на двучлен $1 = x$ можно вычислить по следующему алгоритму:

1. Принять $(b_0^{t-1}, \ldots, b_{15}^{t-1}) = T(a_0^{t-1}, \ldots, a_{15}^{t-1})$;
2. Для $j = t-2, \ldots, 0$ выполнить
   если $b_0^{j+1} = 0$, то $(b_0^j, \ldots, b_{15}^j) = T(a_0^j, \ldots, a_{15}^j)$;
   иначе $(b_0^j, \ldots, b_{15}^j) = \bar{T}(a_0^j, \ldots, a_{15}^j)$.

Здесь и ниже $\bar{T}(\ldots)$ – вектор, получаемый логическим отрицанием каждого разряда вектора T.

Пример 3.4.6 а) Пусть $t = 2$,

$$(a_0^0, a_1^0 ..., a_{15}^0) = (1010101010101010),$$
$$(a_0^1, a_1^1 ..., a_{15}^1) = (0111111111111111).$$

Тогда

$$T(a_0^1, a_1 1^1 ..., a_{15}^1) = T(0111111111111111) = (1101010101010101),$$
$$T(a_1^0, a_1 1^0 ..., a_{15}^0) = T(1010101010101010) = (0110011001100110).$$

искомое частное описывается вектором

$$(10011001100110011101010101010101)$$

б) Если при тех же условиях

$$(a_0^1, a_1^1 ..., a_{15}^1) = (1111111111111111),$$

то

$$T(a_0^1, a_1^1 ..., a_{15}^1) = T(1111111111111111) = (0101010101010101),$$

и искомое частное описывается вектором («младшая» половина не инвертируется)

$$(01100110011001100101010101010101)$$

Рассмотренный метод деления многочлена на двучлен $1 + x$ позволяет построить эффективный алгоритм приведения многочлена степени не более $2n - 2$ по модулю трехчлена

$$1 + x^{n-1} + x^n,$$

для использования в алгоритме умножения в поле $GF(2^n)$ в полиномиальном базисе[1].

[1]Если такой многочлен порождает нормальный базис, то предлагаемый алгоритм рекомендуется использовать в комбинированных алгоритмах, когда умножение осуществляется в полиномиальном базисе, а возведение в степень характеристики поля – в нормальном с переходами из одного базиса в другой по мере необходимости.

Пусть s – длина машинного слова, $p_1(x)$ – полином делимое степени не выше $2n-2$, $p_2(x)$ – трехчлен-делитель степени n указанного вида, q $\deg_s p_2$. Объединим машинные слова длины s в виртуальные машинные слова длины $\tilde{s} = s \cdot (q+1)$.

Теперь полином-делимое представляется двумя виртуальными машинными словами

$$U = \tilde{U}^{(0)} || \tilde{U}^{(1)},$$

а полином-делитель – составляет одно виртуальное машинное слово $V = \tilde{V}^{(0)}$. Обозначим $\deg_{\tilde{s}} p_1 = \tilde{p}$, $\deg_{\tilde{s}} p_2 = \tilde{q}$. В данном случае $\tilde{p} = 1$, $\tilde{q} = 0$. В списке L имеем элементы O

$$\tilde{l}_{r_1} = \lfloor \frac{\tilde{p}\tilde{s} - \deg\, p_2}{\tilde{s}} \rfloor = 0$$

$$\tilde{l}_{r_2} = \lfloor \frac{\tilde{p}\tilde{s} - \deg\, p_2 + \deg\, p_2 - 1}{\tilde{s}} \rfloor = 0$$

в списке $\tilde{L}$ и соответствующие элементы

$$t_{r_1} = t_0 = \tilde{s} - \deg\, p_2.$$

$$t_{r_1} = t_{\deg\, p_2 - 1} = \mathrm{rem}(\tilde{p}\tilde{s} - \deg\, p_2 + \deg\, p_2 - 1, \tilde{s}) = \tilde{s} - 1.$$

Алгоритм приведения по модулю трехчлена $p_2(x)$ степени n указанного вида можно описать следующим образом.

1. Если $\tilde{U}^{(1)} \neq 0$ вычислить вектор $\tilde{S}$ коэффициентов частного $s(x)$ от деления многочлена $x \cdot \tilde{U}^{(1)}$ на двучлен $1+x$ (при этом следует применить описанный выше использующий заранее приготовленную таблицу T алгоритм). Младший разряд вектора $\tilde{S}$ прибавить к старшему разряду вектора $\tilde{U}^{(0)}$.

Образовать из вектора $\tilde{S}$ вектор $\tilde{U}^{(p_0)}$ сдвигом $\tilde{S}$ на t_0 разрядов в сторону старших разрядов. Принять

$$\tilde{U}^{(0)} = \tilde{U}^{(0)} + \tilde{U}^{(p_0)}{}^{2},$$

2. Найти остаток от деления многочлена $U = \tilde{U}^{(0)}$ на многочлен V.

Упражнение 3.4.1 Докажите, что описанный алгоритм соответствует описанному выше алгоритму деления на многочлен с единственным ненулевым коэффициентом, кроме старшего, степени более $\deg\, p_2(x) - s - 1$.

Указание. Прибавлением младшего разряда вектора $\tilde{s}$ к старшему не изменяемому в данной итерации разряду учитывает действие второго ненулевого коэффициента.

[2] В итоге $\tilde{s} - t_0$ младших разрядов вектора $\tilde{S}$ будут прибавлены к $\tilde{s} - t_0$ старшим разрядам вектора $\tilde{U}^{(0)}$.

Пример 3.4.7 Приведем многочлен x^{14892} в полиномиальном базисе по модулю неприводимого трехчлена

$$1 + x^{7446} + x^{7447}.$$

(порождающего и нормальный базис).

В данном случае $q = \lfloor \frac{7447}{32} \rfloor = 232,\ \tilde{s} = s(q+1) = 7456$

$$\tilde{U}^{(0)} = 0,\ \tilde{U}^{(1)} = x^{14892-7456} = x^{7436}.$$

Вычислим $t_{r_1} = \tilde{s} - \deg p_2 = 7456 - 7447 = 9$.

1.$\tilde{U}^{(1)} \neq 0$, вычислим вектор S коэффициентов частного $s(x)$ от деления многочлена $x \cdot x^{7436} = x^{7437}$ на двучлен $1 + x$, получим $S = (1)^{7437}$ (вектор из 7437 единиц и старших 19 нулевых разрядов). Прибавим единицу младшего разряда вектора $\tilde{S}$ к старшему разряду вектора $\tilde{U}^{(0)}$:

$$\tilde{U}^{(0)} = 0^{7456} + 0^{(7455)}1^1 = 0^{(7455)}1^1$$

Из вектора $\tilde{S}$ получим вектор

$$\tilde{U}^{(p_0)} = 0^9 1^{7437} 0^{10},$$

Далее получим

$$\tilde{U}^{(0)} = \tilde{U}^{(0)} + \tilde{U}^{(p_0)} = 0^{7455}1^1 + 0^9 1^{7437} 0^{10} = 0^9 1^{7437} 0^9 1^1,$$

2. Найдем остаток от деления многочлена $U = \tilde{U}^{(0)}$ на многочлен V.

В данном случае $s = 32$, $p = q = \lceil \frac{7447}{32} \rceil = \lfloor \frac{7455}{32} \rfloor = 233$, Составим список $L = [0, 7446]$. Имеем

$$l_0 = \lfloor \frac{7456 - 7447}{32} \rfloor = 0,$$

$$l_{7446} = \lfloor \frac{7456 - 7447 + 7446}{32} \rfloor = 232.$$

Вычислим

$$t_0 = \mathrm{rem}(32 \cdot 233 - 7447, 32) = 9,$$

$$t_{7445} = \mathrm{rem}(32 \cdot 2323 - 7447 + 7446, 32) = 31,$$

Выполним единственную итерацию базового алгоритма. П.1 не выполняется, так как $p = q$.

Выполним п. 2. $\deg U^{233} = 7455 > \deg V^{233} = 23$.

Образуем вектор W как частное от деления многочлена $x \cdot U^{233} = 0^1 1^{22} 0^0 1^1$ на двучлен $1 + x$. Подучим

$$W = 0^1 1^1 0^1 1^1 0^1 1^1 0^1 1^1 0^1 1^1 0^1 1^1 0^1 1^1 0^1 1^1 0^1 1^1 0^1 1^1 0^1 1^1 0^1 1^1 0^1 1^1 0^1 1^9.$$

Обнулив 23 младших разряда вектора W, получим вектор $W = 0^{23}1^9$.

Остальные вычисления представлены в таблице (где показаны только изменяемые части вектора U).

i	l_i	t_{r_i}	$U^{(p_0)}$	$U^{(p_1)}$	$U^{(0)}$	$U^{(233)}$
					$0^9 1^{23}$	$1^{22}0^9 1^1$
1	0	0	$0^9 1^{23}$	$1^9 0^{23}$	$U^{(0)} + U^{p_1} = 1^{32}$	
2	231	31	0^{32}	$0^{22}1^9 0^1$		$1^3 2$

Установим в нулевое состояние разряды с $\deg V^{(232)} = 23$ по 31 (младший разряд имеет номер 0) полученного вектора $U(233)$, получим $\tilde{U}^{(233)} = 1^{23}0^9$.

Вернем вектор $U = 1^{7447}$.

Описанные в этом параграфе алгоритмы обеспечивают приведение по модулю модулярного «малочлена» за время, составляющее малую (около 10 процентов) долю времени умножения.

Эти алгоритмы соответствуют принципам ускорения деления многочленов путем организации параллельного исполнения однотипных операций, описанным в работе [?], они опубликованы авторами настоящей монографии в [11, 12].

В заключение опишем кратко в качестве примера алгоритм приведения многочлена $c(x)$ степени не больше 324 по модулю $f(x) = x^{163} + x^7 + x^6 + x^3 + 1$. Этот пример заимствован из [107]. Этот пример показывает, что подобные алгоритмы были описаны независимо разными авторами, что подтверждает их актуальность. Коэффициенты $c(x)$ представляются в виде массива $C[10], \ldots, C[0]$ 32-битных целых чисел; результат $c(x) \bmod f(x)$ представляется в виде массива $C[5], C[6], \ldots, C[0]$. Эти массивы мы также будем представлять в виде битовых массивов $c[324], \ldots, c[0]$ и $c[162], \ldots, c[0]$. Обработка числа $C[9]$, например, соответствует выполнению преобразования многочлена $c(x)$ согласно равенствам:

$$x^{319} = x^{163} + x^{162} + x^{159} + x^{156} \bmod f(x)$$

$$x^{318} = x^{162} + x^{161} + x^{158} + x^{155} \bmod f(x)$$

и т.д. вплоть до

$$x^{288} = x^{132} + x^{131} + x^{128} + x^{126} \bmod f(x).$$

Преобразования заключаются в замене одночленов из левых частей равенств на четырехчлены из правых частей. Эти преобразования соотвепствуют четырехкратному прибавлению к битовому массиву C коэффициентов многочлена $c(x)$ 32-битного вектора $C[9]$, причем первый раз он прибавляется так, что его младший бит складывается с с 132 битом массива C, следующий бит складывается со 133 битом массива C и так далее, во второй раз он прибавляется к массиву C, начиная со 131 бита, в третий раз — начиная со 128 бита, и в четвертый раз — начиная со 125 бита. Так как мы работаем не с битами, а с 32 битными целыми числами, то первая процедура прибавления к битовому массиву C сдвинутого (к 132-му биту) вектора $C[9]$ выполняется последовательностью операций:

$$(c[128], \ldots, c[159]) = C[4] \leftarrow C[4] \oplus (C[9] << 4),$$

(операция $C[9] << 4$ сдвига влево на 4 бита превращает вектор $(c[288], \ldots c[319])$ в вектор $(0, 0, 0, 0, c[288], \ldots c[315])$ потому что реально в числе $C[9]$ младший бит $c[288]$ стоит на самом правом месте)

$$(c[160], \ldots, c[191]) = C[5] \leftarrow C[4] \oplus (C[9] >> 28),$$

(операция $C[9] >> 28$ сдвига вправо на 28 битов превращает вектор $(c[288], \ldots c[319])$ в вектор $(c[316], c[317], c[318], c[319], 0, \ldots, 0)$). Аналогично вторая процедура прибавления к битовому массиву C сдвинутого (к 131-му биту) вектора $C[9]$ выполняется последовательностью операций:

$$(c[128], \ldots, c[159]) = C[4] \leftarrow C[4] \oplus (C[9] << 3),$$

$$(c[160], \ldots, c[191]) = C[5] \leftarrow C[4] \oplus (C[9] >> 29).$$

Третья процедура прибавления к битовому массиву C сдвинутого (к 128-му биту) вектора $C[9]$ выполняется операций более просто:

$$(c[128], \ldots, c[159]) = C[4] \leftarrow C[4] \oplus C[9].$$

Последняя (четвертая) операция прибавления к битовому массиву C сдвинутого (к 125-му биту) вектора $C[9]$ выполняется последовательностью выполняется последовательностью операций:

$$(c[96], \ldots, c[127]) = C[3] \leftarrow C[3] \oplus (C[9] << 29),$$

$$(c[128],\ldots,c[159]) = C[4] \leftarrow C[4] \oplus (C[9] >> 3).$$

Поэтому вся программа редукции многочлена $c(x)$ по модулю $f(x)$ состоит из цикла, в котором при уменьшении i от 10 до 6 выполняется последовательность операций присваивания $T \leftarrow C[i]$, $C[i-6] \leftarrow C[i-6] \oplus (T << 29)$, $C[i-5] \leftarrow C[i-5] \oplus (T << 4) \oplus (T << 3) \oplus T \oplus (T >> 3)$, $C[i-4] \leftarrow C[i-4] \oplus (T >> 28) \oplus (T >> 29)$, потом выполняется присваивания $T = C[5]AND0xFFFFFFF8$ (очистка битов $0,1,2$ $C[5]$),

$$C[0] \leftarrow C[0] \oplus (T << 4) \oplus (T << 3) \oplus T \oplus (T >> 3),$$

$C[1] \leftarrow C[1] \oplus (T >> 28) \oplus (T >> 29)$, $C[5] = C[5]AND0x00000007$ (очистка остальных битов $C[5]$), и возвращается результат в виде массива $C[5], C[6], \ldots, C[0]$.

3.5 Умножение многочленов с использованием ДПФ

3.5.1 Преобразование Фурье над конечным полем

Рассмотрим многочлен

$$f(X) = \sum_{i=0}^{N-1} a_i X^i$$

над полем $\mathcal{F} = (\{x_0, x_1, \ldots, x_{N-1}), +, \times\}$ конечного порядка $N+1$, определяемый набором

$$(a_0, a_1, \ldots, a_{N-1}) \in \mathcal{F}^{\mathcal{N}} \tag{3.8}$$

коэффициентов – *сигнальным вектором.*

Набор

$$f(x_0), f(x_1), \ldots, f(x_{N-1}) \in \mathcal{F}^{\mathcal{N}} \tag{3.9}$$

значений этого многочлена в точках x_i, $i = 0, 1, \ldots, N-1$ называется *спектральным вектором.*

Операция $D : \mathcal{F}^N \to \mathcal{F}^N$ преобразования сигнального вектора (3.8) в спектральный вектор (3.10) называется *дискретным преобразованием Фурье над полем* $\mathcal{F}$ (ДПФ).

Обратная операция D^{-1} называется *обратным дискретным преобразованием Фурье.* Эта операция восстанавливает сигнальный вектор (3.8) по спектральному вектору (3.10).

Оба преобразования осуществляются в поле $\mathcal{F}$ с использованием корня N-й степени из единицы – образующего элемента λ мультипликативной группы $\mathcal{F}^*$ поля $\mathcal{F}$: коэффициенты $f(x_i)$ получаются по формуле

$$f(x_i) = \sum_{k=0}^{N-1} a_k \lambda^{ki}, \tag{3.10}$$

а элементы a_k сигнального вектора восстанавливаются по спектральному вектору следующим образом:

$$a_k = \frac{1}{N} \sum_{i=0}^{N-1} f(x_i) \lambda^{-ki}, \tag{3.11}$$

При выполнении ДПФ, как и обратного ДПФ по этим формулам осуществляется по n^2 умножений в поле $\mathcal{F}$, что и определяет сложность этих операций (сложение выполняются несравнимо быстрее умножения, а все степени корня λ можно вычислить при инициализации программы.

Сверткой сигнальных векторов

$$(a_0^1, a_1^1, \ldots, a_{N-1}^1) \in \mathcal{F}^{\mathcal{N}},$$
$$(a_0^2, a_1^2, \ldots, a_{N-1}^2) \in \mathcal{F}^{\mathcal{N}} \tag{3.12}$$

называется вектор

$$(c_0, c_1, \ldots, c_{2N-2}) \in \mathcal{F}^{(2N-1)},\ c_t = \sum_{i+j=t} c_i^1 \times c_j^2. \tag{3.13}$$

Не трудно видеть, что свертка (3.13) есть вектор коэффициентов произведения

$$f^1(X) \cdot f^2(X),$$

многочленов $f^1(X)$ и $f^2(X)$, коэффициенты которых определяется векторами (3.12).

Применением ДПФ к свертке (3.13) получается спектральный вектор свертки

$$(d_0, d_1, \ldots, d_{2N-2}) \in \mathcal{F}^{(2N-1)}. \tag{3.14}$$

Элементы последнего могут быть получены также как произведения

$$d_t = d_t^1 \times d_t^2, t = 0, \ldots, 2N-2 \tag{3.15}$$

в поле $\mathcal{F}$ элементов спектральных векторов

$$(d_0^1, d_1^1, \ldots, d_{2N-2}^1) \in \mathcal{F}^{\mathcal{N}},$$
$$(d_0^2, d_1^2, \ldots, d_{2N-2}^2) \in \mathcal{F}^{\mathcal{N}}, \tag{3.16}$$

соответствующих сигнальным векторам (3.12), дополненным N "старшими" нулевыми разрядами

$$(a_0^1, a_1^1, \ldots, a_{N-1}^1, 0, \ldots, 0) \in F^{(2N-1)},$$
$$(a_0^2, a_1^2, \ldots, a_{N-1}^2, 0, \ldots, 0) \in F^{(2N-1)}.$$

Отсюда следует, что произведение[3] двух многочленов $f^1(X)$ и $f^2(x)$ степени не более $N-1$ над полем $\mathcal{F}$ порядка $N+1$ можно получить следующей суперпозицией операций над многочленами

$$f^1(X) \cdot f^2(x) = \frac{1}{N} D^{-1}(D(f^1(X)) \times D(f^1(X))), \tag{3.17}$$

где $D(f^i(X))$ — спектральные векторы длины $2N-1$, N младших элементов которых суть коэффициенты многочленов $f^i(X)$, а остальные элементы – нулевые, $\times$ – операция умножения в поле $\mathcal{F}$. Как видим, сложность вычисления произведения этим методом определяется сложностью дискретного преобразования Фурье ($O(N^2)$, если оно осуществляется по (3.10)), поскольку сложность покомпонентное умножения линейна). Если порядок N мультипликативной группы $\mathcal{F}^*$ поля $\mathcal{F}$ не является простым числом, то появляется возможность ускорения ДПФ.

3.5.2 Быстрый метод вычисления ДПФ над конечным полем

Для ускорения ДПФ в случае составного порядка N мультипликативной группы $\mathcal{F}^*$ мы используем способ ускорения ДПФ над комплексным полем, описанный в [?] для ускорения ДПФ, когда сигнальный вектор представляет собой набор значений действительной

[3] То есть вектор коэффициентов произведения

функции $f(X)$, в конечном числе точек $x_i = \frac{1}{N}$, $i = 0, 1, \ldots, N-1$, спектральный вектор — набор коэффициентов аппроксимирующего эту функцию многочлена и вместо образующего элемента конечного поля λ используется корень N-ой степени из единицы $\omega = \exp\frac{2\pi i}{N}$.

Пусть порядок мультипликативной группы $\mathcal{F}^*$ разложим: $N = N_1 \cdot N_2$, где N_2 есть простое число, $N_1 > 2$.

Представим в формуле (3.10) индексы i в виде

$$i = i_1 N_1 + i_0,\ i = 0, 1, \ldots N-1,$$

где $0 \le i_0 < N_1$, а индексы k – в виде

$$k = k_1 N_2 + k_0,\ i = 0, 1, \ldots N-1,$$

где $0 \le k_2 < N_2$,

Тогда степени ki представляются формулой

$$ki = (k_1 N_2 + k_0)(i_1 N_1 + i_0) = k_1 i_1 N + k_1 i_0 N_2 + k_0 i_1 N_1 + k_0 i_0 =$$
$$= k_1 i_1 N + k_1 i_0 N_2 + k_0 i.$$

и формула (3.10) приобретает вид

$$f(x_i) = \sum_{k=0}^{N-1} a_k \lambda^{k_1 i_1 N + k_1 i_0 N_2 + k_0 i_1} = \sum_{k=0}^{N-1} a_k \lambda^{k_1 i_0 N_2 + k_0 i_1}, \tag{3.18}$$

(Слагаемое $k_1 i_1 N$ в показателе степени можно опустить, так как λ-примитивный элемент поля.)

Заменяя в (3.18) суммирование по i операцией повторного суммирования по по индексам k_0 и k_1, получаем

$$f(x_i) = \sum_{k_0=0}^{N_2-1} \sum_{k_1=0}^{N_1-1} a_{k_1 N_2 + k_0} \lambda^{k_1 i_0 N_2} \lambda^{k_0 i} = \sum_{k_0=0}^{N_2-1} \tilde{a}(k_0, i_0) \lambda^{k_0 i}, \tag{3.19}$$

где

$$\tilde{a}(k_0, i_0) = \sum_{k_1=0}^{N_1-1} a_{k_1 N_2 + k_0} \lambda^{k_1 i_0 N_2}. \tag{3.20}$$

Массив $\tilde{a}$ содержит N чисел и для его вычисления требуется NN_1 операций умножения в поле $\mathcal{F}$, затем на вычисления по формуле (3.19) потребуется еще NN_2 таких операций. Таким образом, используя разложение (если оно возможно) порядка мультипликативной группы $\mathcal{F}^*$ на два множителя, мы можем осуществить преобразование Фурье, выполнив $N(N_1 + N_2)$ операций умножения в поле $\mathcal{F}$. Этим достигается экономия операций, так как $N_1 + N_2 < N_1 N_2 = N$ и $N > 4$. Если $N = N_1 \cdot N_2 \cdot \ldots N_m,\ 2 \le N_s$, то преобразование Фурье в конечном поле F можно выполнить, затратив $N(N_1 + N_2 + \cdots + N_m)$ операций умножения в этом поле.

Обратим внимание, что применение ДПФ с использованием комплексного поля путем выбора числа $N = 2^n$ значений аппроксимируемой функции f в качестве сигнального вектора позволяет достичь максимального ускорения, понизив число операций умножения до $2N \log_2 N$. В случае применения конечного поля $\mathcal{F}$, большого порядка эта оценка, видимо, недостижима[4].

Описанный подход к ускорению ДПФ получил название *быстрого преобразования Фурье* (БПФ). Его наиболее раннее описание (1866 г.) принадлежит Гауссу , а современное изложение, повлекшее многочисленные применения, содержится в работе Д.Кули и Д.Тьюки []

[4]Порядок конечного поля, имеющего примитивный элемент порядка 2^n, есть число Ферма F(n), например, F(3), но маловероятно, что существуют иные числа Ферма, являющиеся степенью простого числа.

$$N = \underbrace{1 \cdot N_1 \cdot N_2 \cdots N_{i-2} \cdot N_{i-1} \cdot N_i \cdots N_{s-2} \cdot N_{s-1} \cdot N_s}_{N^s = N} \cdot \underbrace{1}_{N_2^s = 1}$$

$$\underbrace{\qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad}_{N^{s-1}} \underbrace{\quad}_{N_2^{s-1}}$$

а)

$$N = \underbrace{1 \cdot N_1 \cdot N_2 \cdots N_{i-1} \cdot N_i}_{N^i} \cdot \underbrace{N_{i+1} \cdots N_s \cdot 1}_{N_2^i}$$

$$\underbrace{\qquad\qquad\qquad}_{N^{i-1}} \underbrace{\qquad\qquad\qquad}_{N_2^{i-1}}$$

б)

$$N = \underbrace{1 \cdot N_1}_{N^1} \cdot \underbrace{N_2 \cdots N_{i-2} \cdot N_{i-1} \cdot N_i \cdots N_s \cdot 1}_{N_2^1}$$

$$\underbrace{\qquad\qquad\qquad\qquad\qquad\qquad}_{N^0 = 1 \qquad\qquad N_2^0 = N}$$

в)

Рис. 3.3: Изменение верхних пределов суммирования N^i в схеме совместной примитивной рекурсии рекурсии и N_2^i в доказательстве этой схемы: а) первый шаг, б) общий шаг, в) заключительный шаг.

3.5.3 Алгоритм БПФ над конечным полем

Для построения алгоритма БПФ над конечным полем покажем, что имеется рекурсивная схема вычисления значений спектрального вектора.

Теорема 3.5.1 *Коэффициенты $f(x_j)$ дискретного преобразования Фурье над полем порядка $N^s = \Pi_{i=1}^{s} N_i + 1$, где N_i – натуральные числа, можно вычислить по следующей схеме примитивной рекурсии* [5]

$$\begin{aligned} N^1(x_j) &= N_1; \\ F^1(x_j) &= \sum_{k=0}^{N_1^1-1} a_k \lambda^{k\cdot j}, \qquad (3.21) \\ N^i(x_j) &= N_1^{i-1} \cdot N_i, \end{aligned}$$

$$F^{i+1}(x_j) = \sum_{k=0}^{N_{i+1}-1} F^i(k) \cdot \lambda^{k\cdot j_0 \cdot N_i},\ j = j_1 N^i + j_0,\ 0 \le j_0 < N^i, \qquad (3.22)$$

как значение $f(x_j) = F^s(x_j)$, получаемое по завершении заключительного шага обратного хода рекурсии.

Обозначим $N_2^i = \frac{N^s}{N^i}$. Справедливость теоремы следует из следующей Леммы.

Лемма 3.5.1 *При любом i, $i = 1, \dots s$, выполняется равенство*

$$f^(x_j) = \sum_{k=0}^{N_2^{i+1}-1} F^i(k) \cdot \lambda^{k\cdot j_0 \cdot N_2^i},\ j = j_1 N^i + j_0,\ 0 \le j_0 < N^i. \qquad (3.23)$$

Доказательство. (Метод математической индукции по числу s шагов рекурсии). Изменение верхних предела N^i суммирования в ходе рекурсии показано на Рис. 3.5.1. Детали доказательства оставляем читателю.

Следствие 3.5.1 *Если порядок N мультипликативной группы поля $\mathcal{F}$ разложим на s натуральных множителей, $N = N_1 \cdot N_2, \cdots N_s$ то каждый коэффициент $f(x_j)$, $j = 0, \dots N^s - 1$, спектрального вектора ДПФ над этим полем по сигнальному вектору можно вычислить, выполнив не более $\sum_{i=1}^{s} N_i$ умножений в этом поле, в предположении, что все степени примитивного элемента λ вычислены заранее. Для вычисления спектрального времени при этих условиях достаточно выполнить не более $N \cdot \sum_{i=1}^{s} N_i$ умножений.*

Примечание. Следующий параграф подлежит существенной переработке в духе данного параграфа и будет еще параграф о результатах экспериментов, материалы имеются!

[5] Обратим внимание, что описание (??) шага рекурсии в общем случае оказывается более полным по сравнению с частным случаем

(3.19), когда первый шаг рекурсии является и заключительным, то есть порядок N разложен на два натуральных множителя.

3.5.4 Преобразование Фурье над полем $GF(2^m)$ и умножение многочленов над полем Галуа $GF(2)$

Рассмотрим особенности применения дискретного преобразования Фурье при умножении многочленов над полем Галуа $GF(2)$.

При этом будем полагать, что многочлен степени $n-1$ представляется с использованием элементарных многочленов $p_i(x)$ степени $s-1$ в качестве коэффициентов при степенях переменной $y=x^s$ многочленом степени $k-1=]n/s[-1$:

$$\sum_{i=0}^{k-1} p_i(x)y^i.$$

Если эти коэффициенты рассматривать как элементы поля Галуа $GF(2^m),\ m \geq s$, то обычным "школьным"алгоритмом, перемножая "коэффициенты"в поле Галуа $GF(2^m)$ получим их произведение над этим полем.

заметим, что при $s \leq]m/2[$ и произведения "коэффициентов"над полем Галуа совпадут с произведениями в поле Галуа $GF(2^m)$. Поэтому в этом случае полученный многочлен (по переменной y)

$$\sum_{t=0}^{2k-2} s_t(x)y^t,$$

где $s_t(x)$ – многочлены степени не более $m-1$, эквивалентен произведению исходных многочленов над поле Галуа $GF(2)$.

Многочлен, представляющий произведение исходных многочленов над полем $GF(2)$, получается из этого выражения заменой переменной y степенью x^s и последующим приведением к полиномиальному виду.

Для использования метода Фурье надо правильно выбрать размерность m используемого поля $GF(2^m)$ и зависящий от нее параметр s.

Произведение двух многочленов (от переменной y) степени $k-1$ является многочленом степени $2k-2$ (по переменной $y=x^s$), и размерность 2^m-1 матрицы дискретного преобразования Фурье, предназначенной для умножения многочленов по методу Фурье, рассчитывается с учетом степени $2k-2$ произведения:

$$2^m-1 \geq (2k-1),\ .. \ 2^m \geq 2k,$$

откуда $m \geq \log_2 2k$.

Только в этом случае элементы второй строки матрицы будут разными. Таким образом, коэффициенты $p_i(x)$ сомножителей представляются как многочлены степени $m-1$.

$$\sum_{i_0}^{m-1} a_i x^i,\ a_s = a_{s+1} = \ldots = a_{2s-1} = 0.$$

Итак, для умножения многочленов степени $n-1$ следует использовать поле Галуа $GF(2^m)$, где m определяется неравенством

$$k \leq 2^{m-1}.$$

при этом $k-1$ есть максимально возможная степень сомножителей (по переменной y) и

$$k-1 \leq 2^{m-1}-1.$$

Допустимая степень $n-1$ (по переменной x) многочленов-сомножителей определяется неравенством $n-1 \leq k \times s-1 = k \times]\frac{m}{2}[-1$.

Например, при $m = 8$, то есть при использовании преобразования Фурье над полем $GF(2^8)$

$$k - 1 \leq 2^7 - 1 = 127,\ n - 1 \leq 128 \times 4 - 1 = 511.$$

При $m = 4$

$$k - 1 \leq (2^3) - 1 = 7,\ n - 1 \leq 8 \times 2 - 1 = 15.$$

При $m = 3$

$$k - 1 \leq (2^2) - 1 = 3,\ n - 1 \leq 4 \times 2 - 1 = 7.$$

При $m = 2$

$$k - 1 \leq [2^1 - 1] = 1,\ n - 1 \leq 2 \times 1 - 1 = 1.$$

Обратим внимание, что несоблюдение этих правил выбора поля приводит к ошибкам, если при перемножении элементарных многочленов возникает необходимость приведения по модулю неприводимого многочлена данного поля.

Приведем таблицу в которой для некоторых степеней n расширения поля $GH(2)$ даны разложения порядков мультипликативных групп $GF(2^n)^*$ и указаны максимальные степени m многочленов, допустимые для умножения методом ДПФ над полем $GF(2^n)$.

n	$\lvert GF2^n * n\rvert = 2^n - 1$	m
2	3	1
3	7	7
4	$3 \cdot 5 = 15$	15
5	31	47
6	$3 \cdot 3 \cdot 7 = 63$	95
7	127	255
8	$3 \cdot 5 \cdot 17 = 255$	511
9	$7 \cdot 73 = 511$	1279
10	$3 \cdot 11 \cdot 31 = 1023$	2669
11	$23 \cdot 89 = 2047$	6143
12	$3 \cdot 3 \cdot 5 \cdot 7 \cdot 13 = 4095$	12288

Как видно из таблицы, наиболее значительное ускорение за счет применения БПФ можно получить при использовании БПФ над полем $GF(2^{12})$, где порядок мультипликативной группы раскладывается на пять простых множителя. Эксперименты показали, что и при использовании БПФ над $GF(2^8)$ скорость умножения практически такая же, как при использовании «школьного» алгоритма умножения. При использовании БПФ над полем $GF(2^{12})$ скорость умножения примерно такая же, как при умножении по методу Карацубы. Заметим, однако.что предварительное вычисление степеней примитивного корня связано с использованием большого объема памяти. Для его понижения до практически приемлемого возможно на основе компромисса «память - время», то-есть хранения в памяти некоторых «опорных степеней примитивного элемента поля,» позволяющих многократно, но достаточно быстро вычислять остальные по мере необходимости. При этом время вычисления спектрального вектора возрастает не более, чем в два раза.

Изложенное выше основано на вычислительных экспериментах, выполненных Я.Ю.Грачевым [?].

Пример 3.5.1 Пусть $p(x) = 1 + x^2 + x^3$ и $q(x) = 1 + x + x^3$ два многочлена над полем Галуа $GF(2)$. Здесь $n = 4$ (степень многочленов $n - 1 = 3$). Отсюда получим, что наименьшее значение $m = 3$ при $s = 2$. если же взять, например, $m = s = 3$, то получим многочлены степени $k = 2$ (по переменной $y = x^3$). $P(x) = (1 + 0x + x^2) + (1 + 0x + 0x^2)y$ и $Q(x) = (1 + 1x + x^2) + (1 + 0x + 0x^2)y$, $y = x^3$ над полем Галуа $GF(2^3)$, порождаемом неприводимым многочленом $x^3 + x + 1$. Выберем примитивный корень $\omega = (1 + 1x)$. тогда матрица $\mathbf{D_n}$

дискретного преобразования Фурье, как и матрица обратного преобразования $\mathbf{D_n^{-1}}$ имеют размер 7×7 :

$$\mathbf{D_n} = \begin{pmatrix} & 7 & 1 & 2 & 3 & 4 & 5 & 6 \\ 7 & 100 & 100 & 100 & 100 & 100 & 100 & 100 \\ 1 & 100 & 110 & 101 & 001 & 111 & 010 & 011 \\ 2 & 100 & 101 & 111 & 011 & 110 & 001 & 010 \\ 3 & 100 & 001 & 011 & 101 & 010 & 110 & 111 \\ 4 & 100 & 111 & 110 & 010 & 101 & 011 & 001 \\ 5 & 100 & 010 & 001 & 110 & 011 & 111 & 101 \\ 6 & 100 & 011 & 010 & 111 & 001 & 101 & 110 \end{pmatrix}$$

$$\mathbf{D_n^*} = \begin{pmatrix} & 7 & 1 & 2 & 3 & 4 & 5 & 6 \\ 7 & 100 & 100 & 100 & 100 & 100 & 100 & 100 \\ 1 & 100 & 011 & 010 & 111 & 001 & 101 & 110 \\ 2 & 100 & 010 & 001 & 110 & 011 & 111 & 101 \\ 3 & 100 & 111 & 110 & 010 & 101 & 011 & 001 \\ 4 & 100 & 001 & 011 & 101 & 010 & 110 & 111 \\ 5 & 100 & 101 & 111 & 011 & 110 & 001 & 010 \\ 6 & 100 & 011 & 010 & 111 & 001 & 101 & 110 \\ 6 & 100 & 110 & 101 & 001 & 111 & 010 & 011 \end{pmatrix}$$

Построим спектральные векторы для многочленов $p(x)$ и $q(x)$.

$$\mathbf{D_n} \times p(x) = \begin{pmatrix} & 7 & 1 & 2 & 3 & 4 & 5 & 6 \\ 7 & 100 & 100 & 100 & 100 & 100 & 100 & 100 \\ 1 & 100 & 110 & 101 & 001 & 111 & 010 & 011 \\ 2 & 100 & 101 & 111 & 011 & 110 & 001 & 010 \\ 3 & 100 & 001 & 011 & 101 & 010 & 110 & 111 \\ 4 & 100 & 111 & 110 & 010 & 101 & 011 & 001 \\ 5 & 100 & 010 & 001 & 110 & 011 & 111 & 101 \\ 6 & 100 & 011 & 010 & 111 & 001 & 101 & 110 \end{pmatrix} \times \begin{pmatrix} 101 \\ 100 \\ 000 \\ 000 \\ 000 \\ 000 \\ 000 \end{pmatrix} = \begin{pmatrix} 001 \\ 011 \\ 000 \\ 100 \\ 010 \\ 111 \\ 110 \end{pmatrix}$$

$$\mathbf{D_n} \times p(x) = \begin{pmatrix} & 7 & 1 & 2 & 3 & 4 & 5 & 6 \\ 7 & 100 & 100 & 100 & 100 & 100 & 100 & 100 \\ 1 & 100 & 110 & 101 & 001 & 111 & 010 & 011 \\ 2 & 100 & 101 & 111 & 011 & 110 & 001 & 010 \\ 3 & 100 & 001 & 011 & 101 & 010 & 110 & 111 \\ 4 & 100 & 111 & 110 & 010 & 101 & 011 & 001 \\ 5 & 100 & 010 & 001 & 110 & 011 & 111 & 101 \\ 6 & 100 & 011 & 010 & 111 & 001 & 101 & 110 \end{pmatrix} \times \begin{pmatrix} 111 \\ 100 \\ 000 \\ 000 \\ 000 \\ 000 \\ 000 \end{pmatrix} = \begin{pmatrix} 011 \\ 001 \\ 010 \\ 110 \\ 000 \\ 101 \\ 100 \end{pmatrix}$$

Спектральным вектором произведения является вектор

$$\begin{pmatrix} 101 \\ 101 \\ 000 \\ 110 \\ 000 \\ 011 \\ 110 \end{pmatrix}.$$

Построим сигнальный вектор произведения

Являясь произведением двух многочленов (от переменной y) над полем $GF(2^3)$, он не соответствует произведению исходных многочленов над полем $GF(2)$.

Правильный результат получается при использовании того же поля $GF(2^3)$, но при значении $s =]3/2[= 2$, то есть и представлении исходных многочленов степени (по перменной x) не более семи многочленами от переменной $y = x^2$.

Пример 3.5.2 Возведем в квадрат многочлен $1 + y + y^2 + y^3, y = x2$. Для этого умножим метрицу $\mathbf{D_n}$ на соответствующий сигнальный вектор, получим при этом спектральный вектор:

$$\mathbf{D_n} \times p(x) = \begin{pmatrix} & 7 & 1 & 2 & 3 & 4 & 5 & 6 \\ 7 & 100 & 100 & 100 & 100 & 100 & 100 & 100 \\ 1 & 100 & 110 & 101 & 001 & 111 & 010 & 011 \\ 2 & 100 & 101 & 111 & 011 & 110 & 001 & 010 \\ 3 & 100 & 001 & 011 & 101 & 010 & 110 & 111 \\ 4 & 100 & 111 & 110 & 010 & 101 & 011 & 001 \\ 5 & 100 & 010 & 001 & 110 & 011 & 111 & 101 \\ 6 & 100 & 011 & 010 & 111 & 001 & 101 & 110 \end{pmatrix} \times \begin{pmatrix} 10 \\ 100 \\ 100 \\ 100 \\ 000 \\ 000 \\ 000 \end{pmatrix} = \begin{pmatrix} 000 \\ 110 \\ 101 \\ 011 \\ 111 \\ 001 \\ 010 \end{pmatrix}$$

Далее для получения квадрата исходного многочлена умножим матрицу $\mathbf{D_n^*}$ обратного преобразования Фурье на вектор, полученный возведением в квадрат (в поле $FG(2^m)$) каждой компоненты спектрального вектора:

$$\mathbf{D_n^*} \times p(x) = \begin{pmatrix} & 7 & 1 & 2 & 3 & 4 & 5 & 6 \\ 7 & 100 & 100 & 100 & 100 & 100 & 100 & 100 \\ 1 & 100 & 011 & 010 & 111 & 001 & 101 & 110 \\ 2 & 100 & 010 & 001 & 110 & 011 & 111 & 101 \\ 3 & 100 & 111 & 110 & 010 & 101 & 011 & 001 \\ 4 & 100 & 001 & 011 & 101 & 010 & 110 & 111 \\ 5 & 100 & 101 & 111 & 011 & 110 & 001 & 010 \\ 6 & 100 & 110 & 101 & 001 & 111 & 010 & 011 \end{pmatrix} \times \begin{pmatrix} 000 \\ 101 \\ 111 \\ 110 \\ 011 \\ 011 \\ 001 \end{pmatrix} = \begin{pmatrix} 100 \\ 000 \\ 100 \\ 000 \\ 100 \\ 000 \\ 100 \end{pmatrix}$$

В приложении приведены матрицы $\mathbf{D_n}$, $\mathbf{D_n^*}$ дискретного преобразования Фурье над полем $GF(2^4)$, порождаемом неприводимым многочленом $x^4 + x + 1$, которые построены для примитивного корня $\omega = 1 + x$.

Их можно использовать для умножения многочленов вида $p'(y) = a'_0(x) + a'_1(x)y + a'_2(x)y^2 \cdots + a'_7(x)y^7 = (a_0 + a_1x) + (a_2 + a_3x)x^2 + \cdots + (a_{12} + a_{13}x)x^{14}))$.

При этом матрица $\mathbf{D_n^*}$ используется для получения произведения

$$c'_0(x) + c'_1(x)y + c'_2(x)y^2 + \cdots + c'_{14}(x)y^14 = \sum_{i=0}^{14}(c_{0i}(x) + c_{1i}(x) + c_{2i}(x) + c_{3i}(x))x^{2i}.$$

В общем случае матрицы размеров $(2^m - 1)x(2^m - 1)$, где m четно, позволяют умножать многочлены над полем Галуа $GF(2)$ степени $n - 1 = 2^{m-2}m - 1$. Для этого многочлены

$$a_0 + a_1x + a_2x^2 + \cdots x^s$$

рассматриваются как многочлены

$$a'_0(x) + a'_1(x)y + a'_2(x)y^2 + a'_3(x)y_3 + \cdots + a'_p(x)y^p,\ y = x^{m/2},$$

над полем Галуа ($GF(2^m)$. Таким образом, для умножения многочленов методом дискретного преобразования Фурье можно а) Заранее вычислить и хранить для многократного использования матрицы $\mathbf{D_n}$ и $\mathbf{D_n}^{-1}$ для заранее выбранного корня n-ой степени из единицы. Тогда преобразования заключаются в умножении вектора на соответствующую матрицу. Объем такой матрицы при $n = 2^8 - 1$ составляет $(2^8 - 1)x(2^8 - 1) \approx 2^16$ байт, или 64Кбайт. (при этом можно умножать многочлены степени 511 с использование таблицы умножения байтов), при $n = 10$ он составил бы $\approx 2^{10}x2^{10} = 2^{20}$ байт, или 1Мбайт, можно было бы умножать многочлены степени 2559 используя алгоритм умножения многочленов в поле $GF(2^16)$.

Для уменьшения объема памяти, а также ускорения вычислений используем тот факт, что таблицы, как правило (если порядок 2^m используемого поля Галуа разлагается на нетривиальные простые множители) избыточны и если это не заметить, то многие вычисления повторяются многократно. Избыточность проявляется в периодическом характере некоторых столбцов. Период равен степени соответствующего столбцу корня их единицу. Только столбцы, соответствующие примитивным корням не раскладываются в три или более простых периодов.

Заметим, что если порядок $2^m - 1$ поля $GF(2^m)$ – простое число, то подобное упрощение вычислений невозможно. Например, при $m = 3$ или $m = 7$.

На основе матрицы $\mathbf{D_n}$ построим следующие матрицы $\mathbf{D_{n,1}}$, $\mathbf{D_{n,3}}$, $\mathbf{D_{n,5}}$ $\mathbf{D_{n,15}}$, соотвтетствующие столбцам, представляющим корни степени 1, 15, 5 и 3.

$$\mathbf{D_{n,1}} =$$

	15	1	2	3	4	5	6	7	8	9	10	11	12	13	14
15	1000	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**

$$\mathbf{D_{n,3}} =$$

	15	1	2	3	4	5	6	7	8	9	10	11	12	13	14
15	**0**	**0**	**0**	**0**	**0**	1000	**0**	**0**	**0**	**0**	1000	**0**	**0**	1000	**0**
1	**0**	**0**	**0**	**0**	**0**	0110	**0**	**0**	**0**	**0**	1110	**0**	**0**	1101	**0**
2	**0**	**0**	**0**	**0**	**0**	1110	**0**	**0**	**0**	**0**	0110	**0**	**0**	1001	**0**

$$\mathbf{D_{n,5}} =$$

	15	1	2	3	4	5	6	7	8	9	10	11	12	13	14
15	**0**	**0**	**0**	1000	**0**	**0**	1000	**0**	**0**	1000	**0**	**0**	1000	**0**	**0**
1	**0**	**0**	**0**	1111	**0**	**0**	0101	**0**	**0**	0011	**0**	**0**	0001	**0**	**0**
2	**0**	**0**	**0**	0101	**0**	**0**	0001	**0**	**0**	1111	**0**	**0**	0011	**0**	**0**
3	**0**	**0**	**0**	0011	**0**	**0**	1111	**0**	**0**	0001	**0**	**0**	0101	**0**	**0**
4	**0**	**0**	**0**	0001	**0**	**0**	0011	**0**	**0**	0101	**0**	**0**	1111	**0**	**0**

$\mathbf{D_{n,15}} =$

	15	1	2	3	4	5	6	7	8	9	10	11	12	13	14
15	**0**	1000	1000	**0**	1000	**0**	**0**	1000	1000	**0**	**0**	1000	**0**	**0**	1000
1	**0**	1100	1010	**0**	0100	**0**	**0**	1011	0010	**0**	**0**	1001	**0**	**0**	0111
2	**0**	1010	0100	**0**	0010	**0**	**0**	0111	1100	**0**	**0**	1011	**0**	**0**	1101
3	**0**	1111	0101	**0**	0001	**0**	**0**	0101	0011	**0**	**0**	1111	**0**	**0**	0001
4	**0**	0100	0010	**0**	1100	**0**	**0**	1101	1010	**0**	**0**	0111	**0**	**0**	1001
5	**0**	0110	1110	**0**	0110	**0**	**0**	0110	1110	**0**	**0**	1110	**0**	**0**	1110
6	**0**	0101	0001	**0**	0011	**0**	**0**	0001	1111	**0**	**0**	0101	**0**	**0**	0011
7	**0**	1011	0111	**0**	1101	**0**	**0**	0100	1001	**0**	**0**	1010	**0**	**0**	0010
8	**0**	0010	1100	**0**	1010	**0**	**0**	1001	0100	**0**	**0**	1101	**0**	**0**	1011
9	**0**	0011	1111	**0**	0101	**0**	**0**	1111	0001	**0**	**0**	0011	**0**	**0**	0101
10	**0**	1110	0110	**0**	1110	**0**	**0**	1110	0110	**0**	**0**	0110	**0**	**0**	0110
11	**0**	1001	1011	**0**	0111	**0**	**0**	1010	1101	**0**	**0**	1100	**0**	**0**	0100
12	**0**	0001	0011	**0**	1111	**0**	**0**	0011	0101	**0**	**0**	0001	**0**	**0**	1111
13	**0**	1101	1001	**0**	1011	**0**	**0**	1100	0111	**0**	**0**	0010	**0**	**0**	1010
14	**0**	0111	1101	**0**	1001	**0**	**0**	0010	1011	**0**	**0**	0100	**0**	**0**	1100

Удалив нулевые столбцы, получим совсем простые записи этих матриц.

$\mathbf{D_{n,1}} =$

	15
15	1000

$\mathbf{D_{n,3}} =$

	5	10	13
15	1000	1000	1000
1	0110	1110	1101
2	1110	0110	1001

$\mathbf{D_{n,5}} =$

	3	6	9	12
15	1000	1000	1000	1000
1	1111	0101	0011	0001
2	0101	0001	1111	0011
3	0011	1111	0001	0101
4	0001	0011	0101	1111

$\mathbf{D_{n,15}} =$

	1	2	4	7	8	11	14
15	1000	1000	1000	1000	1000	1000	1000
1	1100	1010	0100	1011	0010	1001	0111
2	1010	0100	0010	0111	1100	1011	1101
3	1111	0101	0001	0101	0011	1111	0001
4	0100	0010	1100	1101	1010	0111	1001
5	0110	1110	0110	0110	1110	1110	1110
6	0101	0001	0011	0001	1111	0101	0011
7	1011	0111	1101	0100	1001	1010	0010
8	0010	1100	1010	1001	0100	1101	1011
9	0011	1111	0101	1111	0001	0011	0101
10	1110	0110	1110	1110	0110	0110	0110
11	1001	1011	0111	1010	1101	1100	0100
12	0001	0011	1111	0011	0101	0001	1111
13	1101	1001	1101	1101	1001	1001	1001
14	0111	1101	1001	0010	1011	0100	1100

Не трудно убедиться, что

$$\mathbf{D_n} \times p'(y) = \mathbf{D_{n,5}} \times p'(y) || ((\mathbf{D_{n,1}} \times p'(y) || \mathbf{D_{n,3}} \times p'(y)) || \mathbf{D_{n,15}} \times p'(y))$$

Здесь $||$ означает кратное прибавление первого вектора ко второму, то есть первый вектор итерируется p раз и затем прибавляется, p есть частное от деления длины второго вектора на длину первого.

При этом вместо $n^2 = 225$ умножений и сложений элементарных многочленов будет выполнено $1 + 15 \times 7 + 5 \times 5 + 3 \times 2 = 1 + 95 + 25 + 6 = 127$ пары таких операций плюс несколько операций сложения (сколько ?) при "сборке."

Точно также можно упростить обратное преобразование.

Подобным образом можно поступать всегда, когда имеются непримитивные корни единицы. Максимальное ускорение получится, если порядок используемого поля есть степень двойки. (Это возможно, но не в расширении поля).

Рекомендуется рассмотренный прием использовать, пересмотрев примеры предыдущего парграфа.

Во всех случаях порядок его реализации следущий:

Определяются порядки корней ω^i единицы, $i = 1, \ldots p$, где p – порядок поля.

Строятся матрицы $\mathbf{D_{n,j}}$, $\mathbf{D^*_{n,j}}$, соответствующие коряням порядка j.

Эти матрицы используются для быстрого вычисления спектрального и сигнального векторов. При этом они используются по возрастанию порядка корней.

Заметим, что умножение матрицы $\mathbf{D_{n,15}}$ на соответствующую часть сигнального вектора можно упростить, заметив, что строки 1,3,5,6,10,12,13 содержат повторяющиеся элементы и поэтому лучше сначала сложить соответствующие компоненты сигнального вектора, а затем один раз умножить на элемент строки, сложив затем результаты.

Тогда останется только умножить квадратную матрицу, образованную оставшимися строками на остальные компоненты сигнального вектора. Заметим, что намера строк можно заменить таким образом, что они совпадут с номерами соответтвующих столбцов. (Строки со старыми номерами совпадают со строками с измененными номерами)

Эта квадратная матрица имеет вид.

$$\mathbf{D'_{n,15}} = \begin{pmatrix} & 1 & 2 & 4 & 7 & 8 & 11 & 14 \\ 1 & 1100 & 1010 & 0100 & 1011 & 0010 & 1001 & 0111 \\ 2 & 1010 & 0100 & 0010 & 0111 & 1100 & 1011 & 1101 \\ 4 & 0100 & 0010 & 1100 & 1101 & 1010 & 0111 & 1001 \\ 7 & 1011 & 0111 & 1101 & 0100 & 1001 & 1010 & 0010 \\ 8 & 0010 & 1100 & 1010 & 1001 & 0100 & 1101 & 1011 \\ 11 & 1001 & 1011 & 0111 & 1010 & 1101 & 1100 & 0100 \\ 14 & 0111 & 1101 & 1001 & 0010 & 1011 & 0100 & 1100 \end{pmatrix}$$

По этой же причине удаляется и первая строка матриц $\mathbf{D_{n,3}}$ и $\mathbf{D_{n,5}}$, изменяются также и номера строк:

$$\mathbf{D'_{n,1}} = \begin{pmatrix} & 15 \\ 15 & 1000 \end{pmatrix} \quad \mathbf{D'_{n,3}} = \begin{pmatrix} & 5 & 10 & 13 \\ 5 & 0110 & 1110 & 1101 \\ 10 & 1110 & 0110 & 1001 \end{pmatrix} \quad \mathbf{D'_{n,5}} = \begin{pmatrix} & 3 & 6 & 9 & 12 \\ 3 & 1111 & 0101 & 0011 & 0001 \\ 6 & 0101 & 0001 & 1111 & 0011 \\ 9 & 0011 & 1111 & 0001 & 0101 \\ 12 & 0001 & 0011 & 0101 & 1111 \end{pmatrix}$$

Заметим, что матрица $\mathbf{D'_{n,5}}$ может быть преобразована в матрицу циклической сверки. для этого следует переставить последние два столбца и затем две последние строки. Умножение полученной матрицы на соответсвующуч часть сигнального вектора (с учетом перстановки) может быть произведено по алгоритму короткой свертки. По существу надо амножить первую строку матрицы на часть сигнального вектора по модулю многочлена $X^4 + 1$. Это свойство матрицы обусловлено тем, что ее столвцы (и строки) образуют неединичные элементы мультипликативной группы $FG(5)^*$.

Матрицы $\mathbf{D'_{n,3}}$ и $\mathbf{D'_{n,15}}$ подобным свойством не обладают, но содержат <хорошие> в этом смысле подматрицы: из первой гадо удалть последний столбец, а из второй - 11-й столбец и 11-ю строку (их влияние на результирующий сигнальный вектор учитывается отдельно, при этом не удастся что-либо упростить, как в случае перехода к матрицам $\mathbf{D'_{n,i}}$ так как нет одинаковых элементов). После удаления получаем <хорошие> матрицы

$$\mathbf{D''_{n,3}} = \begin{pmatrix} & 5 & 10 \\ 5 & 0110 & 1110 \\ 10 & 1110 & 0110 \end{pmatrix}$$

$$\mathbf{D''_{n,15}} = \begin{pmatrix} & 1 & 2 & 4 & 7 & 8 & 14 \\ 1 & 1100 & 1010 & 0100 & 1011 & 0010 & 0111 \\ 2 & 1010 & 0100 & 0010 & 0111 & 1100 & 1101 \\ 4 & 0100 & 0010 & 1100 & 1101 & 1010 & 1001 \\ 7 & 1011 & 0111 & 1101 & 0100 & 1001 & 0010 \\ 8 & 0010 & 1100 & 1010 & 1001 & 0100 & 1011 \\ 14 & 0111 & 1101 & 1001 & 0010 & 1011 & 1100 \end{pmatrix}$$

Эти матрицы соответсвуют мультипликативным группам $GF(3)$ и $GF(7)$.

Приложение. Матрицы дискретного преобразования Фурье над полем $GF(2^4)$

$\mathbf{D_n} =$

	15	1	2	3	4	5	6	7	8	9	10	11	12	13	14
15	1000	1000	1000	1000	1000	1000	1000	1000	1000	1000	1000	1000	1000	1000	1000
1	1000	1100	1010	1111	0100	0110	0101	1011	0010	0011	1110	1001	0001	1101	0111
2	1000	1010	0100	0101	0010	1110	0001	0111	1100	1111	0110	1011	0011	1001	1101
3	1000	1111	0101	0011	0001	1000	1111	0101	0011	0001	1000	1111	0101	1000	0001
4	1000	0100	0010	0001	1100	0110	0011	1101	1010	0101	1110	0111	1111	1101	1001
5	1000	0110	1110	1000	0110	1110	1000	0110	1110	1000	0110	1110	1000	1001	1110
6	1000	0101	0001	1111	0011	1000	0101	0001	1111	0011	1000	0101	0001	1000	0011
7	1000	1011	0111	0101	1101	0110	0001	0100	1001	1111	1110	1010	0011	1101	0010
8	1000	0010	1100	0011	1010	1110	1111	1001	0100	0001	0110	1101	0101	1001	1011
9	1000	0011	1111	0001	0101	1000	0011	1111	0001	0101	1000	0011	1111	1000	0101
10	1000	1110	0110	1000	1110	0110	1000	1110	0110	1000	1110	0110	1000	1101	0110
11	1000	1001	1011	1111	0111	1110	0101	1010	1101	0011	0110	1100	0001	1001	0100
12	1000	0001	0011	0101	1111	1000	0001	0011	0101	1111	1000	0001	0011	1000	1111
13	1000	1101	1001	1000	1101	1001	1000	1101	1001	1000	1101	1001	1000	1101	1001
14	1000	0111	1101	0001	1001	1110	0011	0010	1011	0101	0110	0100	1111	1001	1100

Матрица обратного преобразования имеет вид

$\mathbf{D_n}^{-1} =$

	15	1	2	3	4	5	6	7	8	9	10	11	12	13	14
15	1000	1000	1000	1000	1000	1000	1000	1000	1000	1000	1000	1000	1000	1000	1000
1	1000	0111	1101	0001	1001	1110	0011	0010	1011	0101	0110	0100	1111	1010	1100
2	1000	1101	1001	1000	1101	1001	1000	1101	1001	1000	1101	1001	1000	1101	1000
3	1000	0001	1000	0101	1111	1000	0001	0011	0101	1111	1000	0001	0011	0101	1111
4	1000	1001	1101	1111	0111	1110	0101	1010	1101	0011	0110	1100	0001	0010	0100
5	1000	1110	1001	1000	1110	0110	1000	1110	0110	1000	1110	0110	1000	1110	0110
6	1000	0011	1000	0001	0101	1000	0011	1111	0001	0101	1000	0011	1111	0001	0101
7	1000	0010	1101	0011	1010	1110	1111	1001	0100	0001	0110	1101	0101	0111	1011
8	1000	1011	1001	0101	1101	0110	0001	0100	1001	1111	1110	1010	0011	1100	0010
9	1000	0101	1000	1111	0011	1000	0101	0001	1111	0011	1000	0101	0001	1111	0011
10	1000	0110	1101	1000	0110	1110	1000	0110	1110	1000	0110	1110	1000	0110	1110
11	1000	0100	1001	0001	1100	0110	0011	1101	1010	0101	1110	0111	1111	1011	1001
12	1000	1111	1000	0011	0001	1000	1111	0101	0011	0001	1000	1111	0101	0011	0001
13	1000	1010	0100	0101	0010	1110	0001	0111	1100	1111	0110	1011	0011	1001	1101
14	1000	1100	1010	1111	0100	0110	0101	1011	0010	0011	1110	1001	0001	1101	0111

3.6 Возведение в степень и инвертирование в $GF(2^n)$

3.6.1 Имплементация возведения многочленов над $GF(2)$ в степень 2^n

Для ускорения выполнения операции возведения в квадрат многочлена над полем $GF(2)$ мы заранее составляем таблицу, в которой для каждого байта име-

ются двухбайтовые слова, полученные вставкой дополнительных нулей между битами.

(Напомним, что возведение в степень p многочлена над простым полем Галуа $GF(p)$ сводится к замене каждого терма многочлена его p-й степенью, см. Лекцию 2.)

При возведения в квадрат с использованием построенной таблицы мы вставляем между битами, задающими элемент поля (т.е. многочлен), нули и затем приводим полученный результат по модулю неприводимого многочлена, который порождает данное поле.

Отметим, что возведение в квадрат в нормальном базисе осуществляется просто циклическим сдвигом коэффициентов полинома.

Многократным повторением подобного преобразования получают степень $f(x)^{2^m}$ данного многочлена

3.6.2 Имплементация инвертирования в полиномиальном или нормальном базисах

Рассмотрим модификацию метода возведения многочленов в степень и их инвертирования в полиномиальном или нормальном базисах, отличающуюся тем, что используется не рекурсия, а явные вычисления, основанные на двоичном разложении степени и машинном представлении многочленов. Описанная здесь реализация использует также некоторые программные эвристики.

Пусть многочлен $f(x)$ степени меньшей t задан в полиномиальном S или нормальном N базисе. Требуется получить представление многочлена $f^{-1}(x)$ в заданном базисе B, $\quad B = S$ или $B = N$.

Из равенств

$$f^{-1}(x) = f^{2^t-2}(x) = \left(f^{2^{t-1}-1}\right)^2(x)$$

следует, что для вычисления $f^{-1}(x)$ можно сначала найти $g(x) = f^{2^{t-1}-1}$, а затем возвести полученный многочлен $g(x)$ в квадрат.

Опишем часть алгоритма, касающуюся возведения многочлена $f(x)$ в степень $2^t - 1$, где t - натуральное. Воспользуемся равенством

$$2^t - 1 = \begin{cases} \left(2^{t/2} - 1\right)\left(2^{t/2} + 1\right), \\ \text{если} \quad t \quad \text{четно}, \\ 2^{t-1} + \left(2^{(t-1)/2} - 1\right)\left(2^{(t-1)/2} + 1\right), \\ \text{если} \quad t \quad \text{нечетно}. \end{cases}$$

Отсюда

$$2^t - 1 = \begin{cases} \left(2^{t/2} - 1\right)2^{t/2} + \left(2^{t/2} - 1\right), \\ \text{если } t \quad \text{четно}, \\ 2^{t-1} + \left(2^{(t-1)/2} - 1\right)2^{(t-1)/2} + \left(2^{(t-1)/2} - 1\right), \\ \text{если } t \quad \text{нечетно}. \end{cases}$$

Таким образом, для f^{2^t-1} получаем разложение

$$f^{2^t-1} = \begin{cases} \left(f^{2^{t/2}-1}\right)^{2^{t/2}} \cdot f^{2^{t/2}-1}, \\ \text{если} \quad t \quad \text{чётно}, \\ f^{2^{t-1}} \cdot \left(f^{2^{(t-1)/2}-1}\right)^{2^{(t-1)/2}} \cdot f^{2^{(t-1)/2}-1}, \\ \text{если} \quad t \quad \text{нечетно}. \end{cases}$$

Покажем, как с использованием приведенных выше разложений вычислить f^{2^t-1}, если

$$t = \sum_{i=0}^{s} a_i 2^i.$$

Старший разряд a_s числа t равен 1. Пусть для некоторого s', $s' < s$, уже вычислен многочлен

$$f^{2^{t'}-1},$$

где

$$t' = \sum_{i=0}^{s'} a_{s-s'+i} 2^i.$$

Найдем многочлен $f^{2^{t''}-1}$ для

$$t'' = \sum_{i=0}^{s'+1} a_{s-s'-1+i} 2^i.$$

Для этого рассмотрим два случая.

Случай 1. Пусть $a_{s-s'-1} = 0$. Тогда $t'' = 2t'$ и, учитывая приведенное выше разложение, получаем

$$f^{2^{t''}-1} = \left(f^{2^{t'}-1}\right)^{2^{t'}} \cdot f^{2^{t'}-1}.$$

Таким образом, нужно использовать один раз алгоритм возведения в степень для вычисления $\left(f^{2^{t'}-1}\right)^{2^{t'}}$ и один раз алгоритм умножения.

Случай 2. Пусть $a_{s-s'-1} = 1$. Тогда $t'' = 2t' + 1$ и, используя приведенное выше разложение, получаем

$$f^{2^{t''}-1} = \left(f^{2^{t'}-1}\right)^{2^{t'}} \cdot f^{2^{t'}-1} \cdot f^{2^{2t'}}.$$

Следовательно, нужно использовать два раза алгоритм возведения в степень для вычисления многочленов $\left(f^{2^{t'}-1}\right)^{2^{t'}}$, $f^{2^{2t'}}$, а затем два раза алгоритм умножения.

Выполняя описанные выше действия, начиная с $s'=0$, чему соответствует $t'=a_s$, мы вычислим f^{2^t-1} с использованием $n(t)+b(t)-2$ умножений, где $b(t)$, как указывалось выше, - число разрядов в двоичном разложении числа t, а $n(t)$ - число единичных разрядов в этом разложении. Таким образом порядок числа умножений равен $O(\log t)$. Кроме того, понадобится $n(t)+b(t)-2$ раз использовать алгоритм возведения в степень.

Рассмотренный выше способ возведения многочлена f в степень 2^t-1 с использованием двоичного представления числа t, $t=(a_s,\dots a_1,a_0)$, $a_s=1$, $s>0$. можно оформить в виде следующего алгоритма.

1.Принять $\varphi=f$.

2.Для $s'=0,s-1$ вычислить

$t'=\sum_{i=0}^{s'} a_{s-s'-i}\cdot 2^i$.

Если $a_{s-s'-1}=0$ принять

$\varphi=\varphi^{2^{t'}}\cdot\varphi$,

иначе принять

$\varphi=\varphi^{2^{t'}}\cdot\varphi\cdot f^{2^{t'}}$.

Алгоритм работает как в полиномиальном, так и в нормальном базисах.

Пример 3.6.1 В поле $GF(2^3)$ многочленов, рассматриваемых по модулю неприводимого многочлена $p(x)=1+x^2+x^3$ вычислим многочлен $f^{2^5-1}(x)$, если $f(x)=1+x$.

Здесь $t=5$, и двоичное разложение этого числа есть $t=(a_2,a_1,a_0)=(1,0,1)$.

1. Примем $\varphi=f=(1+x)$,

2.1 Для $s'=0$ имеем

$t'=a_2=1$,

$a_{s-s'-1}=a_{2-0-1}=a_1=0$, поэтому

$\varphi=\varphi^{2^1}\cdot\varphi=\varphi^2\cdot\varphi=(1+x^2)(1+x)=1+x+x^2+x^3=x$.

2.2 При $s'=1$ получаем

$t'=a_{2-1-0}2^0+a_{2-1-1}2^1=a_12^0+a_02^1=2$.

$a_{s-s'-1}=a_{2-1-1}=a_0=1$, поэтому

$\varphi=\varphi^{2^{t'}}\cdot\varphi\cdot f^{2^{2t'}}=$

$=x^{2^2}\cdot x\cdot f^{2^{2^2}}=x^4\cdot x\cdot(1+x)^{2^4}=$

$(1+x+x^2)\cdot x\cdot 1+x=(1+x^2)(1+x)=x$.

Пусть в рассмотренном выше поле нужно вычислить многочлен $f^{-1}(x)$, где, как и прежде, $f(x)=1+x$. Тогда

$$f^{-1}(x)=f^{2^3-2}(x)=\left(f^{2^2-1}(x)\right)^2.$$

(для вычисления $f^{2^2-1}(x)$ принимаем $\varphi=f=(1+x)$, далее для $s'=0$ находим $t'=1$, $a_{1-0-1}=a_0=0$, что влечет $\varphi^{2^1}\cdot\varphi=\varphi^2\dot{\varphi}=(1+x)^2\cdot(1+x)=x$.

Таким образом, $f^{-1}(x)=\varphi^2=x^2$.

3.6.3 Быстрый алгоритм возведения в степень в конечном поле малой характеристики в случае использования полиномиального базиса

Для возведения многочлена с коэффициентами из поля $GF(2)$

$$f(x)=a_0+a_1x+\ldots+a_{k-1}x^{k-1}$$

в степень 2^m по модулю заданного неприводимого над $GF(2)$ многочлена $q(x)$ можно воспользоваться формулой

$$f(x)^{2^m} = a_0 + a_1 x^{2^m} + \ldots + a_{k-1} x^{(k-1)2^m},$$

в которой степени $x^{2^m i}, i = 0, \ldots, k-1$ берутся по модулю многочлена $q(x)$ степени k. Составим из коэффициентов этих многочленов $k \times k$ матрицу Q_m.

Эту матрицу можно вычислить со сложностью $O(mk^2)$ единственный раз, так как она не зависит от $f(x)$. Умножая вектор a коэффициентов многочлена $f(x)$ на матрицу Q_m, что можно сделать со сложностью $O(k^2/\log k)$, получаем коэффициенты многочлена $f(x)^{2^m}$. Последовательно возводя в квадрат, можно вычислить $f(x)^{2^n}$ со сложностью $O(nk)$. Но если выбрать $m = \frac{O(k)}{\sqrt{\log k}}, s = \lceil n/m \rceil, r = n - sm < m$, то можно с помощью указанного выше способа возвести f в степень 2^m, выполнить это s раз, получив $f^{2^{ms}}$, а потом возведя тем же способом, но с помощью матрицы Q_r, в степень 2^r получаем в итоге $(f^{2^{ms}})^{2^r} = f^{2^n}$ со сложностью $O(sk^2/\log k) = \frac{O(nk)}{\sqrt{\log k}}$, причем это можно сделать для любого $n > m$, если предварительно вычислить все матрицы $Q_r, r = 1, \ldots, m$

Пример 3.6.2 Пусть $k = 5, \quad p(x) = x^4 + x + 1, \quad f(x) = x^3 + 1, \quad m_0 = 2$ и требуется найти $f^{2^3}(x)$, выразив его в полиномиальном базисе. Сначала представим в полиномиальном базисе одночлены $\{1, x^6, x^{12}, x^{24}\}$. Получим бинарную матрицу

Таблица 1.

x^3	x^2	x	x^0	
0	0	0	1	x^0
1	1	0	0	x^6
1	1	1	1	x^{12}
1	0	1	0	x^{24}

Образуя линейную комбинацию строк этой матрицы с коэффициентами многочлена $f(x)$, получим (как суммы первой и третьей строк)

$$f^{2^3}(x) = x^3 + x + 1.$$

Заметим теперь, что при n порядка 200 лучше использовать стандартный метод умножения матрицы на вектор, разбивая матрицу на полосы по 32 строки и заменяя булеву n на n матрицу на матрицу размера $n/32$ на n, состоящую из «длинных» целых чисел, и умножая последнюю матрицу на вектор, используя операции с 32-битовыми числами. Тогда сложность умножения будет равна $n^2/16$. Так как сложность возведения в квадрат в полиномиальном базисе равна $3n/8$, то при вычислении обратного элемента в поле $GF(2^n)$ последнюю операцию возведения в степень $2^{(n-1)/2}$ выгоднее выполнить описанным выше методом (может быть и предпоследнюю тоже).

Приведём детальное описание рассмотренного алгоритма. В нем будет использованы операции над матрицами, элементами которых являются «длинные» целые числа, и операция умножения вектора на матрицу, описываемые следующим образом.

Пусть задана таблица T с k строчками и n столбцами, на пересечении i-ой строки и j-го столбца которой находится «длинное» целое неотрицательное число t_{ij}, не превосходящие $2^s-1, \quad (n-1)s < k \le ns$. Таким образом, таблица T задает квадратную бинарную матрицу M с k строками и k столбцами, на пересечении i-ой строки и j-го столбца которой находится число a_{ij}. Введем операцию "U"над таблицами. Результатами применения этой операции к таблице T являются таблица $U(T)$, имеющая n строк и k столбцов.

Содержательно мы преобразуем сначала таблицу T к матрице M и тем самым определяем элементы a_{ij}. Через q и r обозначим соответственно числа $]j/s[$ и $j-(q-1)s$. Тогда число a_{ij} равно остатку от деления $[t_{iq}/2^{r-1}]$ на 2.

Определим таблицу $U(T)$, на пересечении i-ой строки и j-го столбца которой находится целое неотрицательное число u_{ij}, для каждого $j, \quad 1 \le j \le k$ и для каждого $i, \quad 1 \le i \le n$ полагая

$$u_{ij} = \sum_{i'=(i-1)s}^{\min(is-1,k)} a_{i'j} 2^{i'-(i-1)s}.$$

Примечание. Преобразовать таблицу T в таблицу $U(T)$ можно и не прибегая к промежуточному построению бинарной таблицы M.

Далее, пусть $v = (t_1, t_2, \ldots, t_n)$, где t_i - целое неотрицательное число не превосходящее $2^s-1, \quad i = 1, 2, \ldots, n$. Для каждого $j, \quad j = 1, 2, \ldots, k$ положим

$$v'_j = \sum_{i=1}^{n} t_i u_{ij},$$

где произведение целых неотрицательных чисел t_i и u_{ij} осуществляется покомпонентно, т.е. если $t_i = \sum_{i_1=0}^{s-1} a_{i_1} 2^{i_1}, \quad u_{ij} = \sum_{i_1=0}^{s-1} b_{i_1} 2^{i_1}$, то $t_i u_{ij} = \sum_{i_1=0}^{s-1} a_{i_1} b_{i_1} 2^{i_1}$; эти произведения суммируются также покомпонентно по модулю 2. В полученном векторе $v' = (v'_1, v'_2, \ldots, v'_k)$ каждая компонента v'_j является целым неотрицательным числом не превосходящим 2^s-1. Положим $c_j = 0$, если число единиц в двоичном разложении v'_j четно и $c_j = 1$ в противном случае. Вектор $c = (c_1, c_2, \ldots, c_k)$ преобразуем в вектор $u = (u_1, u_2, \ldots, u_n)$, положив

$$u_j = \sum_{i_1=s(j-1)+1}^{\min(sj,k)} c_{i_1} 2^{i_1-s(j-1)-1}.$$

Полученный вектор u является результатом умножения вектор-строки v на таблицу T.

Пример 3.6.3 Матрице M из таблицы 1 (см. пример выше) соответствуют таблицы

$$T = \begin{matrix} 0 & 1 \\ 3 & 0 \\ 3 & 3 \\ 2 & 2 \end{matrix}; \; U(T) = \begin{matrix} 1 & 1 & 0 & 2 \\ 3 & 2 & 3 & 1 \end{matrix}$$

Многочлен $f(x) = X+3+1$ представляется вектором $v = (2, 1)$. Умножая его на таблицу T, получим вектор $v' = (1, 0, 1, 1)$, где 0 и 1 – двухразрядные "целые"числа. заменяя их одноразрядными числами, получаем вектор $c = (1, 0, 1, 1)$, который представляет результирующий многочлен и в сжатом виде записывается как $u = (2, 3)$.

Возведение многочлена $f(x)$ в степень 2^m непосредственно в полиномиальном базисе можно выполнить следующим образом.

1. Последовательно для $i = 0, 1, \ldots, k-1$ найти разложение одночленов x^{i2^m} в полиномиальном базисе. Так, если разложение $g_{i,m-1}(x)$ для $x^{i2^{m-1}}$ уже найдено, то искомое разложение в полиномиальном базисе для одночлена $x^{i2^m} = g^2_{i,m-1}(x)$ получаем, вычислив остаток от деления многочлена $g^2_{i,m-1}(x)$ на многочлен $p(x)$.

2. Построить $k \times k$-матрицу $M = Q_m$ из нулей и единиц, i-я строка которой совпадает с набором

$$a_{i-1,k-1}, a_{i-1,k-2}, \ldots, a_{i-1,1}, a_{i-1,0}$$

коэффициентов многочлена $g_{i-1,m}(x)$,

$$g_{i-1,m}(x) = \sum_{j=0}^{k-1} a_{i-1,j} x^j.$$

3. По матрице M построить таблицы T_m и $U_m = U(T_m)$.

4. Для того, чтобы получить набор коэффициентов многочлена $f^{2^m}(x)$ в полиномиальном базисе, нужно, представив набор коэффициентов многочлена $f(x)$ в виде $]\frac{k}{s}[$-вектора с элементами из множества $\{0, 1, \ldots, 2^s - 1\}$ и умножить его на таблицу U_m.

Примечание. При совместном выполнении п.п.2,3 можно избежать сохранения матрицы M в полном объеме, если вычислять ее «полосами» по s строк с преобразованием этих «полос» в соответствующие строки таблицы U. Тогда не требуется строить и таблицу T.

Для реализации этой последовательности действий понадобится:

1. Асимптотически $C_1 k^2 m$ операций для построения матрицы M, где C_1 - некоторая константа.

2. Асимптотически $C_2 k^2$ операций для построения таблицы U_m по известной матрице M, где C_2 - константа.

3. Асимптотически $C_3 \frac{k^2}{s}$ операций над s - разрядными числами для умножения вектор-строки на таблицу U_m.

Таким образом, всего понадобится выполнить асимптотически $C_1 k^2 m + C_2 k^2 + C_3 \frac{k^2}{s}$ операций.

Если заранее вычислить таблицу U_m, то количество операций во время вычислений снизится до $C_3 \frac{k^2}{s}$, правда, необходимо будет помнить таблицы $U_m, \quad m = 1, 2, \ldots, k-1$, что потребует сохранения асимптотически k^3/s чисел.

Ниже мы покажем, как можно сократить объем требуемой памяти, уменьшив скорость работы алгоритма. Эта скорость все же будет больше, чем в случае, когда вычисления каждый раз начинаются с построения таблицы $U(T)$.

Пусть c – некоторое натуральное число (параметр), значение которого будет выбрано позже. Через m_0 обозначим число $]k/c[$. Заранее вычислим таблицы $U_m, \quad m = 1, 2, \ldots, m_0$. Объем требуемой памяти для запоминания массива $U_m, \quad m = 1, 2, \ldots, m_0$ асимптотически в c раз меньше, чем для запоминания массива $U_m, \quad m = 1, 2, \ldots, k-1$.

Пусть нужно вычислить x^{2^n} в полиномиальном базисе. Если $n \geq k$, то найдем остаток n_1 от деления n на k,

$$n = qk + n_1, \quad 0 \leq n_1 < k.$$

Ввиду эквивалентности $x^{2^n} \equiv x^{2^{n_1}} \mod p(x)$, нужно вычислить $x^{2^{n_1}}$.

Таким образом, в дальнейшем достаточно указать способ вычисления многочлена x^{2^n} при $n < k$.

Если $n \leq m_0$, то искомый многочлен получим умножая вектор-строку, соответствующую $f(x)$ на U_n. При этом будет использовано асимптотически $C_3 \frac{k^2}{s}$ операций.

В противном случае ($n > m_0$) через r обозначим остаток от деления числа n на m_0. тогда для некоторого натурального числа q_1 выполнено $n = q_1 m_0 + r$.

Пусть $f_i(x), \quad i = 0, 1, \ldots, q_1, q_1 + 1$ – последовательность многочленов, представленных в полиномиальном базисе и полученных по следующей схеме.

$$f_0(x) = f(x),$$

$$f_i(x) = f_{i-1}^{2^{m_0}}(x), \quad i = 1, 2, \ldots, q_1.$$

$$f_{q_1+1}(x) = f_{q_1}^{2^r}(x).$$

Используя эту намеченную в начале данного параграфа схему, многочлен $f^{2^n}(x) = f_{q_1+1}(x)$ можно получить, выполнив асимптотически не более

$$C_3(q_1 + 1)\frac{k^2}{s} \leq C_3\left(\frac{n}{m_0} + 1\right)\frac{k^2}{s} \asymp C_3\frac{nkc}{s}$$

операций.

Следовательно, выбрав значение параметра $c = [\sqrt{s}]$, получаем, что для возведения многочлена $f(x)$ в степень 2^n потребуется асимптотически $C_3\frac{nk}{\sqrt{s}}$ операций.

Пример 3.6.4 Пусть $k = 5, \quad p(x) = x^4 + x + 1, \quad f(x) = x^3 + 1, \quad m_0 = 2$ и требуется найти $f^{2^3}(x)$, выразив его в полиномиальном базисе.

Имеем $n = 3 = q \cdot m_0 + r$. Построим в полиномиальном базисе таблицы $Q_{m_0} = Q_2$ и $Q_r = Q_1$.

Для построения таблицы Q_{m_0} представим в полиномиальном базисе одночлены $\{1, x^4, x^8, x^{12}\}$. Одночлен 1 преобразовывать не нужно.

$$x^4 \equiv 1 + x((\mathrm{mod} p(x)),$$

$$x^8 = \left(x^4\right)^2 \equiv 1 + x^2((\mathrm{mod} p(x)),$$

$$x^{12} \equiv \left(x^6\right)^2 \equiv \left(x^2 + x^3\right)^2 \equiv x^4 + x^6 \equiv x^3 + x^2 + x + 1((\mathrm{mod} p(x)).$$

Чтобы получить таблицу Q_r представим в полиномиальном базисе одночлены $\{1, x^2, x^4, x^6\}$. Первые два одночлена из этого множества преобразовывать не нужно, одночлен x^4 уже преобразован.

$$x^6 \equiv x^2 + x^3((\mathrm{mod} p(x)).$$

$$Q_{m_0} = Q_2 = \begin{array}{|cccc|c|} \hline x^3 & x^2 & x & x^0 & \\ \hline 0 & 0 & 0 & 1 & x^0 \\ 0 & 0 & 1 & 1 & x^4 \\ 0 & 1 & 0 & 1 & x^8 \\ 1 & 1 & 1 & 1 & x^{12} \\ \hline \end{array} \quad O_r = Q_1 = \begin{array}{|cccc|c|} \hline x^3 & x^2 & x & x^0 & \\ \hline 0 & 0 & 0 & 1 & x^0 \\ 0 & 1 & 0 & 0 & x^2 \\ 0 & 0 & 1 & 1 & x^4 \\ 1 & 1 & 0 & 0 & x^6 \\ \hline \end{array}.$$

Так как $3 = 1 \cdot m_0 + 1$, то вектор $(1, 0, 0, 1)$ коэффициентов многочлена $f(x)$ нужно сначала умножить на матрицу Q_2. Получится вектор $(1, 1, 1, 0)$, который соответствует многочлену $x^3 + x^2 + x$. Поэтому для того, чтобы получить искомый результат, нужно вектор $(0, 1, 1, 1)$ (полученный вектор "переворачивается") умножить на матрицу Q_1. (Очевидно, при имплементации лучше предварительно переставить столбцы матрицы Q_r.) Получим вектор $(1, 0, 1, 1)$, который соответствует многочлену $1 + x + x^3$. Таким образом,

$$f^{2^3}(x) \equiv 1 + x + x^3(\mathrm{mod} p(x).$$

3.6.4 Быстрое инвертирование в конечном поле малой характеристики с использованием полиномиального базиса

Как известно, можно использовать для вычисления мультипликативного обратного в поле $GF(p^n)$ расширенный алгоритм Евклида. Время его работы возможно будет сильно зависеть от выбранного для обращения многочлена.

Можно использовать также теоретически более быстрый вариант Шенхаге-Моенка алгоритма Евклида, который дает в наихудшем случае оценку $O(M(n)\log n)$, где $M(n)$ — сложность умножения многочленов степени n

Известно также, что можно выполнять инвертирование в поле $GF(p^n)$ с помощью тождества Ферма $f^{-1} \bmod q = f^{p^n-2} \bmod q$. Используя для возведения в степень указанные выше соображения, можно получить для этого алгоритма оценку сложности $O(M(n)\log n + n^2)$. Остаточный член $O(n^2)$ можно несколько уменьшить, заменив его на $O(n^2/\sqrt{\log n})$ при условии предварительного вычисления некоторых матриц.

Для этого заметим, что для возведении f в (p^n-2)-ю степень приходится возводить в самом конце сразу в степень $p^{n/2+O(1)}$, до этого — в степень $p^{n/4+O(1)}$, и так далее. Если выполнять эти операции так как указано в предыдущем пункте, то суммарная сложность этих операций будет равна $\frac{O(n^2)}{\sqrt{\log n}}$ (сложность предварительных вычислений не учитываем), сложность оставшихся умножений оценивается также, как и раньше.

3.7 Быстрое умножение и экспоненцирование

3.7.1 Быстрое умножение чисел и многочленов

Первым придумал быстрый алгоритм умножения в 1962 г. А.А.Карацуба [38].

Идею его метода можно пояснить на следующем примере. Пусть перемножаются восьмизначные числа $U = \overline{u_1 \dots u_8}$ и $V = \overline{v_1 \dots v_8}$. Представим их как двузначные числа в 10^4-значной системе счисления: $U = U_1U_2,\ V = V_1V_2$. Тогда их произведение можно представить в следующем виде:

$$UV = U_1V_1 10^8 + ((U_1 - U_2)(V_2 - V_1) + U_1V_1 + U_2V_2)10^4 + U_2V_2.$$

Эта формула сводит умножение 8-значных чисел к трем операциям умножения и шести операциям сложения-вычитания 4-значных чисел (с учетом переносов в следующие разряды). Обычный способ требует четырех умножений и трех сложений-вычитаний, но так как три раза сложить 4-значные числа можно быстрее, чем один раз перемножить, то метод Карацубы уже 8-значные числа перемножает быстрее. В общем случае он требует для перемножения n-значных

чисел по порядку не больше

$$n^{\log_2 3} < n^{1,585}$$

операций над цифрами, для школьного же метода требуется по порядку n^2 операций.

Рассмотрим вопрос о сложности умножения более подробно. Начнем с умножения многочленов.

Лемма 3.7.1 *Умножение двух многочленов степеней меньших $2n$ можно свести к умножению трех пар многочленов степеней меньших n и сложению четырех пар многочленов степеней меньших n и вычитанию двух пар многочленов степеней меньших $2n$ с помощью тождества*

$$(f_1x^n + f_0)(g_1x^n + g_0) =$$

$$= f_1g_1x^{2n} + ((f_1 + f_0)(g_1 + g_0) - f_1g_1 - f_0g_0)x^n + f_0g_0.$$

Обозначим $M(n)$ наименьшее количество операций сложения, вычитания и умножения (выполняемых над коэффициентами многочленов и промежуточными числовыми результатами), требующихся для перемножения двух многочленов степеней меньших 5.

Лемма 3.7.2 *Справедливы неравенства*

$$M(n5 \leq 2M(\lceil n/2 \rceil) + M(\lfloor n/2 \rfloor) + 4\lfloor n/2 \rfloor + 2n - 4.$$

Доказательство. Применим равенство

$$\left(f_1x^{\lfloor n/2 \rfloor} + f_0\right)\left(g_1x^{\lfloor n/2 \rfloor} + g_0\right) =$$

$$= f_1g_1x^{2\lfloor n/2 \rfloor} + ((f_1 + f_0)(g_1 + g_0) - f_1g_1 - f_0g_0)x^{\lfloor n/2 \rfloor} + f_0g_0,$$

где степени многочленов f_1 и g_1 меньше $\lceil n/2 \rceil$, а степени многочленов f_0 и g_0 меньше $\lfloor n/2 \rfloor$ и заметим, что для вычисления произведений $f_1g_1,\ f_0g_0$ требуется не более $M(\lceil n/2 \rceil) + M(\lfloor n/2 \rfloor)$ операций, для вычисления сумм $f_1 + f_0$, $g_1 + g_0$, $f_1g_1 + f_0g_0$ нужно не более $2\lfloor n/2 \rfloor + 2\lfloor n/2 \rfloor - 1$ операций (так как число операций равно наименьшему из количеств ненулевых коэффициентов у складываемых многочленов), для вычисления произведения $(f_1 + f_0)(g_1 + g_0)$ используется не более $M(\lceil n/2 \rceil)$ операций, для вычисления разности

$$(f_1 + f_0)(g_1 + g_0) - f_1g_1 - f_0g_0$$

достаточна $n - 1$ операция , так как

$$(f_1 + f_0)(g_1 + g_0) - f_1g_1 - f_0g_0 = f_1g_0 + f_0g_1,$$

значит степень этого многочлена равна $\lfloor n/2 \rfloor + \lceil n/2 \rceil - 2 = n - 2$, сложение многочленов $f_0 g_0$ и $f_1 g_1 x^{2\lfloor n/2 \rfloor}$ выполняется "бесплатно", так как они не имеют подобных членов, причем в их сумме отсутствует член вида $x^{2\lfloor n/2 \rfloor - 1}$, поэтому для сложения многочленов

$$f_0 g_0 + f_1 g_1 x^{2\lfloor n/2 \rfloor}$$

и

$$\left(f_1 g_0 + f_0 g_1\right) x^{\lfloor n/2 \rfloor}$$

достаточно $n - 2$ операции. В итоге требуется дополнительно $4\lfloor n/2 \rfloor + 2n - 4$ операции.

Теорема 3.7.1 *При $2^k \mid n$ справедливо неравенство*

$$M(n) \leq 3^k \left(M\left(n/2^k\right) + 8n/2^k - 2\right) - 8n + 2,$$

а при любом n - неравенство

$$M(n) < (35/3)\, n^{\log_2 3}.$$

Доказательство.

Пусть $2^k m = n$. Тогда неравенство

$$M(n) \leq 3^k \left(M(m) + 8m - 2\right) - 8n + 2$$

доказывается индукцией по k. База ($k = 1$) доказана в лемме 2. Шаг индукции обосновывается тем же неравенством.

Выберем k так, чтобы $2^k < n \leq 2^{k+1}$. Тогда если $3 \cdot 2^{k-1} < n$, то

$$M(n) \leq M\left(2^{k+1}\right) < 3^{k-1}\left(M(4) + 30\right) \leq 3^{k-1} \cdot 55 <$$

$$< 55 \cdot \left(\frac{n}{3}\right)^{\log_2 3} < \frac{35}{3} n^{\log_2 3}.$$

Если же $n \leq 3 \cdot 2^{k-1}$, то

$$M(n) \leq M\left(3 \cdot 2^{k-1}\right) < 3^{k-1}\left(M(3) + 22\right) \leq 3^{k-1} \cdot 35 \leq (35/3)\, n^{\log_2 3}.$$

Перейдем теперь к умножению чисел.

Обозначим $M(n)$ наименьшее количество операций сложения, вычитания и умножения, выполняемых над числами, меньшими a, требующихся для перемножения двух n - значных чисел, записанных в позиционной системе счисления по основанию a.

Лемма 3.7.3 *Справедливы неравенства*

$$M(2n) \leq 3M(n) + 19n, M(2n+1) \leq 2M(n+1) + M(n) + 17n + 10.$$

Доказательство. Применим тождества

$$\left(f_1 b^{\lceil n/2\rceil}+f_0\right)\left(g_1 b^{\lceil n/2\rceil}+g_0\right)=$$

$$=f_1g_1b^{2\lceil n/2\rceil}+\left(f_1g_1+f_0g_0-\left(f_1-f_0\right)\left(g_1-g_0\right)\right)b^{\lceil n/2\rceil}+f_0g_0,$$

где числа f_1 и g_1 — $\lfloor n/2\rfloor$-разрядные, а числа f_0 и g_0 соответственно $\lceil n/2\rceil$ - разрядные и заметим, что для вычисления произведений f_1g_1 и f_0g_0 требуется $M\left(\lceil n/2\rceil\right)+M\left(\lfloor n/2\rfloor\right)$ операций, для вычисления разностей и суммы

$$f_0-f_1,g_0-g_1,f_1g_1+f_0g_0$$

требуется не более

$$n\left(1+\lfloor n/2\rfloor-\lceil n/2\rceil\right)+2\left(\lfloor n/2\rfloor+\lceil n/2\rceil-1\right)+2\left(\lfloor n/2\rfloor+\lceil n/2\rceil\right)-1=$$

$$=4n-3+n\left(1+\lfloor n/2\rfloor-\lceil n/2\rceil\right)$$

операций, так как числа f_1g_1 и f_0g_0 имеют не более чем $2\lfloor n/2\rfloor$ и $2\lceil n/2\rceil$ разрядов соответственно, а в случае четного n нужно еще $2\lfloor n/2\rfloor=n$ операций для предварительного сравнения чисел (чтобы не вычитать из меньшего большее). Заметим далее, что для вычисления произведения $\left(f_1-f_0\right)\left(g_1-g_0\right)$ требуется не более $M\left(\lceil n/2\rceil\right)+1$ операций (одна операция для вычисления знака у произведения), для вычисления разности

$$f_1g_1+f_0g_0-\left(f_1-f_0\right)\left(g_1-g_0\right)=f_1g_0+f_0g_1$$

требуется не более $2\lceil n/2\rceil+1+2\lceil n/2\rceil-1=4\lceil n/2\rceil$ операций, сложение чисел f_0g_0 и $f_1g_1b^{2\lceil n/2\rceil}$ осуществляется «бесплатно» (записи этих чисел просто объединяются в одну запись), а для сложения чисел $f_1g_1b^{2\lceil n/2\rceil}+f_0g_0$ и $\left(f_1g_0+f_0g_1\right)b^{\lceil n/2\rceil}$ требуется не более $2n-\lceil n/2\rceil+n+1-1=2n+\lfloor n/2\rfloor$ операций (так как число $f_1g_0+f_0g_1$ имеет не более $n+1$ разряда, а младшие $\lfloor n/2\rfloor$ разрядов числа f_0g_0 не участвуют в операциях) В итоге требуется дополнительно

$$4n-3+n\left(1+\lfloor n/2\rfloor-\lceil n/2\rceil\right)+1+4\lceil n/2\rceil+2n+\lfloor n/2\rfloor=$$

$$=7n+3\lceil n/2\rceil+n\left(1+\lfloor n/2\rfloor-\lceil n/2\rceil\right)-2$$

операций.

Упражнение 3.7.1 Проверьте, что обычный способ умножения многочленов дает оценку

$$M\left(n\right)\leq M_s\left(n\right)=n^2+\left(n-1\right)^2.$$

Упражнение 3.7.2 Для умножения многочленов найдите наименьшее n, при котором

$$M\left(n\right)<M_s\left(n\right).$$

Упражнение 3.7.3 Докажите, что сложение двух n-значных чисел можно выполнить за $2n-1$ операцию, вычитание из большего меньшее - за $3n-1$ операцию, вычитание с определением знака разности - за $4n-1$ операцию, а сложение $n+m$-значного числа с n-значным можно выполнить за $2n+m-1$ операцию.

Упражнение 3.7.4 Докажите, что обычный способ умножения чисел дает оценку

$$M(n) \leq M_s(n) < 5n^2.$$

Обозначим через $K(n)$ сложность возведения n-разрядного числа в квадрат и такое же обозначение будем использовать для сложности возведения в квадрат многочлена степени $n-1$.

Упражнение 3.7.5 Используя тождество

$$ab = \frac{(a+b)^2 - (a-b)^2}{4},$$

докажите для случая операций с числами неравенство

$$M(n) \leq 2K(n) + 13n + O(1),$$

а для случая операций с многочленами — неравенство

$$M(n) \leq 2K(n) + 6n + 4.$$

Использование этого трюка не ускоряет вычисления, но позволяет экономить используемую память.

Упражнение 3.7.6 Модифицируйте алгоритм Карацубы для случая возведения чисел в квадрат и покажите, что

$$K(2n) \leq 3K(n) + 16n - 2,\ K(2n+1) \leq 2K(n+1) + K(n) + 15n + 10.$$

В случае операций с многочленами получается оценка

$$K(n) \leq 2K(\lceil n/2 \rceil) + K(\lfloor n/2 \rfloor) + 3\lfloor n/2 \rfloor + 2n - 4.$$

Таким образом, при возведении в квадрат методом Карацубы можно достичь дополнительного ускорения примерно на 10 процентов.

Упражнение 3.7.7 Докажите, что сложность деления с остатком многочлена степени m на многочлен степени n со старшим коэффициентом 1 оценивается как $(m-n+1)(2n+1)$, а без остатка — как $(m-n)(2n+1)$.

Упражнение 3.7.8 Докажите, что обычный алгоритм Евклида для многочленов степени не выше n имеет оценку сложности

$$GD(n) < 5n^2/2 + 3n/2.$$

Упражнение 3.7.9 Докажите, что предыдущую оценку нельзя существенно улучшить, рассмотрев вычисление (T_n, T_{n-1}), где T_n — многочлены Чебышева.

Упражнение 3.7.10 Докажите, что сложность деления с остатком $n-$разрядного числа на $m-$ разрядное имеет оценку сложности $C(m-n)n$, где $C > 0$ — константа.

Упражнение 3.7.11 Докажите, что обычный алгоритм Евклида для $n-$разрядных чисел имеет оценку сложности

$$GD(n) \leq Cn^2.$$

Упражнение 3.7.12 Докажите, что обычный алгоритм Евклида для многочленов степени не выше n имеет оценку сложности

$$5(n^2 - m^2)/2 + O(n),$$

где m — степень вычисленного .

3.7.2 О возможности применения дискретного преобразования Фурье (ДПФ) для умножения многочленов над полем $GF(2)$. Сравнение с методом Карацубы и школьным методом.

Для умножения многочленов известен асимптотически быстрый алгоритм, использующий быстрое дискретное преобразование Фурье (ДПФ). Мы кратко опишем его применение (более подробное изложение см., например, в [5] или [53]) и убедимся, что он будет эффективен только при достаточно больших степенях перемножаемых многочленов (порядка нескольких тысяч).

Представим многочлены-сомножители в виде

$$\sum_{i=0}^{n/8-1} f_i X^i, \ \sum_{i=0}^{n/8-1} g_i X^i, \ X = x^8,$$

где f_i, g_i— многочлены степени 7 над полем $GF(2)$ и выполним умножение так, как будто перемножаются многочлены степени $n/8-1$ с коэффициентами из кольца многочленов над полем $GF(2)$, при этом коэффициенты произведения будут многочленами степени 14, и чтобы привести результат к окончательному виду нужно каждый из них представить в виде $h_i = h_{i,0} + h_{i,1}X, X = x^8$, где $h_{i,j}$— многочлены степени 7 над полем $GF(2)$, и вычислить коэффициенты c_i по формулам $h_{i,0} + h_{i-1,1}$.

Коэффициенты многочленов степени $n/8-1$ будем хранить в виде 8-битовых целых чисел. Операцию умножения многочленов степени 7 можно заранее затабулировать. В результате умножения двух многочленов степени 7 получается многочлен степени 14, который представляем в виде 16-битового целого числа. Операция сложения многочленов выполняется одной командой побитового сложения по модулю два целых чисел.

Выберем неприводимый многочлен $p(x)$ 8-й степени над полем $GF(2)$ и будем рассматривать исходные многочлены как многочлены с коэффициентами из кольца многочленов по модулю p, т. е. из поля $GF(2^8)$, тогда коэффициенты произведения упомянутых исходных многочленов тоже будут принадлежать указанному полю и будут получаться из многочленов h_i приведением по модулю p (т.е. заменой на остаток от деления на многочлен p).

Применим метод умножения многочленов с помощью ДПФ. Напомним, что ДПФ F_d порядка d определяется как линейное преобразование, которое переводит вектор f коэффициентов многочлена степени $d-1$

$$f(x) = \sum_{l=0}^{d-1} f_l x^l$$

над произвольным заданным полем GF в вектор g той же размерности d с компонентами

$$G_K = F_d(k)[f] = \sum_{l=0}^{d-1} f_l \omega_d^{k \cdot l},$$

где ω_d - первообразный корень из единицы степени d, принадлежащий полю GF (первообразным корнем степени d называется такой элемент поля, у которого d-я степень равна единице, и никакая степень меньшего положительного порядка единице не равна). Известно (см., например, [10] или [53]), что обратное преобразование F_d^{-1} восстанавливает вектор f по вектору g с помощью формул

$$f_k = F_d^{-1}(k)[f] = \frac{1}{d}\sum_{l=0}^{d-1} g_l \omega_d^{-k\cdot l}.$$

Это преобразование в случае полей GF, порядок которых равен степени двух, существует только при нечетном d, и тогда в указанных полях $\frac{1}{d}$ просто равно 1, поэтому обратное ДПФ отличается от прямого ДПФ только заменой первообразного корня ω_d на первообразный корень $\omega_d^{-1} = 1/\omega_d$.

Вычисление произведения многочленов степени $n/8-1$ с коэффициентами из поля $GF(2^8)$ можно выполнить следующим образом. Добавляем к каждому из многочленов $n/8-1$ старших членов с нулевыми коэффициентами, так чтобы эти многочлены формально стали многочленами степени $n/4-2$ (не изменившись по существу). К векторам коэффициентов полученных многочленов применяем ДПФ над полем $GF(2^8)$ порядка $m = n/4-1$ и получаем два вектора V и W размерности m с компонентами из поля $GF(2^8)$. Далее вычисляем их покомпонентное произведение $U = V \otimes W$, определяемое формулами $U_i = V_i \cdot W_i, i = 0, \ldots, m-1$ и к полученному вектору применяем обратное ДПФ над полем $GF(2^8)$. Найденный в результате выполнения этого преобразования вектор будет совпадать с вектором коэффициентов искомого произведения заданных многочленов (подробности см. в [5] или [53]).

Все указанные процедуры будут выполнимы, только если m нечетно, и в поле $GF(2^8)$ существует первообразный корень из единицы порядка m. Последнее справедливо при m, делящем $2^8-1 = 255$, так как в качестве такого корня можно взять степень порядка $255/m$ примитивного элемента рассматриваемого поля, ведь примитивный элемент поля $GF(2^8)$ - это первообразный корень из единицы порядка 2^8-1 в этом поле (о существование примитивных элементов в конечных полях см., напр. [53]). Полная сложность всех указанных процедур равна трехкратной сложности вычисления ДПФ порядка m плюс m операций умножения в поле $GF(2^8)$. Отметим, что операцию умножения в поле $GF(2^8)$ можно заранее затабулировать, и поэтому ее сложность далее считается единичной.

После выполнения указанного умножения многочленов в качестве коэффициентов произведения мы находим не нужные нам многочлены h_i, а только их остатки по заданному модулю p. Но если повторить указанную операцию при другом выборе неприводимого многочлена, например q, мы найдем остатки от тех же многочленов по модулю q, и восстановим эти многочлены с помощью китайской теоремы об остатках (о китайской теореме об остатках см. [53] или [5]).

Для этого надо взять остатки r_1 и r_2 по модулям p и q, найти такие многочлены p_1 и q_1 степени 7, что $p \cdot p_1 = 1 \bmod q, q \cdot q_1 = 1 \bmod p$, и вычислить искомый многочлен по формуле $h = r_1 \cdot q \cdot q_1 + r_2 \cdot p \cdot p_1 \bmod pq$. Действительно,

$$h \bmod q = r_2 \cdot p \cdot p_1 \bmod q = r_2, h \bmod p = r_1 \cdot q \cdot q_1 \bmod p = r_1.$$

Многочлены qq_1 и pp_1 зависят только от p и q и их можно вычислить заранее. Произведения r_1qq_1 и r_2pp_1 вычисляются за шесть операций сложения и умножения многочленов степени 7, которые можно не реализовывать программно, а взять из заранее введенной в память машины таблицы умножения. После этого с помощью трех таких же операций сложения получаем многочлен $r_1qq_1 + r_2pp_1$ степени 22.

Для того, чтобы найти вычет многочлена $r_1qq_1 + r_2pp_1$ по модулю pq, заранее вычислим такой многочлен R степени 6, что многочлен $H = Rpq + x^{22}$ имеет степень не выше 15, (т. е. разделим одночлен x^{22} на многочлен pq с остатком, при этом частное будет равно R, а остаток будет равен $H = x^{22} - Rpq = Rpq + x^{22}$, так как в рассматриваемом поле вычитание совпадает со сложением). Тогда для нахождения многочлена $r_1qq_1 + r_2pp_1 \bmod pq$ нужно будет вначале вычислить многочлен 6-й степени Q, коэффициенты которого совпадают со старшими разрядами произведения $(r_1qq_1 + r_2pp_1)R$, для чего достаточно одной операции умножения многочленов 7-й степени.

Действительно, многочлен Q равен целой части алгебраической дроби

$$\frac{(r_1qq_1 + r_2pp_1)R}{x^{22}} = \frac{(r_1qq_1 + r_2pp_1)(H + x^{22})}{pqx^{22}} =$$

$$= \frac{(r_1qq_1 + r_2pp_1)H}{pqx^{22}} + \frac{(r_1qq_1 + r_2pp_1)}{pq},$$

откуда видно, что он также равен целой части дроби

$$\frac{(r_1qq_1 + r_2pp_1)}{pq},$$

так как дробь

$$\frac{(r_1qq_1 + r_2pp_1)H}{pqx^{22}}$$

правильная (ведь степень ее числителя не больше 7+8+7+15, а степень знаменателя равна 8+8+22), т. е. он также равен частному от деления многочлена $r_1qq_1 + r_2pp_1$ на многочлен pq.

Поэтому для вычисления искомого остатка по модулю pq достаточно вычислить $r_1qq_1 + r_2pp_1 + pqQ$, для чего нужно еще 2 арифметические операции с многочленами 7-й степени.

В результате общее число элементарных операций будет равно $5n/2$ плюс $n/2$ умножений в поле $GF(2^8)$ плюс шестикратную сложность преобразования

Фурье порядка $m = n/4 - 1$ в этом же поле. Кроме того, нужно с помощью $2m$ операций сдвига каждый из m 16-битовых коэффициентов произведения многочленов разбить на два 8-битовых числа и с помощью m операций побитового сложения по модулю 8-битовых чисел вычислить окончательно коэффициенты произведения многочленов.

Значит, сложность умножения многочленов равна $15n/4 + 6F(m)$, где $F(m)$ — сложность ДПФ порядка m в поле $GF(2^8)$.

Оценим ее при $m = 255$. Согласно теореме Гуда-Томаса (см. [10] или [53]) имеем

$$F(255)/255 = F(3)/3 + F(5)/5 + F(17)/17.$$

Очевидно, что $F(m) = (2m-1)(m-1)$ при тривиальном способе вычисления преобразования Фурье, откуда $F(3) \leq 10, F(5) \leq 36, F(17) \leq 528$. Так как преобразование Фурье третьего порядка выполняется по формулам

$$f_1 = x_1 + (x_2 + x_3), f_2 = x_1 + x_2\omega + x_3\omega^2 = x_1 + (x_2 + x_3)\omega + x_3,$$

$$f_3 = x_1 + x_2\omega^2 + x_3\omega^4 = x_1 + x_2(\omega + 1) + x_2\omega = x_1 + (x_2 + x_3)\omega + x_2, \omega \in GF(2^2),$$

очевидно $F(3) \leq 7$. Использования алгоритма Блюстейна [10] при нечетных m позволяет уменьшить число умножений до $m(m-1)/2$. Поэтому $F(5) \leq 30$.

Согласно теореме Райдера (см., напр. [10]) при простом m справедливо неравенство $F(m) \leq C(m-1) + 2(m-1)$, где $C(m-1)$ — сложность циклической свертки порядка $m-1$, то есть сложность умножения многочленов степени $m-2$ по модулю $x^{m-1}+1$. Циклическая свертка четного порядка m может быть вычислена по формулам Карацубы

$$(a_0 + a_1x^{m/2})(b_0 + b_1x^{m/2}) \bmod x^m + 1 =$$

$$= a_0b_0 + a_1b_1 + ((a_0 + b_0)(a_1 + b_1) + a_0b_0 + a_1b_1)x^{m/2} \bmod x^m + 1 =$$

$$= a_0b_0 + a_1b_1 + c_1 + c_0x^{m/2},$$

где

$$(a_0 + a_1)(b_0 + b_1) + a_0b_0 + a_1b_1 = c_0 + c_1x^{m/2},$$

a_i, b_i, c_i— многочлены степени $m/2-1$. Из них вытекает следующая оценка сложности циклической свертки

$$C(m) \leq 3M(m/2) + 4m - 4,$$

где $M(n)$ — сложность умножения многочленов степени $n-1$.

Оценка Карацубы сложности умножения многочленов четной степени, основанная на формуле

$$(a_0 + a_1x^{m/2})(b_0 + b_1x^{m/2}) =$$

$$= a_0b_0 + a_1b_1x^m + ((a_0 + a_1)(b_0 + b_1) + a_0b_0 + a_1b_1)x^{m/2},$$

также имеет вид $M(m) \leq 3M(m/2)+4m-4$ (см.[25],[101]). Применяя ее, имеем $M(8) \leq 3M(4)+28 = 3 \cdot 25+28 = 103$, откуда следует, что $C(16) \leq 3M(8)+60 \leq 369$, значит $F(17) \leq C(16)+32 \leq 401$. Алгоритм Блюстейна дает общую оценку числа операций 408 и даже меньшее число умножений, причем эти умножения выполняются только на 24 конкретных скаляра из поля $GF(2^4)$. Методом Карацубы в рассматриваемом случае можно добиться уменьшения числа умножений до 81, но при этом общее число операций возрастет до 473.

Учитывая полученные неравенства, имеем оценку

$$F(255) \leq 255\left(\frac{F(3)}{3}+\frac{F(5)}{5}+\frac{F(17)}{17}\right)=$$

$$=85 \cdot 7+51 \cdot 30+15 \cdot 401 = 8140.$$

В результате $M(1024) \leq 15 \cdot 255+6F(255) = 52665$. Если везде использовать алгоритм Блюстейна, то оценка остается почти такой же, но во всем алгоритме умножения будут использоваться только не более $2n = 2048$ умножений многочленов степени 7 и умножений в поле $GF(2^8)$, а все остальные умножения в этом поле будут выполняться на фиксированные $24+6 = 30$ скаляров из этого поля.

Оценка сложности умножения двух многочленов степени $n = 1024$ указанной выше модификацией стандартного алгоритма равна $(n/8)^2+(n/8-1)^2+2(n/4-1)+n/4-1 = 33790$. Значит это алгоритм в рассматриваемом случае работает быстрее.

Еще быстрее работает алгоритм Карацубы. Оценка его сложности имеет вид $2(n/4-1)+n/4-1+M(n/8) = 765+M(128)$, а $M(128) \leq 3M(64)+508, M(64) \leq 3M(32)+252, M(32) \leq 3M(16)+124, M(16) \leq 3M(8)+60 = 369$, откуда $M(128) \leq 12343$, в итоге окончательно имеем оценку 13108.

Замечание 7.1 Степень перемножаемых указанными выше методами многочленов можно было бы увеличить в два раз практически без уменьшения скорости работы алгоритмов, если его выполнять, используя операции с многочленами 15−й степени и операции в поле $GF(2^{16})$, однако операции умножения затабулировать не удается в виду слишком большого размера таблиц. Поэтому школьный алгоритм и алгоритм Карацубы ускорить таким способом (по отношению к размерам операндов) не удается. Поэтому при двукратном увеличении степени время работы первого из них возрастет в четыре, а второго — примерно в три раза. Некоторого их ускорения можно добиться, используя другой подход, изложенный далее.

Но для алгоритма, использующего ДПФ, это возможно. Действительно, если затабулировать таблицу дискретных логарифмов для поля $GF(2^{16})$ (что требует 128 Кб памяти), то умножение в нем сводится к сложению по модулю $2^{16}-1$ и трехкратному обращению к загруженной в память таблице. Конечно эта операция в несколько раз медленнее, чем сложение по модулю два, но число умножений в алгоритме в несколько раз меньше числа сложений. Умножение

многочленов 15−й степени таким образом затабулировать не удается, но эта операция используется только $3n/4$ раз и если ее свести к четырехкратному умножению и двукратному сложению многочленов степени 7, то скорость работы алгоритма практически не уменьшится. Есть и другая возможность для ускорения операции умножения. Если использовать для вычисления ДПФ только комбинацию методов Гуда-Томаса и Блюстейна, то, как уже отмечалось, большинство умножений выполняется на небольшое количество скаляров (в рассматриваемом примере 32). Поэтому для их выполнения достаточно иметь таблицу размером $2^{16} \cdot 4 = 256$ Кб и пользоваться ей для каждого умножения однократно.

Замечание 7.2 Для n, кратных 8, можно использовать другое представление элементов поля $GF(2^n)$, а именно в виде многочленов степени $n/8-1$ над полем $GF(2^8)$, и операцию умножения проводить тоже по модулю неприводимого многочлена степени $n/8-1$ над тем же полем. Тогда операцию умножения многочленов над этим полем можно выполнять также, как описано в предыдущем разделе, причем не надо его дублировать для другой реализации поля $GF(2^8)$. В итоге сложность умножения будет оцениваться формулой $3F(m)$, а не $15n/4 + 6F(m)$, причем в качестве операций умножения многочленов будут использоваться операции умножения в поле $GF(2^8)$, которые можно просто затабулировать также, как указано было выше. Тогда оценка сложности умножения имеет вид $3F(255) = 24420$ и будет лучше школьной, но хуже оценки Карацубы. Еще некоторого ускорения можно достичь, если воспользоваться приемом, указанным в предыдущем замечании.

Замечание 7.3. Использование ДПФ позволяет ускорить умножение только в очень больших полях. Но это верно только для полей характеристики 2, так как в них нельзя использовать ДПФ порядка 2^k, для которых известны наиболее быстрые алгоритмы. Для полей нечетной характеристики перспективы более оптимистичные. Пусть например, надо реализовать умножение в поле $GF(31^{160})$. Если рассматривать его как расширение поля $GF(31^2)$, то для этого достаточно реализовать умножение многочленов степени 79 над полем $GF(31^2)$. Операции над этим полем можно считать элементарными, так как их можно затабулировать и хранить в памяти (использовать поле порядка 31^4 уже затруднительно, так как объем таблицы умножения будет несколько мегабайт, но можно и здесь применить тот же прием, что и в предыдущих замечаниях). Число операций в стандартном алгоритме умножения больше 12000, число операций в алгоритме Карацубы равно 5761. Умножение с помощью ДПФ требует $3F(160) + 320$ операций. Величину $F(160)$ можно с помощью методов Гуда-Томаса и Блюстейна как

$$5F(32) + 32F(5) = 5 \cdot 3 \cdot 16 \cdot 5 + 32 \cdot 30 = 2160.$$

В результате получается чуть худшая, чем в методе Карацубы оценка, но уже для поля $GF(31^{320})$ получаем оценку $3F(320) + 640$, где $F(320) = 5F(64) +$

$64F(5) = 5 \cdot 576 + 64 \cdot 30 = 4800$, и в результате полное число операций около 15000. Метод Карацубы дает оценку чуть больше 18000.

sectionОб асимптотически быстрых методах умножения многочленов.

Стандартная схема быстрого умножения многочленов с действительными коэффициентами, основанная на циклической теореме о свертке (см., например, [53], [10]) такова: если нужно перемножить два многочлена степени $n-1$, то сначала применяют дискретное преобразование Фурье (ДПФ) порядка $2n$ к двум $2n$-мерным векторам, образованным коэффициентами многочленов и дополненным каждый n нулями, потом полученные два $2n$-мерных вектора покомпонентно перемножаются, и к полученному в результате $2n$-мерному вектору применяется обратное преобразование Фурье порядка $2n$ (подробное изложение имеется, также, например в [5]). Для выполнения ДПФ порядка $2n$ требуется использование первообразного корня порядка $2n$ из единицы, поэтому необходимо использование поля комплексных чисел (или подходящего расширения поля коэффициентов перемножаемых многочленов в общем случае). Выполнение ДПФ в случае n, равного степени двойки, по существу эквивалентно вычислению методом деления пополам (называемого также «разделяй и властвуй»[5] или «стратегия дублирования» [10]) множества значений многочлена во всех n корнях порядка n из единицы, или, что равносильно, нахождению всех остатков от деления этого многочлена на линейные двучлены, получающиеся в результате разложения на множители (в поле комплексных чисел) двучлена $x^n - 1$. Об этом можно прочитать, например, в [5], где этот метод применен также в более общей ситуации, а именно в быстрой реализации вычисления системы остатков данного многочлена по данным взаимно простым модулям, и восстановления многочлена по эти остаткам («быстрый китайский алгоритм»). В [22] показано, как при умножении многочленов можно вообще обойтись без ДПФ, заменив его применение использованием «быстрого китайского алгоритма» (при этом применение обратного преобразования Фурье маскируется под быструю интерполяцию многочлена в корнях порядка n из единицы и не используется ни формула для обратного ДПФ, ни теорема о циклической свертке). О других алгоритмах БПФ (быстрого преобразования Фурье) и их применениях см. также [20], [48], [10].

В реальных вычислениях операции с комплексными числами заменяются на операции с действительными числами и сложность $M(n)$ умножения многочленов с действительными коэффициентами можно при $n = 2^k$ оценить как $M(n) = 16n \log n + O(n)$. В общем случае асимптотически оценка остается такой же. Для ее получения выбираем $m = 2^k 3^l$ так, чтобы $n \le m = n(1+\varepsilon_n)$, $\varepsilon_n \to 0, k \to \infty$ (пользуясь равномерной распределенностью по модулю единица последовательности $\{n \log_2 3\}$), потом сводим задачу к умножению многочленов степени $m-1$, а для БПФ порядка m применяем комбинацию алгоритма Гуда–Томаса (см., например,[10]), произвольного БПФ порядка 3^l и указанного выше БПФ порядка 2^k.

Использование дискретного преобразования Хартли (ДПХ) позволяет для

сложности умножения действительных многочленов получить при $n = 2^k$ оценку

$$M(n) = 12n \log_2 n + O(n).$$

(см. [23]).

В указанных алгоритмах умножения операции с действительными числами выполняются в реальности приближенно, так как используемые корни n-й степени из единицы иррациональны. В случае многочленов с целыми коэффициентами произведение можно вычислить точно, но с не очень хорошей оценкой битовой сложности (см. [5].) При этом используется ДПФ над кольцом вычетов по модулю числа Ферма.

Умножение многочленов над простыми конечными полями можно свести к умножению чисел. Для этого достаточно заметить, что, например, в случае $p = 2$ умножение многочленов степени n сводится к умножению $n \log n$ - битовых чисел, если сопоставить произвольному многочлену

$$a_0 + a_1 x + \ldots + a_{n-1} x^{n-1}$$

число

$$a_0 + a_1 2^k + \ldots + a_{n-1} 2^{(n-1)k},$$

где $k = \log n + O(1)$ и заметить, что при умножении таких чисел

$$a_0 + a_1 2^k + \ldots + a_{n-1} 2^{(n-1)k}$$

и

$$b_0 + b_1 2^k + \ldots + b_{n-1} 2^{(n-1)k},$$

получается число

$$c_0 + c_1 2^k + \ldots + c_{2n-2} 2^{(2n-2)k},$$

где $c_j = \sum a_i b_{j-i} \leq n < 2^k$, и поэтому двоичные записи всех коэффициентов как обычного, так и по модулю два, произведения соответствующих многочленов будут естественным образом считываться с двоичной записи указанного числа, а умножение чисел можно провести методом Шенхаге-Штрассена [74], использующим ДПФ над кольцом вычетов по модулю числа Ферма и отрицательно обернутую циклическую свертку.

Используя идею работы [74], Шенхаге показал в [147], что над любым конечным кольцом вычетов сложность умножения многочленов степени n равна $64(n \log_2 n \log_2 \log_2 n) + O(n \log_2 n)$. Его алгоритм использует ДПФ порядка n над кольцом многочленов по модулю $x^n + 1$, при n, равных как степеням двойки, так и степеням тройки (в случае полей характеристики два используется только ДПФ порядка 3^k). Благодаря этому он избежал трудностей, связанных с использованием расширений поля коэффициентов многочлена. Работа [147] остается малоизвестной, и ее подробное изложение появилось в учебной литературе только в [101]. Недавно появилось ее изложение и на русском языке в

[17] (почему-то со ссылкой на работу [159], выполненную позднее Шенхаге). Алгоритм Шенхаге довольно изощренный и поэтому мы его излагать не будем. Кроме того, из-за большой мультипликативной константы при при n порядка нескольких тысяч он все еще хуже стандартных алгоритмов.

Некоторое улучшение алгоритма Шенхаге было получено в работе [91] за счет использования одной идеи из [140]. Далее мы даем ее краткое изложение, опираясь на работу D.Bernstein Multidigit multiplications for mathematicians. Построенный в ней алгоритм перемножает многочлены степени $n \geq 7$ над любым кольцом R, используя не более $(3 + 36/\log_2 3)n \log_2 n$ умножений и не более $(12 + 54/\log_2 3)n \log_2 n \log_2(16 \log_2 n)$ сложений в R. Этот алгоритм использует два вспомогательных алгоритма, основанных на использовании ДПФ порядков 2^k и 3^k. В первом из них два многочлена степени меньшей $2^m, 2^{e-1} < m \leq 2^e$, перемножаются по модулю $x^{2^m} + 1$ с появлением дополнительного множителя 2^{m+e-1} (вычисляется так называемая отрицательно обернутая свертка) с 2^{m+e-1} умножений и не более чем $2^m(2^e(3e+8)-7)$ сложений в R. Во втором алгоритме два многочлена степени меньшей $2 \cdot 3^m$, перемножаются по модулю $x^{2 \cdot 3^m} + x^{3^m} + 1$ с появлением дополнительного множителя 3^{m+e-1} с $3^{m+1}2^{e+2}$ умножений и менее чем $3^m(2^e(18e + 39) - 26)$ сложений в R.

В работе [90] предложен асимптотически более медленный, но, возможно, более практичный алгоритм умножения многочленов над конечными полями. Вместо ДПФ в нем используется интерполяция в точках, лежащих в аддитивных подгруппах подходящего расширения данного конечного поля (ДПФ с этой целью использует мультипликативные подгруппы — а именно, группы корней n-й степени из единицы.) При этом для умножения многочленов степени меньшей n над полем $GF(p)$ используется $O(mp^m)$ умножений и $O(m^2p^m)$ скалярных операций (сложений или умножений на элементы из $GF(p)$) в поле $GF(p^m)$, где $4n \leq mp^m$. Окончательные оценки сложности, полученные в [90], имеют вид $O(n \log_2 n$ для числа умножений в поле $GF(p)$ и $O\left(n(\log_2 n)^{1+\log_p((p+1)/2)}\right)$ для числа скалярных операций (сложений и умножений на фиксированные константы).

Тем же методом более аккуратные оценки получены в [102] для умножения многочленов над полем $GF(2^m)$ степени $n \leq 2^m$. Число используемых умножений в этом поле оценивается как

$$\frac{3}{2}n \log_2^2 n + \frac{15}{2}n \log_2 n + 8n,$$

а число скалярных операций — как

$$\frac{3}{2}n \log_2^2 n + \frac{29}{2}n \log_2 n + 4n + 9.$$

Однако этот метод довольно изощренный и оценка его сложности достаточно велика.

В некоторых случаях для умножения многочленов над конечными полями можно эффективно применять ДПФ. Но, в отличие от полей действительных и

комплексных чисел, для ДПФ порядка n над конечным полем порядка $q = p^m$ не удается стандартным образом получить оценку $F(n) = O(n \log n)$. Затруднения связаны с тем, что указанное преобразование существует лишь, когда $q - 1$ кратно n, и оценку его сложности нельзя так просто свести к случаю, когда n является степенью двойки.

Поэтому приходится поступать следующим образом. Разлагая n на простые множители, $n = p_1^{\beta_1} \dots p_r^{\beta_r}$, и применяя алгоритмы Гуда-Томаса и Кули-Тьюки, получаем оценки

$$F(n) = n \left(\frac{F(p_1^{\beta_1})}{p_1^{\beta_1}} + \ldots + \frac{F(p_r^{\beta_r})}{p_r^{\beta_r}} \right) \leq$$

$$\leq n \left(\beta_1 \left(\frac{F(p_1)}{p_1} + 1 \right) + \ldots + \beta_r \left(\frac{F(p_r)}{p_r} + 1 \right) \right).$$

В случае <гладкого> числа n (то есть когда все p_i малы) получается оценка, близкая к $O(n \log n)$.

Если же какое-нибудь число p_i велико, то для оценки сложности $F(p_i)$ можно применить основанный на применении китайской теоремы об остатках алгоритм Винограда (см. [10]) вычисления циклической свертки, представив многочлен $x^{p_i} - 1$ в виде произведения круговых многочленов

$$\prod_{d|p_i} Q_d,$$

с коэффициентами из поля $GF(p)$, среди которых сравнительно мало ненулевых, и поэтому на эти многочлены проще делить обычным школьным алгоритмом, а потом круговые многочлены разложить над полем $GF(q)$ и делить на них.

Можно также методом Райдера свести вычисление ДПФ $F(p_i)$ к вычислению циклической свертки порядка $p_i - 1$, которая, в свою очередь, сводится к трехкратному применению ДПФ порядка $p_i - 1$, которое придется вычислять над полем $GF(p^{m_i})$, где m_i/m — порядок числа p^m по модулю $p_i - 1$ (см. [10]).

В некоторых случаях указанных выше трудностей можно избежать. Так, далее с помощью алгоритма БПФ с расщепленным основанием будет показано, что мультипликативная сложность умножения многочленов степени меньше $n \leq q$ над полем $GF(q)$, где $q = 2^p - 1$ — простое число Мерсенна, оценивается как

$$2\frac{1}{4} n \log_2 n + O(n),$$

а аддитивная сложность — как

$$9\frac{3}{8} n \log_2 n + O(n).$$

Приведем еще примеры. Пусть $n = 3^k$ и $q + 1$ кратно n. Например, это возможно, когда $q = 2^{n/3}$ или когда q – простое вида $a \cdot 3^k - 1$ (таких чисел

бесконечно много согласно теореме Дирихле, встречаются среди них, например, и числа вида $2 \cdot 3^k - 1$). Тогда для умножения многочленов над полем $GF(q)$ степени меньшей $n = (3^k+1)/2$ мультипликативная сложность равна $16n \log_3 n + O(n)$, а аддитивная сложность равна $50\frac{2}{3} n \log_3 n + O(n)$. Если же q четно, то оценка аддитивной сложности улучшается до $45\frac{1}{3} n \log_3 n + O(n)$.

Если $q \equiv 3 \pmod 4$, то мультипликативная сложность умножения многочленов степени меньшей $n = 3^k$ оценивается как

$$8n \log_3 n + O(n),$$

а аддитивная сложность оценивается как

$$16n \log_3 n + O(n).$$

При использовании в качестве элементарных операций сложения и умножения в поле $GF(q^2)$ мультипликативная сложность умножения многочленов степени меньшей $n = 3^k$ над полем $GF(q)$ оценивается как

$$5\frac{1}{3} n \log_3 n + O(n),$$

и аддитивная сложность оценивается как

$$8n \log_3 n + O(n).$$

При четном q оценка мультипликативной сложности умножения многочленов степени меньшей $n = (3^k+1)/2$ над полем $GF(q)$ имеет вид $10\frac{2}{3} n \log_3 n$ и оценка аддитивной сложности имеет вид $11\frac{1}{3} n \log_3 n$.

3.7.3 Об умножении многочленов, ДПФ и ДПХ в конечных полях

О преобразовании Хартли см., например, [15]. Будем обозначать ДПФ порядка n через F_n. Тогда соответствующее преобразование Хартли (ДПХ) по определению равно $H_n = aF_n + \overline{a}\overline{F_n}$, где $a = (1 - \imath)/2$, а знаком $\overline{a}$ будем обозначать переход к комплексно сопряженному числу (или преобразованию). Из определения следует, что $H_n = \Re F_n + \Im F_n$ — сумме действительной и мнимой части преобразования F_n, так как для любого комплексного числа z справедливо тождество $az + \overline{az} = \Re z + \Im z$, из которого следуют равенства

$$a^2 + \overline{a}^2 = \Re a + \Im a = 0, 2a\overline{a} = \Re\overline{a} + \Im\overline{a} = 1$$

и тождество $a\overline{z} + \overline{a}z = \Re z - \Im z$.

Из указанного тождества также следует, что элементы матрицы ДПХ, имеют вид

$$\Re\omega_n^{kl} + \Im\omega_n^{kl} = \sin(2\pi kl/n) + \cos(2\pi kl/n),\ 0 \le k, l \le n-1,$$

где $\omega_n^{kl} = e^{2\pi\imath kl/n}$, — элементы матрицы ДПФ, хотя это нам далее не понадобится. Заметим все же, что обычно вместо $\sin(2\pi kl/n) + \cos(2\pi kl/n)$ обычно пишут $cas(2\pi kl/n)$, и ДПХ представляют в виде

$$H_n(k) = \sum_{l=0}^{n-1} x_l cas(2\pi kl/n),\ k = 0, \ldots, n-1$$

откуда видно, что оно является действительным линейным преобразованием.

Другим удобным его свойством является то, что оно фактически совпадает со своим обратным преобразованием. Действительно, известно, что $F_n\overline{F_n} = nE_n, F_n^2 = nI_n$, где E_n, I_n — единичные матрицы, в которых единицы стоят соответственно на главной и побочной диагоналях (а в остальных местах — нули). Поэтому

$$H_n^2 = (aF_n + \overline{a}\overline{F_n})^2 = a^2F_n^2 + \overline{a^2F_n^2} + a\overline{a}(F_n\overline{F_n} + \overline{F_n}F_n) =$$

$$(a^2 + \overline{a}^2)nI_n + a\overline{a}2nE_n = a\overline{a}2nE_n = nE_n,$$

откуда

$$H_n^{-1} = \frac{1}{n}H_n.$$

Обозначим A^* преобразование, получающееся из преобразования A обратной перестановкой компонент, а именно $A^*(k) = A(n-k), k = 0, \ldots, n-1$. Очевидно, что $A^{**} = A$. Очевидно также, что для действительного ДПФ (т. е. применяемого к действительным векторам) справедливо равенство $F_n^* = \overline{F_n}$. Отсюда следует, что

$$H_n^* = (aF_n + \overline{a}\overline{F_n})^* = aF_n^* + \overline{a}\overline{F_n^*} = a\overline{F_n} + \overline{a}F_n = \Re F_n - \Im F_n.$$

Обозначим сложность ДПХ n-го порядка через $H(n)$. Тогда с помощью равенств

$$H_n = \Re F_n + \Im F_n,\ H_n^* = \Re F_n - \Im F_n\ \Re F_n = \frac{H_n + H_n^*}{2}, \Im F_n = \frac{H_n - H_n^*}{2}$$

легко получить, что $H(n) \le F(n) + n, F(n) \le H(n) + 3n$.

Комплексное ДПХ удобнее определить равенством $H_n = aF_n + \overline{a}F_n^*$, которое можно использовать и в действительном случае т. к. тогда $F_n^* = \overline{F_n}$. Ясно, что $\Re H_n(z) = H_n(\Re z), \Im H_n(z) = H_n(\Im z)$, чего нельзя сказать о ДПФ. Из указанного равенства следует, что

$$H_n^* = aF_n^* + \overline{a}F_n^{**} = aF_n^* + \overline{a}F_n,\ \overline{a}H_n + aH_n^* =$$

$$= 2a\overline{a}F_n + (a^2 + \overline{a}^2)F_n^* = F_n, \overline{a}H_n^* + aH_n = 2a\overline{a}F_n^* + (a^2 + \overline{a}^2)F_n = F_n^*.$$

Из полученных равенств $H_n = aF_n + \overline{a}F_n^*, F_n = \overline{a}H_n + aH_n^*$ легко получить для комплексных преобразований неравенства $H(n) \le F(n) + 14n, F(n) \le H(n) + 14n$ (в которых можно заменить $14n$ на $6n$, если считать, что действительные

умножения на $1/2$ выполняются «бесплатно»), где $F(n)$ означает сложность преобразования F_n, а $H(n)$ — сложность преобразования H_n. Отметим еще, что сложность комплексного ДПХ очевидно совпадает со сложностью совместного вычисления двух действительных ДПХ для независимо выбранных аргументов. Для ДПФ справедливо несколько более слабое утверждение, а именно

$$F(n) \le 2F^{\mathbf{R}}(n) + 2n,\ comp(F_n^{\mathbf{R}}(x), F_n^{\mathbf{R}}(y)) \le F(n) + 8n,$$

где $F^{\mathbf{R}}(n)$ означает сложность действительного ДПФ, т.е. преобразования, применяемого к действительным векторам, а под $comp(F_n^{\mathbf{R}}(x), F_n^{\mathbf{R}}(y))$ понимается действительная сложность совместного вычисления двух действительных ДПФ. Для доказательства достаточно воспользоваться тождествами

$$F_n(z) = F_n(\Re z) + \imath F_n(\Im z),\ F_n^*(z) = \overline{F_n(\overline{z})} = \overline{F_n(\Re z) - \imath F_n(\Im z)},$$

$$\overline{F_n^*(z)} = F_n(\Re z) - \imath F_n(\Im z), F_n(\Re z) = \frac{F_n(z) + \overline{F_n^*(z)}}{2},$$

$$F_n(\Im z) = \frac{F_n(z) - \overline{F_n^*(z)}}{2\imath}.$$

Из известного неравенства Кули-Тьюки ([10]) вытекает, что $F^{\mathbf{R}}(2n) \le 2F^{\mathbf{R}}(n) + 8n$, где под $2F^{\mathbf{R}}(n)$ можно понимать, в частности, сложность совместного вычисления двух действительных ДПФ порядка n. Поэтому

$$F(n) \le 2F^{\mathbf{R}}(n) + 2n,\ F^{\mathbf{R}}(2n) \le F(n) + 16n,$$

откуда следует, что наилучшие из известных алгоритмов комплексного БПФ, имеющие при $n = 2^k$ оценку $F(n) = 4n\log_2 n + O(n)$ (см. [20]) дают для действительного БПФ оценку $F^{\mathbf{R}}(n) = 2n\log_2 n + O(n)$, а значит и для действительного ДПХ имеем оценку $H(n) = 2n\log_2 n + O(n)$. Отметим, что при использовании этих алгоритмов мультипликативная сложность в четыре раза меньше тотальной сложности.

Известно, что для ДПФ выполняется теорема о циклической свертке $F_n(x \odot y) = F_n(x) \otimes F_n(y)$, где $x \otimes y$ — операция покомпонентного умножения векторов x и y, а $u = x \odot y$ — циклическая свертка векторов x и y, которая определяется равенством $u_k = \sum\limits_{i \oplus j = k} x_i y_j$, где $i \oplus j = i + j \pmod n$ — операция сложения по модулю n. Для ДПХ соответствующее тождество принимает несколько менее красивый вид: $H_n(x \odot y) =$

$$= \frac{1}{2}(H_n(x) \otimes H_n(y) + H_n^*(x) \otimes H_n(y) + H_n(x) \otimes H_n^*(y) - H_n^*(x) \otimes H_n^*(y)),$$

так как

$$H_n(x \odot y) = aF_n(x \odot y) + \overline{a}F_n^*(x \odot y) = aF_n(x) \otimes F_n(y) + \overline{a}(F_n(x) \otimes F_n(y))^* =$$

$$= aF_n(x) \otimes F_n(y) + \overline{a}F_n^*(x) \otimes F_n^*(y) =$$

$$a((\overline{a}H_n(x)+aH_n^*(x))\otimes(\overline{a}H_n(y)+aH_n^*(y)))+$$
$$\overline{a}((\overline{a}H_n^*(x)+aH_n(x))\otimes(\overline{a}H_n^*(y)+aH_n(y)))=$$

$$=(a\overline{a}^2+a^2\overline{a})H_n(x)\otimes H_n(y)+(a^2\overline{a}+a\overline{a}^2)H_n^*(x)\otimes H_n(y)+$$
$$+(a^2\overline{a}+a\overline{a}^2)H_n(x)\otimes H_n^*(y)+(a^3+\overline{a}^3)H_n^*(x)\otimes H_n^*(y)=$$
$$=\frac{1}{2}(H_n(x)\otimes H_n(y)+H_n^*(x)\otimes H_n(y)+H_n(x)\otimes H_n^*(y)-H_n^*(x)\otimes H_n^*(y)),$$

благодаря равенствам $a^2\overline{a}+a\overline{a}^2=(a+\overline{a})a\overline{a}=(a+\overline{a})/2=1/2$,
$a^3+\overline{a}^3=(a+\overline{a})((a+\overline{a})^2-3a\overline{a})=1-3a\overline{a}=-1/2$. Отметим, что и ДПФ и ДПХ-варианты теоремы о свертке справедливы, очевидно, как и в действительном, так и в комплексном случаях.

Если обозначить $M(n)$ сложность умножения многочленов степени $n-1$, а $Z(n)$ — сложность вычисления циклической свертки, то, как вытекает из сказанного выше, $M(n)\le Z(2n)$ и в случае действительной свертки

$$Z(n)\le 2F^{\mathbf{R}}(n)+F(n)+8n+1\le 4F^{\mathbf{R}}(n)+10n+1, \tag{3.24}$$
$$Z(n)\le 3H(n)+8n+1. \tag{3.25}$$

Действительно, для вычисления $H_n(x\odot y)$ согласно указанному тождеству нужно выполнить 2 преобразования H_n и $8n$ арифметических операций, а для вычисления циклической свертки $x\odot y$ нужно еще одно такое преобразование и n операций умножения на $1/n$. В силу линейности их можно объединить с операциями умножения на $1/2$. Остается еще учесть одну операцию вычисления $1/(2n)$.

Отметим, что (3.25) сильнее чем (3.24), так как $H(n)\le F(n)+n$, откуда имеем $Z(n)\le 3F(n)+11n+1$. Применяя (3.25), получаем, что сложность умножения действительных многочленов оценивается как

$$M(n)\le 3H(2n)+16n+1\le 3F^{\mathbf{R}}(2n)+22n+1\le$$
$$\le 6F^{\mathbf{R}}(n)+46n+1\le 12n\log_2 n+O(n). \tag{3.26}$$

Неравенство (А.2) верно лишь для n, равных степени двойки. Для произвольных n оно заменяется на асимптотическое неравенство (которое можно получить приемом, указанным выше).

Отметим, что для комплексной сложности справедливы неравенства

$$F(n)\le 2F^{\mathbf{R}}(n)+2n,$$
$$comp(F_n^{\mathbf{R}}(x),F_n^{\mathbf{R}}(y))\le F(n)+6n, \tag{3.27}$$

если к числу арифметических операций добавить операцию сопряжения, а неравенство Кули-Тьюки для комплексной сложности действительного ДПФ принимает вид $F(2n)\le 2F(n)+2n$. Учитывая, что для комплексной сложности

комплексного ДПФ справедлива известная оценка $F(n) = (1 + \frac{1}{2})n \log_2 n$, имеем отсюда, что комплексная сложность действительного ДПФ не превосходит $\frac{3}{4}n \log_2 n + 4n$, а значит, согласно, показанному выше, комплексная сложность действительного ДПХ не превосходит $\frac{3}{4}n \log_2 n + 7n$, Поэтому оценка для комплексной сложности умножения действительных многочленов принимает вид

$$M(n) \leq 3H(2n) + 16n + 1 \leq 4\frac{1}{2}n \log_2 n + 58n + 1. \tag{3.28}$$

Пусть $GF(q^2)$ — расширение конечного поля $GF(q)$ с помощью неприводимого многочлена $x^2 + 1$ (аналог поля комплексных чисел). Известно, что оно существует при $q \bmod 4 = 3$ и только при этом условии. Элементы поля $GF(q^2)$ можно представить в виде $a + \imath b$, где $a, b \in GF(q)$,$\imath \in GF(q^2)$ — корень уравнения $x^2 + 1 = 0$ (в качестве $\imath$ можно взять $\alpha^{(q^2-1)/4}$, где α — любой примитивный элемент поля $GF(q^2)$). Для существования первообразного корня n-й степени из единицы в поле $GF(q^2)$ необходимо и достаточно, чтобы $q^2 - 1$ было кратно n (тогда в качестве этого корня можно взять $\alpha^{(q^2-1)/n}$, где α —упоминавшийся примитивный элемент). Допустим, что $q - 1$ не кратно n, т.е. в поле $GF(q)$ первообразного корня n-й степени из единицы нет. Тогда для выполнения ДПФ порядка n придется вместо поля $GF(q)$ использовать его квадратичное расширение $GF(q^2)$ аналогично тому, что в случае вычисления обычного ДПФ приходится использовать комплексные числа. В указанном расширении можно по аналогии с комплексными числами ввести операцию сопряжения, и также как в случае комплексных чисел сопряженное к произведению двух чисел число будет равно произведению чисел, сопряженных к исходным (мультипликативность операции сопряжения).

Предположим дополнительно, что в поле $GF(q^2)$ существует такой первообразного корень n-й степени из единицы α, что $\alpha\overline{\alpha} = 1$ (здесь $\overline{\alpha}$, как и выше обозначает сопряженное к α число), тогда $\alpha^{n-1} = \overline{\alpha}$, и все приведенные выше результаты могут быть без существенных изменений перенесены в рассматриваемую ситуацию. Операции в поле $GF(q^2)$ также могут быть сведены к операциям в поле $GF(q)$, как и операции над комплексными числами сводятся к операциям над действительными числами. Количество операций в поле $GF(q^2)$, необходимых для вычисления рассматриваемого ДПФ, будет играть роль комплексной сложности, а количество операций в поле $GF(q)$ — роль действительной сложности. Может быть определено для случая конечных полей и ДПХ, и введены аналоги его комплексной и действительной сложности.

Для выполнения умножения многочленов степени не выше $(n-1)/2$ над полем $GF(q)$ аналогично действительному случаю могут быть применены ДПФ и ДПХ порядка n с теми же оценками сложности, что и действительном (комплексном) случае. Отличие от последнего случая только в том, что в поле $GF(q^2)$ нельзя выполнять ДПФ порядка большего n, а в поле комплексных чисел можно выполнять ДПФ любого порядка.

Для получения оценок сложности умножения «действительных» многочленов выгодно использовать ДПФ порядка, равного степени двойки, так как для

этого случая известны алгоритмы с наименьшими мультипликативными константами в оценках сложности. В случае поля $GF(q)$ соответствующий ему порядок n может не иметь такого вида. Однако в одном важном (и наиболее популярном) случае n может иметь такой вид, а именно, когда $q = 2^p - 1$ — простое число Мерсенна, тогда в поле $GF(q^2)$ возможно выполнение ДПФ порядка $n = 2^{p+1}$ ([10],[48]). Как известно ([20], [48]), в случае $n = 2^{p+1}$ любой первообразный корень n-го порядка удовлетворяет условию $\alpha\overline{\alpha} = -1$.

Упражнение 3.7.13 Примером такого корня является $2^{2^{p-2}} + i3^{2^{p-2}} \bmod q$. Квадрат этого элемента является при $n = 2^p$ примером первообразного корня n-го порядка, удовлетворяющего условию $\alpha\overline{\alpha} = 1$ ([48]).

Поэтому из (А.3) следует, что сложность умножения многочленов степени не больше $n = q$ над полем $GF(q)$ не превосходит $12n\log_2 n + O(n)$, где под элементарными операциями понимаются арифметические операции по модулю q, а мультипликативная сложность равна $3n\log_2 n + O(n)$. В отличие от действительного случая используемое здесь ДПХ выполняется абсолютно точно, и умножение многочленов тоже выполняется точно.

Если же в качестве элементарных операций использовать операции в поле $GF(q^2)$, то сложность умножения многочленов в этой ситуации можно, используя А.3, оценить как $4\frac{1}{2}n\log_2 n + 58n + 1$ (аналог оценки комплексной сложности умножения действительных многочленов). Мультипликативная сложность опять будет в асимптотически в четыре раза меньше тотальной сложности.

Если умножение в поле $GF(q^2)$ выполнять сведением к операциям в поле $GF(q)$, то получится оценка несколько хуже, чем полученная без использования операций в поле $GF(q^2)$. Однако возможны ситуации, когда второй метод все же выгоднее.

3.7.4 Применение к умножению многочленов с коэффициентами $0, 1$ и умножению чисел

Если выполнять умножение многочленов с коэффициентами 0 и 1 и степени не выше $q - 2$, над кольцом целых чисел, то его результат можно однозначно восстановить, выполнив умножение тех же многочленов над полем $GF(q)$. По результату умножения этих многочленов над полем $GF(q)$ можно также однозначно восстановить их произведение над полем $GF(2)$. При $q = 2^p - 1$ — простом числе Мерсенна сложность во всех случаях будет оцениваться как $6q\log_2 q + O(q)$, а мультипликативная сложность — как $1.5q\log_2 q + O(q)$. Для оценки битовой сложности нужно приведенную оценку умножить на оценку битовой сложности умножения в поле $GF(q)$, которая очевидно равна $O(M(p))$, где $(M(p))$— сложность умножения p-разрядных двоичных чисел. Согласно результату Шенхаге-Штрассена $M(p) = O(p\log p\log\log p)$. Но даже используя стандартную оценку $M(p) = O(p^2)$, получаем, что сложность умножения многочленов степени n с коэффициентами $0, 1$ при условии, что для некоторого числа Мерсенна $n < 2^p - 1, p = O(\log n)$, оценивается как $O(n\log^3 n)$.

Упражнение 3.7.14 Докажите, что сложение в поле $GF(q), q = 2^p - 1$ выполнятся с битовой сложностью $11p$.

Указание. Обычное сложение p-разрядных двоичных чисел требует $5p$ двоичных операций. Прибавление единицы требует $2p$ операций. Сравнение с числом 2^p-1 требует p операций. Выбор из двух вариантов требует $3p$ операций.

Упражнение 3.7.15 Докажите, что умножение в поле $GF(q), q=2^p-1$ выполнятся с битовой сложностью $M(p)+11p$.

Упражнение 3.7.16 Докажите, что $M(p)<6p^2$.

Упражнение 3.7.17 Докажите, что умножение многочленов с коэффициентами 0 и 1 и степени не выше $n=q-2=2^p-3$, где q— число Мерсенна, имеет битовую сложность

$$3n\log_2 nM(p)+132n\log_2^2 n+O(nM(p))=$$

$$18n\log_2^3 n+132n\log_2^2 n+O(n\log^{\log_2 3} n).$$

Упражнение 3.7.18 Докажите, что умножение n-разрядных двоичных чисел при $n=q-2$, где $q=2^p-1$ — число Мерсенна, имеет битовую сложность

$$3n\log_2 nM(p)+132n\log_2^2 n+O(nM(p))=$$

$$18n\log_2^3 n+132n\log_2^2 n+O(n\log^{\log_2 3} n).$$

Указание. Примените предыдущую задачу и перемножьте многочлены, соответствующие данным числам. Покажите, что из полученного в результате этого умножения многочлена с коэффициентами от 0 до $q-1$ можно получить число, являющееся произведением данных чисел путем сложения n p разрядных чисел со сложностью $5p(n-1)$.

Упражнение 3.7.19 Докажите, что сложность умножения n-разрядных двоичных чисел при условии, что для некоторого числа Мерсенна $n<2^p-1, p=O(\log n)$, оценивается как $O(n\log^{\log_2 6} n)$.

При оценке времени вычисления на компьютере битовая сложность не нужна, вместо нее естественно использовать оценку числа операций в поле $GF(q)$, или в поле $GF(q^2)$, если объем оперативной памяти машины позволяет хранить таблицы умножения и сложения в этих полях. Это можно сделать, например, при простом $q=2^7-1$ для поля $GF(q^2)$, поэтому сложность умножения многочленов степени $n\le 127$ по модулю 127 в этой ситуации можно, используя А.3, оценить как $4\frac{1}{2}n\log_2 n+58n+1$, где элементарными операциями являются операции в поле $GF(q^2)$.

Эта оценка меньше стандартной оценки сложности умножения многочленов при $n=127$, и меньше оценки Карацубы, так как в этих алгоритмах в качестве элементарных операций естественно используются только операции в поле $GF(q)$.

Заметим, что объем каждой упомянутых таблиц равен 2^{18} килобайт (размер таблицы равен 2^{27} и каждый элемент поля кодируется двумя байтами) и может быть слишком велик для малых машин. Тогда, чуть уменьшив скорость работы, можно обойтись без таблицы сложения, заменяя каждое сложение в поле $GF(q^2)$ на два сложения по модулю $q=2^7-1$, а каждое умножение заменяя на

сложение по модулю q^2-1 и трехкратное использование таблицы дискретных логарифмов, в которой в качестве основания можно взять любой примитивный элемент этого поля. Таблицы логарифмов и антилогарифмов (экспонент) вычисляются заранее. Их размер $2(q^2-1) < 2^{15}$ байт. Вместо сложений по модулю $q = 2^7 - 1$ естественно использовать сложение 16-битных чисел и вычитание подходящего 16-битного модуля. Сложение по модулю q^2-1 тоже реализуется как обычное сложение 16-битных чисел и вычитание модуля.

3.7.5 Использование логарифмов Гаусса

Дополнительного ускорения можно достичь, если использовать логарифмы не только для умножения, но и для сложения — это так называемые логарифмы Гаусса. Пусть α — примитивный элемент поля $GF(q^2)$, используемый как основание логарифмов, и $a = \log_\alpha x, b = \log_\alpha y$, тогда вычисление логарифма суммы по формуле

$$\log_\alpha x + y = \log_\alpha x + log_\alpha 1 + y/x = \log_\alpha x + log_\alpha(1 + \alpha^{log_\alpha y - log_\alpha x \bmod q^2-1})$$

требует одной операции вычитания по модулю q^2-1 и однократного обращения к таблице функции $log_\alpha(1+\alpha^x)$. Размер этой таблицы такой же, как и таблицы логарифмов.

Если в предыдущем алгоритме использовались $4\frac{1}{2}n\log_2 n + O(n)$ обращения к таблицам для выполнения умножений, и $5\frac{1}{2}n\log_2 n + O(n)$ операций сложения по по модулям $q-1$ или q^2-1 , то в новом алгоритме мы все время работаем не с элементами поля, а с их логарифмами, и используем только $3n\log_2 n + O(n)$ обращений к таблицам логарифмов Гаусса для выполнения сложений, и $4\frac{1}{2}n\log_2 n + O(n)$ операций сложения по модулю q^2-1 и еще $4n-1$ обращений к обычным таблицам логарифмов в начале и конце работы алгоритма.

3.7.6 Алгоритм БПФ с расщепленным основанием в конечных полях

Приведенные выше оценки сложности умножения многочленов над полями $GF(q^2)$, где $q = 2^p - 1$ — простое число Мерсенна, можно улучшить, если применить алгоритм с расщепленным основанием [20]. Мы приведем полиномиальную версию этого алгоритма, на наш взгляд более легкую для восприятия, чем изложение в [20].

Как известно, для вычисления ДПФ n-мерного вектора a достаточно взять многочлен $n-1$-й степени $a(x)$ с указанным вектором коэффициентов и найти все его остатки $f_k = a(x) \pmod{x - \omega_n^k}, k = 0, \ldots n-1$ по модулям линейных двучленов, коэффициенты которых ω_n^k пробегают множество всех корней из единицы в рассматриваемом поле. Для этого при n кратном 8 воспользуемся разложением

$$x^n - 1 = (x^{n/2} - 1)(x^{n/2} + 1) = (x^{n/2} - 1)(x^{n/4} - \imath)(x^{n/4} + \imath) =$$

$$= (x^{n/2}-1)(x^{n/8}-\varepsilon)(x^{n/8}-\imath\varepsilon)(x^{n/8}+\varepsilon)(x^{n/8}+\imath\varepsilon),$$

где

$$\varepsilon = 2^{(p-1)/2}+\imath 2^{(p-1)/2}, \varepsilon^2 = 2^{p-1}\cdot 2\imath = \imath, \varepsilon^4 = -1,$$
$$\varepsilon^6 = -\imath, \varepsilon^8 = 1, \varepsilon^3 = \imath\varepsilon, \varepsilon^5 = -\varepsilon, \varepsilon^7 = -\imath\varepsilon$$

(идея использовать корни восьмой степени из единицы в поле $GF(q^2)$ при выполнении ДПФ принадлежит Нуссбаумеру, см. [48]). Тогда в силу разложений

$$x^{n/2}-1 = \prod_{k=0}^{n/2-1}\left(x-\omega_n^{2k}\right),$$

$$x^{n/8}-\varepsilon = \prod_{k=0}^{n/8-1}\left(x-\omega_n^{8k+1}\right),\ x^{n/8}+\varepsilon = \prod_{k=0}^{n/8-1}\left(x-\omega_n^{8k+5}\right),$$
$$x^{n/8}-\imath\varepsilon = \prod_{k=0}^{n/8-1}\left(x-\omega_n^{8k+3}\right),\ x^{n/8}+\imath\varepsilon = \prod_{k=0}^{n/8-1}\left(x-\omega_n^{8k+7}\right)$$

справедливы следующие равенства:

$$f_{2k} = (a(x) \pmod{x^{n/2}-1}) \pmod{x-\omega_{n/2}^k},\ \omega_{n/2} = \omega_n^2, \omega_{n/2}^{n/4} = -1,$$

$$f_{8k+1} = (a(x) \pmod{x^{n/8}-\varepsilon}) \pmod{x-\omega_n\omega_{n/8}^k}, \omega_{n/8} = \omega_n^8, \omega_{n/8}^{n/16} = -1,$$
$$f_{8k+3} = (a(x) \pmod{x^{n/8}-\imath\varepsilon}) \pmod{x-\omega_n^3\omega_{n/8}^k},$$
$$f_{8k+5} = (a(x) \pmod{x^{n/8}+\varepsilon}) \pmod{x-\omega_n^5\omega_{n/8}^k},$$
$$f_{8k+7} = (a(x) \pmod{x^{n/8}+\imath\varepsilon}) \pmod{x-\omega_n^7\omega_{n/8}^k},.$$

Очевидно, вычисление остатков

$$a(x) \pmod{x^{n/2}-1},\ a(x) \pmod{x^{n/2}+1}$$

требует n комплексных сложений (вычитаний), а вычисление $f_{2k}, k = 0,\ldots,n/2-1$ выполняется с помощью ДПФ порядка $n/2$.

Далее, вычисление остатков $a(x) \pmod{x^{n/4}-\imath}, a(x) \pmod{x^{n/4}+\imath}$ по формулам

$$a(x) \pmod{x^{n/4}-\imath} = (a(x) \pmod{x^{n/2}+1}) \pmod{x^{n/4}-\imath},$$

$$a(x) \pmod{x^{n/4}+\imath} = (a(x) \pmod{x^{n/2}+1}) \pmod{x^{n/4}+\imath}$$

требует $n/2$ комплексных сложений (умножения на $\pm\imath$ «бесплатные»). После этого вычисление остатков $a(x) \pmod{x^{n/8}\pm\varepsilon}, a(x) \pmod{{}^{n/8}\pm\imath\varepsilon}$ по формулам

$$a(x) \pmod{x^{n/8}-\varepsilon} = (a(x) \pmod{x^{n/4}-\imath}) \pmod{x^{n/8}-\varepsilon},$$
$$a(x) \pmod{x^{n/8}-\imath\varepsilon} = (a(x) \pmod{x^{n/4}+\imath}) \pmod{x^{n/8}-\imath\varepsilon},$$

$$a(x) \pmod{x^{n/8}+\imath\varepsilon} = (a(x) \pmod{x^{n/4}+\imath}) \pmod{x^{n/8}+\imath\varepsilon}$$
$$a(x) \pmod{x^{n/8}+\varepsilon} = (a(x) \pmod{x^{n/4}-\imath}) \pmod{x^{n/8}+\varepsilon}$$

требует еще $n/2$ комплексных сложений и $n/8$ умножений на ε и $\imath\varepsilon$, каждая пара которых ввиду формул $\varepsilon = 2^{(p-1)/2}(1+\imath)$, $\imath\varepsilon = 2^{(p-1)/2}(-1+\imath)$ требует 2 действительных сложений-вычитаний и 4 умножений на степени двойки, т.е. сдвигов массива двоичных цифр, которые выполняются существенно быстрее, чем «общее» умножение.

И, наконец, вычисление

$$f_{8k+1} = (a(x) \pmod{x^{n/8}-\varepsilon}) \pmod{x-\omega_n\omega_{n/8}^k}, \omega_{n/8} = \omega_n^8, \omega_{n/8}^{n/16} = -1$$

по формуле $f_{8k+1} = g(x) \pmod{x-\omega_n\omega_{n/8}^k} = g(\omega_n x) \pmod{x-\omega_{n/8}^k}$ требует вначале $n/8-1$ комплексных умножений коэффициентов многочлена $g(x) = f(x) \pmod{x^{n/8}-\varepsilon}$ на заранее известные скаляры $\omega_n^i, i = 1, \ldots n/8-1$, каждое из которых выполняется, как известно [10], за три действительных умножения и три действительных сложения, а потом ДПФ порядка $n/8$. Аналогично вычисляются и компоненты $f_{8k+3}, f_{8k+5}, f_{8k+7}, k = 0, \ldots n/8-1$.

В результате имеем следующие рекуррентные оценки числа «действительных» сложений, умножений и «сдвигов»:

$$MF(n) \le MF(n/2) + 4MF(n/8) + 3n/2 - 12,$$
$$AF(n) \le AF(n/2) + AMF(n/8) + 23n/4,$$
$$SF(n) \le SF(n/2) + 4SF(n/8) + n/2.$$

Рекуррентное соотношение вида $K(n) = K(n/2) + 4K(n/8) + a_1 n + a_2$ имеет, как можно проверить, частное решение $K(n) = \frac{a_1}{2} n \log_2 n - \frac{a_2}{4}$. Для нахождения общего решения нужно прибавить к указанному частному решению общее решение однородного рекуррентного соотношения $K(n) - K(n/2) - 4K(n/8) = 0$, которое после замены $k(n) = K(2^n)$ принимает вид $k(n) = k(n-1) + 4k(n-3)$, и имеет общее решение

$$k(n) = (a+b\imath)(-1/2+\imath\sqrt{7}/2)^n + (a-b\imath)(-1/2-\imath\sqrt{7}/2)^n + c\cdot 2^n.$$

Коэффициенты подбираются так, чтобы значения функций $MF(n)$, $AF(n)$, $SF(n)$ при $n = 2, 4, 8$, вычисленные по указанным формулам, совпадали со следующими легко проверяемыми значениями: $MF(n) = SF(n) = 0, n = 2, 4, 8$, $AF(2) = 4, AF(4) = 16, AF(8) = 34$. Например, в случае $MF(n)$ эти коэффициенты будут равны $c = -39/16, a = -9/32, b = -3/(32\sqrt{7})$. Окончательная оценка в этом случае имеет вид

$$MF(n) \le \frac{3}{4}\left(n\log_2 n - \frac{13}{4}\right) + \frac{3}{16}\sqrt{n}\left(\frac{1}{\sqrt{7}}\sin(\varphi\log_2 n) - 3\cos(\varphi\log_2 n)\right) + 3 \tag{3.29}$$

где $\varphi = \arccos\left(-\frac{1}{2\sqrt{2}}\right)$. Остальные оценки приведем в упрощенном виде:

$$SF(n) \le \frac{1}{4} n \log_2 n + O(n), AF(n) \le 2\frac{7}{8} n \log_2 n + O(n),$$

и окончательно

$$F(n) \le 3\frac{7}{8} n \log_2 n + O(n), F^{\mathbf{R}}(n) \le 1\frac{15}{16} n \log_2 n + O(n), \tag{3.30}$$

Отсюда, используя А.3, имеем оценку сложности умножения многочленов степени меньшей n полем $GF(q^2)$

$$M(n) = 6F^{\mathbf{R}}(n) + O(n) = 11\frac{5}{8} n \log_2 n + O(n)$$

и оценку мультипликативной сложности $2\frac{1}{4} n \log_2 n + O(n)$.

Упражнение 3.7.20 Докажите, что умножение многочленов с коэффициентами 0 и 1 и степени не выше $n = 2^p - 1$, где $2^p - 1$— число Мерсенна, имеет битовую сложность

$$2\frac{1}{4} n \log_2 nM(p) + 128n \log_2^2 n + O(nM(p)) =$$

$$13\frac{1}{2} n \log_2^3 n + 128n \log_2^2 n + O(n^{\log_2 6}).$$

Упражнение 3.7.21 Докажите, что умножение n-разрядных двоичных чисел при $n \le q-2$, где $q = 2^p - 1$ — число Мерсенна, имеет битовую сложность

$$2\frac{1}{4} n \log_2 nM(p) + 128np \log_2 n + O(nM(p)) =$$

$$13\frac{1}{2} np^2 \log_2 n + 128np \log_2 n + O(np^{\log_2 3}).$$

При $n = 8000, p = 13$ сложность этого алгоритма почти такая же, как у стандартного.

3.7.7 О ДПФ порядка степени тройки в конечных полях

Выше отмечалось, что при $q \bmod 4 = 3$ квадратичное расширение $GF(q^2)$ поля $GF(q)$ аналогично полю комплексных чисел и для этого поля был приведен алгоритм БПФ порядка степени двойки. В некоторых случаях, когда это условие не выполнено, возможно построение ДПФ порядка степени тройки.

Пусть $n = 3^k$ и $q+1$ кратно n. Например, это возможно, когда $q = 2^{n/3}$ или когда q – простое вида $a \cdot 3^k - 1$.

Упражнение 3.7.22 Докажите, что среди степеней двойки число $q = 2^{n/3}$ наименьшее, такое, что $q+1$ делится на $n = 3^k$.

Выберем $\xi \in GF(q^2)$ так, чтобы $\xi^3 = 1, \xi \neq 1$ (если $\alpha \in GF(q^2)$ первообразный корень, то можно взять $\xi = \alpha^{(q^2-1)/3}$). Тогда элементы поля $GF(q^2)$ можно представлять в виде $a + b\xi$, а операции над ними проводить по формулам

$$(a + b\xi) + (c + d\xi) = a + c + (b + d)\xi,$$

$$(a + b\xi)(c + d\xi) = ac + bd\xi^2 + (bc + ad)\xi = ac - bd + (bc + ad - bd)\xi$$

(ввиду равенства $1 + \xi + \xi^2 = 0$).

Указанный базис в рассматриваемом расширении обычно связывают с именем Эйзенштейна (см. [20]). Умножение в нем, если его выполнять по формуле

$$(a + b\xi)(c + d\xi) = ac - bd + (ac - (a - b)(c - d))\xi,$$

требует 3 умножений и 4 сложений(вычитаний) в поле $GF(q)$. Одно вычитание можно сэкономить в случае, когда умножается произвольный элемент поля $GF(q^2)$ на фиксированный элемент того же поля, так как это вычитание можно выполнить раз и навсегда заранее. Особенно просто выполняется умножение на элементы ξ и $\xi^2 = -1 - \xi$. Для этого требуется лишь одно вычитание в поле $GF(q)$.

Отметим еще, что для умножения одного числа $(a + b\xi)$ одновременно на ξ и ξ^2 требуется лишь одна операция вычитания согласно формулам

$$(a + b\xi)\xi = -b + (a - b)\xi,\ (a + b\xi)\xi^2 = (b - a) - a\xi$$

(сложность операции смены знака считаем нулевой.)

Пользуясь разложением

$$x^n - 1 = (x^{n/3} - 1)(x^{2n/3} + x^{n/3} + 1) = (x^{n/3} - 1)(x^{n/3} - \xi)(x^{n/3} - \xi^2)$$

ДПФ порядка n можно вычислять по формулам:

$$f_{3i} = (a(x) \bmod x^{n/3} - 1) \bmod x - \omega_n^{3i},$$

$$f_{3i+1} = ((a(x) \bmod x^{\frac{2n}{3}} + x^{\frac{n}{3}} + 1) \bmod x^{\frac{n}{3}} - \xi) \bmod x - \omega_n^{3i+1},$$

$$f_{3i+2} = ((a(x) \bmod x^{\frac{2n}{3}} + x^{\frac{n}{3}} + 1) \bmod x^{\frac{n}{3}} - \xi^2) \bmod x - \omega_n^{3i+2},$$

$$i = 0, \ldots n/3 - 1,$$

где ω_n — как обычно, первообразный корень n-й степени из единицы.

Последние две формулы можно переписать в виде

$$f_{3i+1} = h_1(x) \bmod x - \omega_n^{3i}, h_1(x) = g_1(\omega_n x),$$

$$g_1(x) = (a(x) \bmod x^{2n/3} + x^{n/3} + 1) \bmod (x^{n/3} - \xi),$$

$$f_{3i+2} = h_2(x) \bmod x - \omega_n^{3i}, h_2(x) = g_2(\omega_n^2 x),$$

$$g_2(x) = (a(x) \bmod x^{2n/3} + x^{n/3} + 1) \bmod (x^{n/3} - \xi^2), 0 \le i < n/3.$$

Оценим сложность вычисления ДПФ по этой схеме над полем $GF(q)$. Сложность вычисления остатка $a(x) \bmod x^{2n/3} + x^{n/3} + 1$, равна $4n/3$, так как если разбить вектор коэффициентов многочлена $a(x)$ на три вектора a_0, a_1, a_2 длины $n/3$ каждый, так что $a(x)$ будет выражаться через соответствующие многочлены по формуле $a(x) = a_0(x) + x^{n/3}a_1(x) + x^{2n/3}a_2(x)$, то

$$a(x) \bmod x^{2n/3} + x^{n/3} + 1 = a_0(x) - a_2(x) + (a_1(x) - a_2(x))x^{n/3} \tag{3.31}$$

(напомним, что сложность вычитания поле $GF(q^2)$ равна 2).

Сложность вычисления остатка $a(x) \bmod x^{n/3} - 1$ тоже равна $4n/3$, так как

$$a(x) \bmod x^{n/3} - 1 = a_0(x) + a_1(x) + a_2(x).$$

В случае четного q (поля характеристики два) формула (3.31) принимает вид

$$a(x) \bmod x^{2n/3} + x^{n/3} + 1 = a_0(x) + a_2(x) + (a_1(x) + a_2(x))x^{n/3},$$

и если вычислять $a_0(x) + a_1(x) + a_2(x)$ по формуле $(a_0(x) + a_2(x)) + a_1(x)$, то $2n/3$ сложений можно сэкономить.

Для вычисления многочленов $g_i(x)$ нужно $2n/3$ сложений и $n/3$ одновременных умножений на ξ и ξ^2, т.е. всего $5n/3$ сложений-вычитаний в поле $GF(q)$. Указанные вычисления проводятся по формулам

$$g_1(x) = a_0(x) - a_2(x) + (a_1(x) - a_2(x))\xi,$$

$$g_2(x) = a_0(x) - a_2(x) + (a_1(x) - a_2(x))\xi^2,$$

которые равносильны стандартным формулам ДПФ третьего порядка

$$g_1(x) = a_0(x) + a_1(x)\xi + a_2(x)\xi^2,\ g_2(x) = a_0(x) + a_1(x)\xi^2 + a_2(x)\xi.$$

Для вычисления каждого из многочленов $h_i(x)$ нужно $n/3 - 1$ умножений коэффициентов многочлена $g_i(x)$ на корни из единицы $\omega_n^i, i = 1, \ldots n/3 - 1$, что требует $n - 3$ умножений и $n - 3$ сложений в поле $GF(q)$. После этого остается 3 раза применить ДПФ порядка $n/3$.

В результате для мультипликативной сложности ДПФ получаем рекуррентную оценку $MF(n) \le 3MF(n/3)+2n-6$, а для аддитивной сложности — оценку $AF(n) \le 3AF(n/3) + 6\frac{1}{3}n - 6$. В случае поля характеристики два последнее соотношение имеет вид $AF(n) \le 3AF(n/3) + 5\frac{2}{3}n - 6$.

Решая рекуррентное соотношение $k(n) = 3k(n-1) + a \cdot 3^n + b$, находим, что $k(n) = a \cdot 3^n n + c \cdot 3^n - b/2$. Отсюда выводим оценки мультипликативной, аддитивной и тотальной сложности

$$MF(n) = 2n \log_3 n + O(n),\ AF(n) = 6\frac{1}{3}n \log_3 n + O(n), F(n) = 8\frac{1}{3}n \log_3 n + O(n).$$

В случае поля характеристики два последние два равенства имеют вид

$$AF(n) = 5\frac{2}{3}n\log_3 n + O(n),\ F(n) = 7\frac{2}{3}n\log_3 n + O(n).$$

Применяя А.4, отсюда выводим, что для умножения многочленов над полем $GF(q)$ степени меньшей $n = (3^k + 1)/2$ мультипликативная сложность равна $16n\log_3 n + O(n)$, а аддитивная сложность равна $50\frac{2}{3}n\log_3 n + O(n)$. Если же q четно, то оценка аддитивной сложности улучшается до $45\frac{1}{3}n\log_3 n + O(n)$. Для $n = 3^{k-1}$ оценка мультипликативной сложности имеет вид $24n\log_3 n + O(n)$, а аддитивная сложность равна $68n\log_3 n + O(n)$.

3.7.8 Использование операций из поля $GF(q^2)$

Перенесем на этот случай результаты предыдущих разделов о ДПФ и ДПХ. Для этого достаточно заметить, что в поле $GF(q^2)$ можно определить аналог операции комплексного сопряжения равенством

$$\overline{a + b\xi} = a + b\xi^2 = (a - b) - b\xi.$$

В следующих далее оценках «комплексной» сложности под элементарными операциями понимаем операции в поле $GF(q^2)$.

Очевидно, что квадрат операции сопряжения, является тождественным преобразованием, и сопряжение сохраняет операции сложения-вычитания и умножения. Например, последнее следует из цепочки сравнений

$$(a + b\xi)(c + d\xi) \equiv e + f\xi \bmod 1 + \xi + \xi^2 \Rightarrow$$

$$(a + b\xi^2)(c + d\xi^2) \equiv e + f\xi^2 \bmod 1 + \xi^2 + \xi^4 \Rightarrow$$

$$(a + b\xi^2)(c + d\xi^2) \equiv e + f\xi^2 \bmod 1 + \xi + \xi^2$$

в силу сравнений

$$\xi^3 \equiv 1 \bmod 1 + \xi + \xi^2,\ 1 + \xi^2 + \xi^4 \equiv 0 \bmod 1 + \xi + \xi^2.$$

Через операцию сопряжения стандартным образом определяется норма

$$\|a + b\xi\| = (a + b\xi)\overline{a + b\xi} = (a + b\xi)(a + b\xi^2) =$$

$$= a^2 + b^2\xi^3 + ba(\xi + \xi^2) = a^2 + b^2 - ba,$$

которая всегда принадлежит полю $GF(q)$ и благодаря мультипликативному свойству операции сопряжения сама обладает аналогичным свойством.

Благодаря этому свойству для любого $\alpha \in GF(q^2)$, такого что $\alpha^n = 1$, его норма $\|\alpha\| \in GF(q)$ и $\|\alpha\|^n = 1$. Ввиду кратности $q + 1$ числу n, числа $q - 1$ и n взаимно просты, и, значит, порядок элемента $\|\alpha\|$ равен 1 (ведь согласно малой теореме Ферма он должен быть делителем $q - 1$), т.е. $\|\alpha\| = 1$ (здесь мы

просто проверили, что в поле $GF(q)$ уравнение $x^n = 1$ имеет только единичное решение). Но у любого элемента с единичной нормой обратный по умножению элемент совпадает с сопряженным, поэтому любой корень n-й степени из единицы в поле $GF(q^2)$ имеет обратный элемент равный сопряженному, благодаря чему для «действительного» и «комплексного» ДПФ порядка n в рассматриваемой ситуации справедливы и все указанные в секции 6 соотношения между их «действительными» и «комплексными» сложностями (иногда при дополнительных ограничениях на q, указываемых далее).

Для обоснования этого утверждения нужно внести лишь небольшие изменения в старые доказательства. Действительно, сохранив старые обозначения (в том числе $\Re z = a, \Im z = b$ для $z = a + \xi b$) имеем, что

$$F(n) \le 2F^{\mathbf{R}}(n) + O(n),\ comp(F_n^{\mathbf{R}}(x), F_n^{\mathbf{R}}(y)) \le F(n) + O(n),$$

(где под $comp(F_n^{\mathbf{R}}(x), F_n^{\mathbf{R}}(y))$ понимается сложность совместного вычисления двух «действительных» ДПФ), в силу тождеств

$$F_n(z) = F_n(\Re z) + \xi F_n(\Im z),\ F_n^*(z) = \overline{F_n(\overline{z})} = \overline{F_n(\Re z) + \xi^2 F_n(\Im z)},$$

$$\overline{F_n^*(z)} = F_n(\Re z) + \xi^2 F_n(\Im z), F_n(\Re z) = \frac{\xi F_n(z) - \overline{F_n^*(z)}}{\xi - 1},$$

$$F_n(\Im z) = \frac{F_n(z) - \overline{F_n^*(z)}}{\xi - \xi^2}.$$

В случае нечетного q над полем $GF(q^2)$ кроме ДПФ порядка n существует и ДПФ порядка $2n$, поэтому из неравенства Кули-Тьюки ([10]) вытекает, что для сложности «действительных» ДПФ $F^{\mathbf{R}}(2n) \le 2F^{\mathbf{R}}(n) + O(n)$, где под $2F^{\mathbf{R}}(n)$ можно понимать, в частности, сложность совместного вычисления двух «действительных» ДПФ порядка n. Отсюда имеем $F^{\mathbf{R}}(2n) \le F(n) + O(n)$. Для «комплексной» сложности ДПФ порядка n, равного степени тройки, справедлива известная оценка

$$F(n) = 3\frac{1}{3} n \log_3 n - 2n/3,$$

вытекающая по индукции из рекуррентного соотношения Кули-Тьюки

$$F(n) = 3F(n/3) + (n/3)F(3) + 2n/3,$$

которое удобно переписать для этого в виде

$$\frac{F(n)}{n} + 2/3 = \frac{F(n/3)}{n/3} + 2/3 + \frac{F(3)}{3} + 2/3,$$

если заметить, что $F(3) = 8$, применив при $n = 1$ рассуждения предыдущего раздела. Так как мультипликативная сложность ДПФ третьего порядка в рассматриваемом случае равна 2, то аналогичным образом получаем, что мультипликативная сложность ДПФ порядка n равна

$$F(n) = 1\frac{1}{3} n \log_3 n - 2n/3.$$

Отсюда имеем для аддитивной сложности действительного ДПФ оценку

$$F^{\mathbf{R}}(2n) \le 2n \log_3 n + O(n),$$

а для мультипликативной сложности оценку

$$MF^{\mathbf{R}}(2n) \le 1\frac{1}{3} n \log_3 n + O(n).$$

Поэтому, при использовании в качестве элементарных операций сложения и умножения в поле $GF(q^2)$, имеем оценку мультипликативной сложности умножения многочленов степени меньшей $n = 3^k$ над полем $GF(q)$ в виде

$$5\frac{1}{3} n \log_3 n + O(n),$$

и оценку аддитивной сложности в виде

$$8n \log_3 n + O(n).$$

В случае четного q предыдущее замечание теряет силу. Но в этом случае, используя при $n = 1$ рассуждение из предыдущего раздела, можно заметить, что $F(3) = 7$, так как аддитивная сложность уменьшается до 5. Поэтому, рассуждая аналогично предыдущему, можно получить чуть лучшую оценку аддитивной сложности $AF(n) = 1\frac{2}{3} n \log_3 n$ и оценку мультипликативной сложности $MF(n) = 1\frac{1}{3} n \log_3 n$.

Отсюда, применяя А.4, имеем оценку мультипликативной сложности умножения многочленов степени меньшей $n = (3^k + 1)/2$ над полем $GF(q)$ в виде $10\frac{2}{3} n \log_3 n$ и оценку аддитивной сложности в виде $11\frac{1}{3} n \log_3 n$.

3.7.9 Использование ДПХ

Если $q \equiv 3 \pmod 4$, то можно определить в рассматриваемой ситуации ДПХ. В качестве элементарных операций Естественно для этого использовать равенство $H_n = aF_n + bF_n^*$ с подходящими $a, b \in GF(q^2)$. Однако для выполнения условия $H_n^2 = cE_n, \in GF(q^2)$ согласно равенству

$$H_n^2 = (aF_n + b\overline{F_n})^2 = a^2F_n^2 + b^2\overline{F_n^2} + ab(F_n\overline{F_n} + \overline{F_n}F_n) =$$

$$= (a^2 + b^2)nI_n + 2abnE_n$$

необходимо выполнение условия $a^2 + b^2 = 0$ которое влечет разрешимость в поле $GF(q^2)$ уравнения $x^2 + 1 = 0$, а это верно при нечетном q. Далее пусть $\imath \in GF(q^2)$ — корень этого уравнения.

Так как матрицы преобразований F_n и F_n^* не пропорциональны, то они линейно независимы, и преобразование $H_n = aF_n + bF_n^*$ будет действительным лишь когда

$$aF_n + b\overline{F_n} = H_n = \overline{H_n} = \overline{a}\overline{F_n} + \overline{b}F_n,$$

т.е. при условии $a = \overline{b}$, но тогда равенство $a^2 + b^2 = 0$, имеет при подходящих $c, d \in GF(q)$ вид

$$0 = (c + d\xi)^2 + (c + d\xi^2)^2 = c^2 - d^2 + (2cd - d^2)\xi + c^2 + d^2\xi + -2cd\xi - 2cd =$$

$$= 3c^2 - (c + d)^2,$$

откуда следует разрешимость в $GF(q)$ уравнения $x^2 = 3$. Но если a — корень этого уравнения, то элемент $(a\imath - 1)/2 \in GF(q^2)$ является кубическим корнем из единицы, т.к.

$$((a\imath - 1)/2)^3 = (-3a\imath - 3a\imath(a\imath - 1) - 1)/8 = (3a^2 - 1)/8 = 1,$$

причем не равным 1 (т.к. $3 \neq -9$ в поле $GF(q)$), значит он равен либо ξ либо ξ^2, а элемент $(-a\imath - 1)/2 = ((a\imath - 1)/2)^2$ тогда равен наоборот либо ξ^2, либо ξ, а так как по нашему предположению в поле $GF(q)$ кубических корней не было, то значит $\imath$ тоже не принадлежит этому полю и его расширение $GF(q^2)$ имеет также базис $\{1, \imath\}$, причем в силу отмеченных равенств операция сопряжения относительно этого базиса совпадает с операцией сопряжения относительно базиса $\{1, \xi\}$. Условие разрешимости в $GF(q)$ уравнения $x^2 = 3$ совпадает с равенством единице символа Лежандра $\left(\frac{3}{q}\right)$, что в силу квадратичного закона взаимности (см., например [53], [45]) возможно лишь когда $\left(\frac{q}{3}\right) = (-1)^{(q-1)/2}$, но $\left(\frac{q}{3}\right) = \left(\frac{2}{3}\right) = -1$, значит искомое условие имеет вид $q \equiv 3 \pmod 4$.

Благодаря выполнению этого условия в рассматриваемую ситуацию можно перенести и определение ДПХ и все утверждения про него, а также доказываемые с его помощью аналогично тому, как это сделано выше. Можно также без каких либо изменений перенести в рассматриваемую ситуацию при $n = 3^k$, и 3^k делящем $q + 1$ оценку из [20]

$$F(n) = 8n \log_3 n + O(n),$$

причем для мультипликативной сложности оценка имеет вид

$$MF(n) = 2\frac{2}{3} n \log_3 n + O(n).$$

Отсюда следует при $n = 2 \cdot 3^k$ оценка для мультипликативной сложности «действительного» ДПФ порядка $2n$

$$1\frac{1}{3} n \log_3 n + O(n),$$

и оценка для аддитивной сложности

$$2\frac{2}{3} n \log_3 n + O(n)$$

Из полученных выше результатов вытекает, что сложность ДПХ порядка $n = 2 \cdot 3^k$ оценивается также, как и сложность «действительного» ДПФ того же порядка, т.е.

$$H(n) = 4n \log_3 n + O(n),$$

причем оценка мультипликативной сложности тоже имеет вид

$$1\frac{1}{3} n \log_3 n + O(n).$$

Поэтому, применяя А.2,получаем, что мультипликативная сложность умножения многочленов степени меньшей $n = 3^k$ оценивается как

$$8n \log_3 n + O(n),$$

а аддитивная сложность оценивается как

$$16n \log_3 n + O(n).$$

3.7.10 Аддитивные цепочки.

Назовем *аддитивной цепочкой* любую начинающуюся с 1 последовательность натуральных чисел $a_0 = 1, a_1, \ldots, a_m$, в которой каждое число является суммой каких-то двух предыдущих чисел (или удвоением какого-то предыдущего числа). Обозначим $l(n)$ наименьшую длину аддитивной цепочки, заканчивающейся числом n. Под длиной цепочки $a_0 = 1, a_1, \ldots, a_m$ понимаем число m.

Пример 3.7.1 # 1,2,3,4,5,6,7,8,9,10,11,12,13,14 — аддитивная цепочка,

1,2,3,5,7,14 —- минимальная цепочка для 14, т.е. $l(14) = 4$.

аддитивные цепочки можно изображать в виде ориентированного графа, в котором в вершину a_i идут ребра от вершин a_j, a_k, если $a_i = a_j + a_k$ (в случае, если такое представление неоднозначно, выбираем любое из них и рисуем только два ребра). Если из какой-то вершины выходит только одно ребро, то можно «склеить» эту вершину с той вершиной, в которую ведет это ребро. Так как удаление вершины с одним выходящим ребром ведет к удалению одного ребра, то выполнив процедуру «склейки», получим граф, однозначно определяющий данную цепочку, в котором из каждой вершины выходит не менее двух ребер, и длина этой цепочки будет на единицу больше разности между числом ребер в графе, и числом вершин в нем, потому что каждое ребро входит в одну вершину, а первоначально в каждую вершину, кроме одной, входили два ребра. Граф для предыдущего примера см. на рис. 3.4

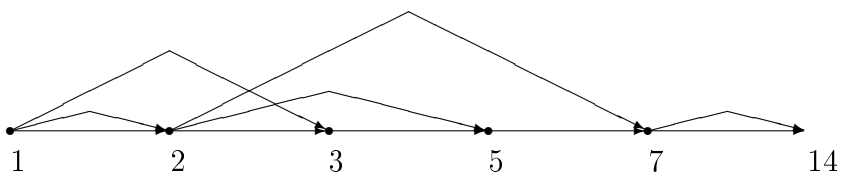

Рис. 3.4:

Можно считать, что все числа в цепочке разные, просто удаляя из нее повторяющиеся числа, и располагать числа в цепочке в порядке возрастания.

Упражнение 3.7.23 Докажите, что наименьшее число операций умножения, требующихся для возведения числа x в степень n, равно $l(n)$.

Таким образом, вычисление x^{14} требует не 13 умножений, а только 5!

Обозначим $\lambda(n) = \lfloor \log_2 n \rfloor$ уменьшенную на единицу длину двоичной записи числа n, а $\nu(n)$ — сумму цифр (другими словами, число единиц) в ней.

Пример 3.7.2 # При $n = 14$ $\lambda(14) = 3$.

Так как $14 = (1110)_2$, то $\nu(14) = 3$.

Записывая число n в двоичной системе и используя схему Горнера, можно написать аддитивную цепочку для числа n длины $\lambda(n) + \nu(n) - 1$ следующим образом

$$n = 2^m b_m + \ldots + b_0 = (\ldots(2b_m + b_{m-1})2 + \ldots + b_1)2 + b_0,$$

$$a_0 = b_m = 1, a_1 = 2a_0 = 2, a_2 = a_1 + b_{m-1}, a_3 = 2a_2, \ldots,$$

$$a_{2m-1} = 2a_{2m-2}, a_{2m} = a_{2m-1} + b_0,$$

где число удвоений равно $m = \lambda(n)$, а число прибавлений единицы равно $\nu(n) - 1$.

Пример 3.7.3 # Так как

$$14 = (1110)_2 = 2^3 + 2^2 + 2 = ((1 \cdot 2 + 1) \cdot 2 + 1) \cdot 2,$$

то получается цепочка

$$1, 2, 3, 6, 7, 14.$$

Этой цепочке соответствует такой алгоритм возведения в степень:

$$x, x^2, x^3 = x^2 \cdot x, x^6 = (x^3)^2, x^7 = x^6 \cdot x, x^{14} = (x^7)^2.$$

Для запоминания последовательности операций в случае ее многократного исполнения, можно заменить в двоичной записи показателя степени $14 = (1110)_2$ каждую единицу, начиная слева (со старших разрядов), кроме самой первой, на слог КУ, а каждый нуль — на букву К, тогда получим слово КУКУК, в котором буквы К означают возведение в квадрат, а буквы У — умножение на основание степени.

Тем самым доказана

Теорема 3.7.2 *Справедливо неравенство*

$$l(n) \leq \lambda(n) + \nu(n) - 1.$$

Интересно, что бинарный метод был фактически известен древним индусам, а задача о нахождении функции $l(n)$ появилась в одном французском журнале в 1894 г. Д.Кнутом доказано, что бинарный метод оптимален при $\nu(n) \leq 4$, за некоторыми исключениями, которые все описаны.

Экспериментально обнаружено, что при $n \leq 1000$ справедлива рекуррентная формула $l(n) = \min\{l(n-1)+1, l_n\} - \delta_n$, где $l_p = \infty$ при простом p и $l_n = l_p + l_{n/p}$, где p — минимальный простой делитель n, а $\delta_n = 0, 1$ причем единицей она бывает редко, только в 105 случаях из 1000.

В доказательстве следующей, многократно переоткрывавшейся, теоремы также используется схема Горнера.

Теорема 3.7.3 *При $k < \log_2 \log_2 n$ справедливо неравенство*

$$l(n) < (1 + 1/k)\,]\log_2 n[+ 2^{k-1} - k + 2.$$

Доказательство. Представим n в двоичной записи :

$$n = \sum_{i=0}^{m} \alpha_i 2^i,$$

где $\alpha_i = 0$ или 1, $m = [\log_2 n]$. Разобьем набор $(\alpha_0, \ldots, \alpha_m)$ не более чем на $]\frac{m+1}{k}[$ блоков $A_0, \ldots, A_s,\ s <]\frac{m+1}{k}[$, каждый из которых, кроме последнего, начинается с 1, состоит из подряд идущих цифр и последняя единица в нем отстоит от первой не более чем на $k-1$ позицию,а последний блок состоит ровно из k цифр (несколько подряд идущих нулей, возможно стоящих в начале, не входят ни в один из рассматриваемых блоков). Числа, двоичными записями которых являются эти блоки, не превосходят $2^k - 1$ и, кроме, возможно, последнего, нечетны. Пусть эти числа суть $a_0, \ldots, a_s$. Тогда n можно представить в виде

$$n = 2^{l_0}\left(2^{l_1}\ldots\left(2^{l_{s-2}}\left(2^{l_{s-1}}\left(2^{l_s}a_s + a_{s-1}\right) + a_{s-2}\right) + \ldots + a_1\right) + a_0\right),$$

где $l_s + l_{s-1} + \ldots + l_0 = m + 1 - k$. Все числа $a_0, \ldots, a_{s-1}$ содержатся в аддитивной цепочке $1, 2, 3, 5, 7, \ldots, 2^k - 1$ длины $2^{k-1} + 1$. Поэтому для вычисления n достаточно добавить к ней последовательность

$$a_s, 2a_s, 4a_s, \ldots, 2^{l_s}a_s, 2^{l_s}a_s + a_{s-1},$$

$$2\left(2^{l_s}a_s + a_{s-1}\right), 4\left(2^{l_s}a_s + a_{s-1}\right), \ldots,$$

$$2^{l_{s-1}}\left(2^{l_s}a_s + a_{s-1}\right), 2^{l_{s-1}}\left(2^{l_s}a_s + a_{s-1}\right) + a_{s-2}, \ldots,$$

$$2^{l_{s-2}}\left(2^{l_{s-1}}\left(2^{l_s}a_s + a_{s-1}\right) + a_{s-2}\right),$$

.....

$$2^{l_1}\left(\ldots 2^{l_{s-2}}\left(2^{l_{s-1}}\left(2^{l_s}a_s + a_{s-1}\right) + a_{s-2}\right) + \ldots + a_1\right) + a_0, \ldots,$$

$$2^{l_0}\left(2^{l_1}\left(\ldots 2^{l_{s-2}}\left(2^{l_{s-1}}\left(2^{l_s}a_s + a_{s-1}\right) + a_{s-2}\right) + \ldots + a_1\right) + a_0\right),$$

длина которой равна

$$l_s + l_{s-1} + \ldots + l_0 + s + 1 = m + 2 + s - k.$$

Поэтому

$$l(n) < 2^{k-1} + 1 + m + 2 + s - k < m + 2 + \left]\frac{m+1}{k}\right[+ 2^{k-1} - k.$$

Можно считать, что $n \neq 2^m$, тогда $m + 1 =]\log_2 n[$ и

$$l(n) <] \log_2 n[(1 + 1/k) + 2^{k-1} - k + 2.$$

Следствие 3.7.1

$$\lim_{n\to\infty} l\left(n\right)/\log_2 n = 1.$$

Доказательство. Применяем доказанную теорему при

$$k = \lambda(\lambda(n)) - 2\lambda(\lambda(\lambda(n))).$$

Аддитивная цепочка называется *линейной*, если в ее графе все вершины лежат на одной ориентированной цепи. Другими словами, если каждый ее элемент равен сумме предыдущего элемента и какого-то еще. Можно ослабить требование линейности, допустив его нарушение для элементов, вычисляемых путем удвоения. Такие цепочки все равно будем называть линейными. Длина кратчайшей линейной цепочки обозначается $l^*(n)$. В предыдущей теореме фактически дана оценка как раз этой величины.

Справедлива еще одна теорема Альфреда Брауэра

$$l(2^n - 1) \leq n + l^*(n) - 1.$$

Для ее доказательства надо взять линейную цепочку

$$1 = a_0, \ldots, a_r = n, r = l^*(n),$$

и построить цепочку для $2^n - 1$, состоящую из чисел

$$2^{a_i} - 1,$$

в промежутки между которыми вставлены числа

$$2^i(2^{a_i} - 1), i = 1, \ldots, a_{i+1} - a_i.$$

Упражнение 3.7.24 Проведите подробное доказательство.

Гипотеза Шольца о том, что

$$l(2^n - 1) \leq n + l(n) - 1$$

до сих пор не доказана. Также не доказано неравенство

$$l(n) \geq \lambda(n) + \lambda(\nu(n)),$$

хотя А.Шенхаге в 1977 г. доказал, что

$$l(n) \geq \lambda(n) + \lambda(\nu(n)) - O(1).$$

Из этого неравенства и теоремы Брауэра вытекает, что

$$l(2^n - 1) - n$$

асимптотически равно $\lambda(n)$.

Если нужно быстро вычислить сразу несколько степеней, то можно построить дерево степеней таким образом. Корнем дерева является 1, единственная вершина первого уровня, из нее выходит ребро в вершину 2 второго уровня, и когда дерево уже построено до $k-$го уровня, то выбрав любое число n из этого уровня, и выписав все числа $a_1 = 1, a_1, \ldots, a_k = n$ лежащие на пути из корня до вершины n, соединяем эту вершину с вершинами $n + a_1, \ldots, n + a_k$, которые помещаем на $(k + 1)-$м уровне. В процессе построения повторно одинаковые вершины в дерево, естественно, не заносятся.

Функция $l(n)$ растет медленно, поэтому интересно рассмотреть обратную функцию $c(r)$, определяемую как $\min\{n : l(n) = r.\}$ Легко видеть, что $c(r) \leq 2^r$. Из теоремы Брауэра следует, что $\lambda(c(r)) \sim r$. Экспериментально проверено, что при $10 \leq r \leq 27$ имеет место аппроксимация $2^{r(1-1/\log_2(r)}$. Таблица первых 15 значений $c(r)$ выглядит следующим образом:

$$2, 3, 5, 7, 11, 19, 29, 47, 71, 127, 191, 379, 607, 1087, 1903.$$

Интересно, что 1903 — первое составное число в ряду ее значений. Из этой таблицы следует, например, что для $n = (2^{13} + 1)/3$ справедливо равенство $l(n) = 15$, в то время как $l(nm) = l(2^{13} + 1)) = 14 < l(n)$.

Хотя поначалу она растет примерно со скоростью чисел Фибоначчи, но потом скорость ее роста согласно теореме Брауэра возрастает.

Функцию $l(n)$ естественно обобщить, обозначив $l(n_1, \ldots, n_k)$ длину кратчайшей цепочки, содержащей все числа $n_1, \ldots, n_k$.

Р.Беллман обобщил понятие аддитивной цепочки, определив векторную аддитивную цепочку $k-$мерных векторов по аналогии с обычной аддитивной цепочкой как последовательность векторов, начинающуюся с k базисных единичных векторов, в которой каждый вектор равен сумме двух предыдущих векторов. Длина кратчайшей цепочки, содержащей вектор $(n_1, \ldots, n_k)$ (базисные векторы в ней не учитываются) обозначается $l([n_1, \ldots, n_m])$; это число очевидно равно наименьшему количеству умножений, необходимому для вычисления одночлена $x_1^{n_1} \ldots x_k^{n_k}$. Естественно предполагать далее, что все $n_i \neq 0$.

Легко обобщить бинарный метод и 2^k-арный метод Брауэра, доказав следующую теорему Э.Страуса: для любого вектора $(n_1, \ldots, n_m])$

$$l([n_1, \ldots, n_k]) \leq 2\lambda(\max\{n_i\}) + 2^k - k - 1,$$

и при любом t

$$l([n_1, \ldots, n_k]) \leq \lambda(\max\{n_i\})(1 + k/t)) + 2^t k.$$

Определенный ранее граф можно ввести и для векторных цепочек, только этот граф будет иметь не один вход (вершину, в которую не входят никакие ребра), а k входов, но, как и ранее, один выход (вершину, из которой не выходит ни одно ребро). Как и ранее, в этом графе можно вершины, из которых выходит

только одно ребро, склеить со своими «потомками,» при этом разность между числом ребер и числом вершин не меняется. Поэтому можно считать, что в полученном графе из каждой вершины (кроме выхода) выходит хотя бы два ребра (а входить может и более двух ребер), по этому графу исходная цепочка восстанавливается однозначно, причем ее длина будет на k больше разности между числом ребер в графе, и числом вершин в нем, потому что каждое ребро входит в одну вершину, а первоначально в каждую вершину, кроме k входов, входили два ребра.

По индукции легко доказать следующую лемму Н.Пиппенджера [142]: $i-$я компонента вектора, вычисляемого в данной вершине, равна числу всех ориентированных путей из i-го входа в эту вершину.

Если в графе изменить ориентацию ребер на противоположную, то получится двойственный граф с одним входом и k выходами. Из леммы Пиппенджера легко следует, что двойственный граф определяет обычную аддитивную цепочку, реализующую все компоненты $n_1, \ldots, n_k$. Эту цепочку назовем двойственной к исходной векторной цепочке. Длина этой цепочки с одним входом на единицу больше разности между числом вершин и ребер в графе, поэтому длина двойственной цепочки на $k-1$ меньше длины векторной цепочки. Очевидно, что если для двойственной цепочки построить двойственную цепочку, то получится исходная цепочка.

Отсюда вытекает следующая теорема двойственности Х.Оливоса:

$$l([n_1, \ldots, n_k]) = l(n_1, \ldots, n_k) + k - 1.$$

Так как очевидно, что

$$l(n_1 m_1 + \ldots n_k m_k) \leq l(m_1, \ldots, m_k) + l([n_1, \ldots, n_k]),$$

то из предыдущей теоремы вытекает лемма Оливоса

$$l(n_1 m_1 + \ldots n_k m_k) \leq l(m_1, \ldots, m_k) + l(n_1, \ldots, n_k) + k - 1,$$

обобщающая метод множителей, в основе которого лежит использование неравенства

$$l(nm) \leq l(n) + l(m).$$

Из теорем Страуса и Оливоса вытекает следующая теорема Э.Яо: при $n_k \rightarrow infty$

$$l(n_1, \ldots, n_k) \leq \lambda(n_k) + k\lambda(n_k)/\lambda(\lambda(n_k)) + O(\lambda(n_k)(k + \lambda(\lambda(\lambda(n_k)))))/(\lambda(\lambda(n_k)))^2,$$

нужно только в теореме Страуса выбрать $t = \lambda(\lambda(n_k)) - 2\lambda(\lambda(\lambda(n_k)))$. Из теоремы Яо следует, что если $k = O(\lambda(\lambda(n_k)))$, то

$$l(n_1, \ldots, n_k) = O(\lambda(n_k)).$$

Первоначальное доказательство теоремы Яо не использовало теорему Оливоса и было более сложным. В [24] получено также следующее уточнение теоремы Яо. Пусть $R = \log_2\left(\prod_{i=1}^{k} n_i\right), n = \max n_i$. Тогда

$$l(n_1,\ldots,n_k)+k-1 = l([n_1,\ldots,n_k]) = \log_2 n + + O(k) + \frac{R}{\log_2 R}\left(1 + O\left(\left(\frac{\log_2\log_2 R}{\log_2 R}\right)^{1/2}\right)\right).$$

Упражнение 3.7.25 Докажите, что $\nu(n)$ можно рекуррентно определить следующим образом

$$\nu(1) = 1, \nu(2n) = \nu(n), \nu(2n+1) = \nu(n) + 1,$$

а функцию $\lambda(n)$ можно рекуррентно определить следующим образом

$$\lambda(1) = 0, \lambda(2n) = \lambda(2n+1) = \lambda(n) + 1.$$

Упражнение 3.7.26 Докажите, что если аддитивная цепочка для числа n имеет длину m, то $n \leq 2^m$.

Упражнение 3.7.27 Докажите неравенство $l(n) \geq \log_2 n$ и приведите примеры, когда оно обращается в равенство.

Упражнение 3.7.28 Докажите следствие из теоремы Брауэра с оценкой

$$\log_2 n\left(1 + \frac{1}{\log_2\log_2 n} + \frac{C\log_2\log_2\log_2 n}{(\log_2\log_2 n)^2}\right),$$

где $C > 0$ - некоторая константа.

Упражнение 3.7.29 Покажите, приведя примеры, что иногда метод множителей лучше, чем бинарный метод, а иногда наоборот. Покажите, что такие примеры встречаются бесконечно часто.

Упражнение 3.7.30 Покажите, приведя примеры, что метод дерева степеней бывает лучше метода множителей и бинарного метода.

Упражнение 3.7.31 Для быстрого вычисления больших чисел Фибоначчи можно использовать следующий прием. Если уже вычислена пара чисел (F_k, F_{k-1}), то пару (F_k, F_{k+1}) находим одним сложением, а пару

$$(F_{2k}, F_{2k-1}) = (F_k^2 + 2F_kF_{k-1}, F_k^2 + F_{k-1}^2))$$

находим еще 6 операциями, и применяем бинарный метод.

Докажите, что сложность вычисления F_n не превосходит $C\log_2 n$.

Упражнение 3.7.32 Покажите, что многочлен

$$p_n(x) = 1 + x + \ldots + x^{n-1}$$

можно вычислить с помощью $l(n)$ умножений, одного вычитания и одного деления.

Упражнение 3.7.33 Покажите, что многочлен $p_n(x)$ можно вычислить с помощью $2l(n)-2$ умножений и $l(n)$ сложений.

Указание: если $x^m = x^k x^j$ — очередной шаг минимальной цепочки для вычисления x^n, то делаем шаг $p_m = p_k x^j + p_j$.

Упражнение 3.7.34 Покажите, как с помощью бинарного метода можно на простейшем калькуляторе с операцией $\sqrt{}$, но без операции возведения в степень, приближенно вычислять функцию x^y.

Упражнение 3.7.35 Можно обобщить понятие аддитивных цепочек, добавив операцию вычитания. Докажите, что для таких цепочек $l(2^n - 1) \leq n + 1$. Верно ли здесь равенство?

Эрдешем доказано однако, что при использовании вычитания все равно для почти всех n нижняя оценка для $l(n)$ имеет вид

$$\log_2 n \left(1 + \frac{1 - \epsilon_n}{\log_2 \log_2 n}\right),$$

где ϵ_n стремится к нулю.

Иногда встречается утверждение о том, что задача вычисления последовательности $l(n)$ является NP полной (см., например, [18]). Однако Д.Кнут в [39] высказывается более осторожно. Утверждается NP-полнота задачи построения не кратчайшей аддитивной цепочки, а кратчайшей векторной аддитивной цепочки, или, что равносильно, задачи вычисления $l(n_1, \ldots, n_k)$ — сложности кратчайшей аддитивной цепочки, содержащей произвольный данный набор чисел $n_1, \ldots, n_k$. Но в любом случае задача построения кратчайшей цепочки весьма сложна, поэтому для ее приближенного решения изобретаются различные эвристики. В указанном выше методе Брауэра установлена связь между $l(n)$ и числом различных отрезков длины k в двоичной записи числа n. Чем меньше число таких отрезков, тем меньше $l(n)$. Другими словами, чем менее случайной выглядит двоичная запись n, тем меньше $l(n)$. Для сжатия «неслучайных» последовательностей известно много различных архиваторов. В [170] предложен алгоритм построения близких к кратчайшим аддитивных цепочек, основанный на использовании сжатия методом Лемпеля-Зива.

3.7.11 Приложения аддитивных цепочек.

Аддитивные цепочки можно использовать для экспоненцирования (возведения в данную степень) в конечных полях и в группах эллиптических кривых. Так как возведение в квадрат в нормальных базисах конечных полей имеет очень малую сложность, а в стандартных базисах с минимальными многочленами, состоящими из трех или пяти одночленов, хотя и несколько большую, но все же незначительную по сравнении с общим умножением. Для эллиптических кривых операция удвоения точки сводится только к операциям сложения и возведения в квадрат в соответствующем поле $GF(2^r)$, и поэтому мала по сложности в сравнении со сложением различных точек кривой, так при его выполнении кроме нескольких общих умножений выполняется и существенно более сложная операция деления в конечном поле. Значит, иногда бывает естественно при

оценке сложности экспоненцирования учитывать только число нетривиальных умножений. В этом случае определим вес операции удвоения в аддитивной цепочке равным нулю.

Аддитивные цепочки с нулевым весом удвоений. Из теоремы Брауэра вытекает тогда, что

$$l(n) \le \frac{\lambda(n)}{\lambda(\lambda(n))} + \frac{O(\lambda(n)\lambda(\lambda(\lambda(n))))}{(\lambda(\lambda(n)))^2},$$

и $l(2^n-1) \le l(n)$ причем линейность цепочки не требуется, так как для любой цепочки $1 = a_0, \ldots, a_r = n, r = l(n)$, и построить цепочку для 2^n-1, состоящую из чисел

$$2^{a_i} - 1,$$

в промежутки между которыми вставлены операции умножения на степени двойки, которые сводятся к удвоениям и поэтому не учитываются, и обычные операции сложения согласно формулам

$$2^{a_i} - 1 = 2^{a_j}\,(2^{a_k} - 1) + 2^{a_j} - 1.$$

Заметим, что алгоритм Яо совместного вычисления нескольких степеней для рассматриваемого выбора весов базисных операций никакой выгоды не дает, он экономит только возведения в квадрат.

Применяя теорему Ферма, выводим из последней оценки оценку сложности инвертирования в поле $GF(2^r)$:

$$I(r) \underset{\sim}{\le} \lambda(r)M(r),$$

где $M(r)$ — сложность умножения в этом поле. Действительно, для инвертирования согласно теореме Ферма можно воспользоваться формулой $x^{-1} = x^{2^n-2} = (x^{2^{n-1}-1})^2$. Поэтому число используемых для инвертирования умножений оценивается как

$$l(2^{n-1}-1) \le l^*(n-1) \le \lambda(n-1) + \frac{\lambda(n-1)}{\lambda(\lambda(n-1))} + \frac{O(\lambda(n-1)\lambda(\lambda(\lambda(n-1))))}{(\lambda(\lambda(n-1)))^2}.$$

Например, при $r \le 127$ имеем оценку

$$I(r) \le 9M(r),$$

при $r \le 191$ имеем оценку

$$I(r) \le 10M(r),$$

при $r \le 379$ имеем оценку

$$I(r) \le 11M(r),$$

при $r \le 607$ имеем оценку

$$I(r) \le 12M(r),$$

при $r \leq 1087$ имеем оценку

$$I(r) \leq 13M(r)$$

и т.д. Напомним, что эти оценки в чистом виде применимы только в случае использования нормальных базисов.

В литературе (см., например, [130], [78]) часто указывается оценка для числа умножений, необходимых для инвертирования в поле $GF(2^n)$, в виде $\lambda(n-1)+\nu(n-1)-1$, принадлежащая Ito-Tsujii. Эта оценка достигается при использовании метода возведения в степень, основанного на следующих рекуррентых формулах:

$$x^{2^{2k}-1} = (x^{2^k-1})^{2^k} x^{2^k-1}, x^{2^{2k+1}-1} = (x^{2^{2k}-1})^2 x^{2^{2k}-1}.$$

На самом деле эта оценка получается из приведенной выше оценки Брауэра

$$l(2^{n-1}-1) \leq l(n-1),$$

если в качестве аддитивной цепочки использовать цепочку, получаемую обычным бинарным методом. Если же использовать вместо него, как выше, асимптотически наилучший метод Брауэра, то оценка Ito-Tsujii улучшается асимптотически вплоть до двух раз. Например, при $n = 191$ оценка Ito-Tsujii имеет вид $\lambda(190)+\nu(190)-1 = 7+6-1 = 12$, а на самом деле выше была указана оценка 10.

Если мы хотим построить схему для инвертирования из схем для умножения указанным выше методом, то представляет интерес задача минимизации глубины такой схемы. Минимальная глубина очевидно будет достигаться, когда глубина соответствующей аддитивной цепочки для $2^{n-1}-1$ будет минимальной. Игнорируя, как и выше, элементы удвоения, можно, повторяя доказательство теоремы Брауэра, получить, что глубина аддитивной цепочки для $2^{n-1}-1$ не превосходит глубины произвольной линейной цепочки для числа $n-1$.

Упражнение 3.7.36 Докажите это утверждение.

Минимальная глубина цепочки для n равна $\lceil \log_2 n \rceil$.

Упражнение 3.7.37 Докажите по индукции, что существует цепочка для $n \neq 2^k$ глубины $\lambda(n)+1$ и сложности $\lambda(n)(n)+\nu(n)-1$.

Указание. Используйте модификацию бинарного метода, не применяющую схему Горнера.

Например, для $n = 28$ цепочкой минимальной сложности 6 является $1, 2, 3, 6, 9, 18, 27$ но цепочка большей сложности $1, 2, 3, 4, 8, 11, 16, 27$ имеет меньшую глубину 5. По ней можно построить цепочку для вычисления $2^{27}-1$, в которой все неудваивающие шаги имеют вид

$$1, 2^2-1, 2^3-1, 2^4-1, 2^8-1, 2^{11}-1, 2^{16}-1, 2^{27}-1$$

и которая также имеет глубину 5, если игнорировать удваивающие шаги.

Все предыдущие результаты переносятся и на случай произвольного поля. В случае поля $GF(p^n), p > 2$ согласно Ito-Tsujii для инвертирования можно вычислить $y = x^{(p^n-1)/(p-1)}$, заметить, что $y \in GF(p)$ так как $y^{p-1} = x^{p^n-1} = 1$, и применить формулу

$$x^{-1} = (x^{(p^{n-1}-1)/(p-1)})^p y^{-1}$$

справедливость которой проверяется путем сравнения степеней в обоих частях равенства $-1 = p(p^{n-1}-1)/(p-1) - (p^n-1)/(p-1)$. Вычисление y^{-1} и умножение на него при малых p делается быстро, так как $y \in GF(p)$. Остается вычислить $x^{(p^{n-1}-1)/(p-1)}$ Предполагая, что возведение в степень p делается быстро (это заведомо так при использовании нормальных базисов), и игнорируя эти операции, можно перенести доказательство теоремы Брауэра на это случай и получить оценку для числа умножений при вычислении $x^{(p^{n-1}-1)/(p-1)}$ в том же виде $l(n-1)$, что и в случае $p = 2$.

Упражнение 3.7.38 Убедитесь в этом.

Для вычисления $y = x(x^{(p^{n-1}-1)/(p-1)})^p$ нужна еще одна операция умножения, но более простая, чем обычное умножение, так как в ней надо вычислить только одну компоненту произведения, ведь $y \in GF(p)$.

3.7.12 Аддитивные цепочки с вычитаниями. Использование уравновешенных позиционных систем.

В отличие от конечных полей, в рассматриваемых далее группах эллиптических кривых операция обращения делается почти бесплатно, значит вычитание делается с той же сложностью, что и сложение. Поэтому при вычислении кратных данной точки кривой имеет смысл использовать цепочки с вычитанием. Так, вычисление кратного $(2^k-1)P$ данной точки P делается примерно в k раз быстрее, чем без использования вычитания. Поэтому при вычислении кратного nP имеет смысл представить его в виде знакопеременной суммы $2^{n_1} - 2^{n_2} + 2^{n_3} - \ldots$, если число слагаемых относительно мало. Для нахождения представления в виде знакопеременной суммы нужно множество единичных битов двоичной записи числа n разбить на кластеры рядом стоящих единиц и применить тождество

$$\sum_{i=0}^{k-1} 2^i = 2^k - 1.$$

Число слагаемых в этой сумме иногда можно уменьшить, отказавшись от требования знакопеременности. Минимизация этого числа равносильна известной задаче о представлении числа в двоичной системе с использованием отрицательных битов, впервые рассмотренной в пятидесятые годы Бутом с целью уменьшения сложности умножения многоразрядных битов. Это единственное минимальное представление можно построить для любого числа n рекурсивно по формулам

$$\alpha_{2n} = \alpha_n 0, \alpha_{4n+1} = \alpha_n 01, \alpha_{4n-1} 0\mathbf{1},$$

где $alpha_0$ пусто и жирные единицы означают отрицательные биты, при этом запись α_n не содержит ни старших нулей, ни соседних ненулевых битов.

Упражнение 3.7.39 Докажите это утверждение.

Представление числа в виде суммы степеней двойки, взятыми как со знаками плюс, так и со знаками минус, равносильно его представлению в двоичной системе с использованием отрицательных цифр. Такие представления можно рассматривать и в системе по любому натуральному основанию, не меньшему двух. Если основание равно b, то любое целое число можно записать с использованием цифр, по модулю не больших $b/2$, причем для нечетных b такое представление единственно.

Упражнение 3.7.40 Докажите это.

Системы с указанным выбором цифр называются *уравновешенными*. Для двоичной системы уравновешенной системы указанный выше метод позволяет всегда найти запись (уравновешенный код), в которой не менее половины цифр будут нулями. Но такая запись тоже может быть определена неоднозначно. Укажем еще несколько алгоритмов для построения таких кодов.

Пусть число представлено в виде двоичного кода

$$a_{m-1}a_{m-2}\ldots a_1a_0.$$

Уравновешенный код вычисляется следующим алгоритмом:

Присвоить $i = 0$. $flag = 0$,

Пока $i < m$ Если $flag = 0$ и $a_i = 1$, то присвоить $c_i = -1$, $flag = 1$, если $a_i = 0$, то присвоить $c_i = 0$.

Если $flag = 1$ и $a_i = 0$, то присвоить $c_i = 1$, $flag = 0$, если $a_i = 1$ то присвоить $c_i = 0$.

Присвоить i значение $i + 1$.

Если $flag = 1$, то присвоить $c_i = 1$.

Можно использовать и смешанное представление, когда уравновешенным кодом замещаются лишь блоки единичных битов и следующий за ними нулевой бит.

Пример 3.7.4 Десятичное число 10045_{10} представляется двоичным кодом 10011100111101_2, который можно представить чистым уравновешенным кодом

$$1 - 10100 - 11000 - 11 - 1$$

или смешанным уравновешенным кодом

$$10100 - 101000 - 101.$$

Если использовать систему счисления по основанию 4, то можно записанное в обычном двоичном коде число

$$a = (a_{m-1}a_{m-2}\ldots a_1a_0)_2$$

записать в уравновешенном четверичном коде с помощью равенств

$$a = 2a - a = \sum_{i=0}^{m-1} a_i 2^{i+1} - \sum_{i=0}^{m-1} a_i 2^{i+1} =$$

$$\sum_{i=0}^{m}(a_{i-1} - a_i)2^i = \sum_{i=0}^{\lfloor m/2 \rfloor} (a_{2i-1} + a_{2i} - 2a_{2i+1})2^{2i}, a_{-1} = 0, a_m = a_{m+1} = 0.$$

Полагая $b_i = a_{2i-1} + a_{2i} - 2a_{2i+1}, i = 0, \ldots, \lfloor m/2 \rfloor$, имеем

$$a = \sum_{i=0}^{\lfloor m/2 \rfloor} b_i 4^i, -2 \le b_i \le 2.$$

Очевидно, этот код можно рассматривать также как уравновешенный двоичный, в котором число ненулевых цифр не превосходит $\lfloor m/2 \rfloor + 1$. Преимущество этого кода Бута в том, что его цифры вычисляются параллельно по трем соседним битам, и это вычисление легко организовать логической схемой глубины 2 и сложности $3(\lfloor m/2 \rfloor + 1)$. Действительно, каждая из цифр вычисляется схемой с тремя входами x, y, z и тремя выходами u, v, w по формулам $x + y - 2z = u + 2v - 4w$, где $u = x \oplus y, v = xy \oplus z, w = z$, где цифры $0, 1, 2$ записываются в виде $000, 001, 010$, а цифры $-1, -2$ записываются в виде $111, 110$. Чтобы перейти к записи по основанию два, достаточно в каждой цифре откинуть правый бит, и если он нулевой, то добавить единицу к удвоенному номеру этой цифры, чтобы получить номер соответствующей цифры в двоичной записи; если же он единичный, то номер цифры просто удваивается. При этом запись 11 будет означать -1, а записи $00, 01$ соответственно $0, 1$.

Указанные приемы можно применять и в случае конечных полей, если надо выполнять экспоненцирование примитивного элемента поля, при этом нужно заранее вычислить его обратный элемент, и тогда операцию вычитания можно заменить на сложение. Эти приемы позволяют ускорять экспоненцирование в условиях сильно ограниченной памяти. Если память не так ограничена, большего ускорения можно добиться, применяя 2^k-арный метод Брауэра, в котором вначале вычисляются и запоминаются $2^k - 1$ элементов цепочки. Из доказательства теоремы Брауэра видно, что на самом деле надо запоминать только нечетные степени, что вдвое экономит память и немного ускоряет вычисления. Но если предварительно показатель степени записать в уравновешенной системе, и воспользоваться, как и выше, возможностью вычитания, то память уменьшается еще вдвое.

3.7.13 Алгоритмы с фиксированной базой

При оптимизации алгоритмов экспоненцирования в случае их многократного применения в одной программе полезно выделять два несколько различающихся случая: когда база g экспоненцирования фиксирована, а показатель n

случаен, и когда база случайна, а показатель фиксирован. Первый случай, например имеет место в алгоритме генерации общего секретного ключа Диффи-Хеллмана, а второй — в алгоритме RSA или ЭльГамаля.

В первом случае можно ускорить экспоненцирование, заранее вычислив и сохранив в памяти некоторые фиксированные степени $\{g^{m_1}, \ldots, g^{m_k}\}$ Тогда если числа $0 \leq n_i \leq h$, то для вычисления $a = g^n$ при $n = n_1 m_1 + \ldots n_k m_k$ в [88] Brickell,Gordon,McCurley,Wilson предложили следующий алгоритм, требующий $k + h - 2$ операций:

$$b \leftarrow 1$$

$$a \leftarrow 1$$

$$for\ d = h\ to\ 1\ by\ -1$$

$$while\ a_i = d$$

$$b \leftarrow b * g^{m_i}$$

$$a \leftarrow a * b$$

$$return\ a.$$

Для обоснования алгоритма положим

$$c_d = \prod_{i, a_i = d} g^{m_i},$$

тогда после i-го шага цикла имеем

$$b = c_h \ldots c_{h-i+1}, a = c_h^i c_{h-1}^{i-1} \ldots c_{h-i+1},$$

и окончательно

$$a = \prod_{d=1}^{n} c_d^d.$$

На самом деле этот алгоритм эквивалентен алгоритму Яо, а также равносилен применению леммы Оливоса, которая дает неравенство

$$l(n_1 m_1 + \ldots n_k m_k) \leq l(m_1, \ldots, m_k) + l(n_1, \ldots, n_k) + k - 1 \leq$$

$$\leq l(m_1, \ldots, m_k) + l(1, 2 \ldots, h) + k - 1 \leq l(m_1, \ldots, m_k) + k + h - 2.$$

Конечно, это неравенство можно использовать и без предположения о предварительном вычислении степеней $(m_1, \ldots, m_k)$, например при выборе $m_{i+1} = 2^{i\lceil \lambda(n)/k \rceil}$ и $h = 2^{\lceil \lambda(n)/k \rceil} - 1$ фактически получаем $2^{\lceil \lambda(n)/k \rceil}$-арный метод. Очевидную оценку $l(n_1, \ldots, n_k) \leq h - 1$ иногда можно улучшить, используя некоторую комбинацию метода степенного дерева с методом множителей, фактически совпадающую с методом, известным как метод Евклида. Сначала упорядочим массив $(n_1, \ldots, n_k)$, удалив из него повторяющиеся элементы, и так делаем после каждого шага. Каждый шаг алгоритма приводит к неравенству

$$l(n_1, \ldots, n_{k-1}, n_k) \leq l(n_1, \ldots, n_{k-1}, r_k) + l(q_k) + 1,$$

где $n_k = n_{k-1}q_k + r_k$. Алгоритм заканчивает работу, когда система $(n_1, \ldots, n_{k-1}, n_k)$ превратится в одно число d, равное НОД этой системы. Алгоритм можно усложнить, выбирая на каждом шаге вместо деления n_k на n_{k-1} такое деление n_k на n_r, при котором $n_k = n_{k-1}q'_k + r'_k$, $l(q'\_k) < l(q_k))$ или $l(q'\_k) = l(q_k))$, $r'_k \in \{n_1, \ldots, n_{k-1}\}$. Ясно, что применять это алгоритм имеет смысл только при достаточно больших k, так как из теоремы Яо следует, что при $k = O(\lambda(\lambda(n_k)))$ получается и без того хорошая оценка

$$l(n_1, \ldots, n_k) = O(\lambda(n_k)).$$

Приведем пример применения указанного алгоритма с использованием предварительно вычисленной таблицы степеней. Пусть необходимо вычислить любую из степеней g^n фиксированного элемента при $n < 2^{1024}$ (задача возведения в степени такой величины возникает в протоколе Диффи-Хеллмана при использовании поля $GF(2^{1024})$). Выберем $k = \lfloor 1024/7 \rfloor = 146$, $h = 2^7 - 1$. Тогда при использовании заранее вычисленной таблицы степеней

$$g^{m_1}, \ldots g^{m_{128}}, m_i = 2^{(i-1)7}, i = 1, \ldots 147,$$

вычисление произвольной степени g^n указанным выше методом требует не более $k + h - 2 = 272$ операции умножения. Так как длина записи произвольного элемента поля $GF(2^{1024})$ равна 128 байт, то объем используемой таблицы равен 19Кб и она вполне умещается в оперативной памяти. Поэтому операция извлечения элемента из этой таблицы выполняется быстро и временем ее выполнения можно пренебречь в сравнении с временем умножения элементов поля. Для сравнения заметим, что алгоритм возведения в степень порядка 2^{1000}, не использующий предварительно вычисленной таблицы, использует не менее 1000 операций умножения в лучшем случае.

В общем случае можно в качестве показателей степеней $\{m_1, \ldots, m_k\}$ взять последовательность $2^{im/k}, i = 0, \ldots k-1, m = \lceil \log_2 n \rceil$, $k = m/(\log_2 m - 2\log_2 \log_2 m) + O(1)$ и выбрать $h = m/(logm)^2 + O(1)$. Тогда оценка сложности имеет вид $k + h - 2$ и асимптотически равна $m/\log_2 m$, как и в алгоритме Брауэра.

Brickell et al рассмотрели много разных вариантов выбора $(m_1, \ldots, m_k)$, в том числе вариант $\{\pm b^i, i = 1, \ldots, k\}$, означающий использование уравновешенной b-ичной системы счисления. Этот вариант удобен тем, что уменьшает величину h примерно в два раза. Например, для вычисления 512-битной экспоненты можно взять $b = 45$, тогда $h = \lceil (b-1)/2 \rceil = 22$, так как используются «цифры» $n_i \in [0, h]$ и формула

$$n = n_1 \pm m_1 + \ldots n_k \pm m_k, m_i = b^i, \ldots, k, k = \lceil \log_b n \rceil \leq 94,$$

но реально слагаемые с нулевыми цифрами отсутствуют, а элементы $g^{\pm m_i}$ заранее вычисляются и запоминаются. Число выполняемых при экспоненцировании

операций при этом не больше $k+h-2=118$, а в среднем меньше, при этом запоминаются $2k=188$ элементов.

Если нет ограничения на объем используемой памяти, то можно максимально уменьшить число умножений, выбрав $h=1$, т.е. используя в качестве $\{m_i\}$ набор $a\cdot b^i, a<b, i<s=\lceil\log_b n\rceil$, тогда все n_i просто равны 1 или 0, причем число единиц не не более $k=s$, значит число умножений не более $s-1$, но запоминать требуется $(b-1)s$ элементов.

Упражнение 3.7.41 Докажите, что в среднем число умножений в этом варианте алгоритма равно $s(b-1)/b$.

При 512-битном n, выбрав $b=256$, получим алгоритм, использующий 63 умножения и требующий запоминания 16320 элементов (т.е. более мегабайта памяти). Понятно, что увеличивать b с целью уменьшить число умножений здесь уже не разумно.

Можно уменьшить расходуемую память, выбрав при той же базе b $h=2$, «цифры» $n_i\in 0,1,2$, а в качестве $\{m_i\}$ взять набор $d\cdot b^i, i<s$, где d любое число, кратное 2^{2j}, но не кратные $2^{2j+1}, j=0,1,2,\ldots$. Число умножений в худшем случае увеличится до s, но память уменьшится в $1/2+1/8+1/32+\ldots=2/3$ раза. Продолжая далее, можно уменьшить b в два раза, выбрать $h=3$, в качестве «цифр» использовать $0,1,2,3$ а в качестве $\{m_i\}$ взять набор $d\cdot b^i, i<s$, где d любое число, разложение которого на множители имеет вид $2^\alpha 3^\beta r, (r,6)=1, \alpha+\beta=2j, j=0,1,\ldots$. Тогда число умножений увеличится в худшем случае достигнет $s+1$, а запоминать потребуется Cbs элементов, где

$$C=(1+1/6)\frac{1}{3}(1+(1/9+1/81+\ldots)+(1/4+1/16+\ldots))=\frac{7}{36}(9/8+1/3)=\frac{245}{864}=0.284.$$

Упражнение 3.7.42 Докажите это.

В [88] также доказано, что если число запоминаемых элементов ограничить $(\log\log n)^k$, то число используемых умножений будет не больше

$$(1/k+o(1))\log_2 n/\log_2\log_2 n.$$

В [124] предложен еще один, во многих случаях более удобный, алгоритм экспоненцирования фиксированного основания, также использующий предварительные вычисления, результаты которых заносятся в память перед вычислением. Впрочем, этот алгоритм содержится как частный случай в известном алгоритме Пиппенджера [142]. Заключается алгоритм [124] в следующем: показатель степени $a<2^n$ записывается 2^k-ичной системе

$$a=\sum_{i=0}^{s-1}A_i 2^{ki},$$

где A_i имеет двоичную запись $(a_{ik+k-1}\ldots a_{ik})_2$, $s=\lceil n/k\rceil$, степени базы g, записываемые далее для удобства в аддитивной нотации, предвычисляются в порядке $2^{ki}g, 0\le i<s$, потом для каждого двоичного вектора $u=(u_{s-1}\ldots u_0)_2, 0\le$

$u < 2^s$ вычисляются в порядке возрастания чисел $u_{s-1} + \ldots + u_0$) суммы

$$P_u = \sum_{i=0}^{s-1} u_i 2^{ki} g,$$

а потом вычисляется степень $a \cdot g$ по формуле

$$a \cdot g = \sum_{j=0}^{k-1} 2^j \left(\sum_{i=0}^{s-1} a_{ik+j} 2^{ki} \cdot g \right) = \sum_{j=0}^{k-1} 2^j P_{I_j},$$

где $I_j = (a_{(s-1)k+j} \ldots a_{k+j} a_j)_2$. Преимущество этой формулы перед стандартной формулой

$$a \cdot g = \sum_{j=0}^{j=s-1} 2^{kj} \left(\sum_{i=0}^{k-1} a_{jk+i} 2^i \cdot g \right)$$

в том, что при условии предвычисления внутренних сумм упрощается вычисление степени $a \cdot g$ за счет уменьшения числа возведений в квадрат (удвоений в аддитивной нотации), которых становится только k вместо $(s-1)k$ во второй формуле. Однако само предвычисление становится более сложным и требует не менее $n - k$, но не более n операций возведения в квадрат и $2^s - s - 1$ операций умножения. При выборе $s = \log_2 n - 2 \log_2 \log_2 n$ получается такая же асимптотическая оценка, что и в методе Брауэра. Также как и метод Брауэра, этот алгоритм использует схему Горнера, но с основанием 2 вместо 2^k, но при этом вместо 2^k-ичных цифр,

$$\sum_{i=0}^{k-1} a_{jk+i} 2^i$$

определяемых блоками битов длины k номера m которых удовлетворяют условию $\lfloor m/k \rfloor = j$ (частное от деления номера m на k равно j) и оказываются соседними, он использует «цифры» вида

$$\sum_{i=0}^{s-1} a_{ik+j} 2^{ki},$$

определяемые блоками битов длины s, номера которых при делении на k дают одинаковый остаток j.

В случае фиксированного основания g результаты предвычислений можно загрузить в память машины в начале работы программы, как и в алгоритме [88]. Для этого требуется объем памяти $O(2^s n)$, если предполагать что для записи любой степени элемента g достаточно n бит; в случае применения алгоритма к конечным полям или группам это естественное предположение. Заметим, что в предыдущем алгоритме при том же выборе параметра k памяти требовалось меньше, а именно $O(sn)$. Зато время работы уменьшается, и становится равным по порядку $k = n/s$ против $2^k + s = 2^k + n/k$. Но при подходящем выборе параметров оба алгоритма имеют время работы $O(n/logn)$, но памяти последний

алгоритм в этом случае расходует меньше, а именно $O(n^2/log_n^2)$. Если объем используемой памяти ограничен величиной $log^k n$ (считаем, что на запоминание одного элемента расходуется единица памяти), то время работы будет равно $O(n/(k \log\log n))$, что теоретически меньше при тех же ограничениях, чем в алгоритме [88].

3.7.14 Ускорение проверки электронной подписи

Во многих алгоритмах электронной подписи для ее верификации надо вычислять либо произведение двух экспонент $g^a f^b$ в случае полей, либо сумму $aP+bQ$ в случае эллиптических кривых. Это вычисление можно ускорить применяя так называемый «трюк Шамира». На самом деле он является просто частным случаем описанного выше метода Страуса вычисления одночлена нескольких переменных (в данном случае — двух). Заключается он в следующем. Числа $a, b < 2^n$ записываются в системе счисления по основанию 2^k. Можно воспользоваться и уравновешенными кодами, что мы далее и предполагаем в случае использования эллиптических кривых. Вычисляем заранее все суммы вида $iP \pm jQ, i, j \leq 2^{k-1}$. После этого просто вычисляем по схеме Горнера сумму

$$aP + bQ = \sum_{i=0}^{\lceil n/k \rceil} \pm(\pm a_i P + \pm b_i Q) 2^{ki}$$

умножая каждый раз полученный результат на 2^k и прибавляя или вычитая выбранное с подходящим знаком слагаемое $\pm a_i P + \pm b_i Q$, уже вычисленное заранее. Алгоритм выполняет n/k сложений и n удвоений. По сравнению с раздельным вычислением число сложений уменьшается в два раза. Алгоритм может быть ускорен теми же методами, что и алгоритмы вычисления кратного одной точки, но на практике это бесполезно, так как в алгоритмах цифровой подписи из двух точек P, Q заранее известна только одна. Некоторой экономии памяти можно достичь, если заранее вычислять только суммы вида $iP \pm jQ$ с i, j разной четности.

3.7.15 Метод Монтгомери быстрого экспоненцирования

Сложность обычного умножения многочленов степени меньшей n по модулю произвольного многочлена степени n оценивается как $M(n) + D(n)$, где $D(n)$ — сложность деления многочлена степени $2n$ на многочлен степени n. При использовании школьных алгоритмов умножения и деления многочленов $M(n)$ и $D(n)$ приблизительно совпадают.

Но при больших n при использовании быстрых алгоритмов умножения школьный алгоритм деления будет сильно тормозить модулярное умножение.

Лишь в некоторых случаях, например, когда модуль состоит из малого числа одночленов, получается для него хорошая оценка $M(n) + O(n)$.

Известен, однако быстрый алгоритм деления, в котором оно фактически сводится к многократно выполняемому умножению многочленов, степени которых убывают в геометрической прогрессии.

С использованием этого алгоритма можно оценить сложность модулярного умножения в случае произвольного модуля как $6M(n) + O(n)$.

Отсюда в общем случае вытекает оценка $l(m)(6M(n) + O(n))$ для сложности модулярного экспоненцирования, где $l(m)$— длина кратчайшей аддитивной цепочки для возведения в m-ю степень.

Можно эту оценку улучшить до

$$(l(m) + 1)(3M(n) + O(n)) + 6M(n),$$

но для этого надо более глубоко вникнуть в алгоритм быстрого деления, и воспользоваться тем, что во всех делениях делитель — один и тот же.

Другой способ получения указанной оценки — применение метода Монтгомери для *быстрого модулярного экспоненцирования*. Он интересен тем, что почти не использует операцию деления с остатком, без которой казалось бы невозможны модулярные вычисления.

Пусть $p(x)$- многочлен степени n. Будем рассматривать операции умножения по модулю этого многочлена. Положим $R = x^{n-1}$. Так как R и p взаимно просты, то можно заранее вычислить многочлен q степени $n-2$ такой, что $qp+1$ будет кратен R. Пусть Q таков, что $QR = 1 \pmod p$.

Сложность предварительного вычисления многочленов Q и q, зависящих только от $p(x)$, может быть оценена с помощью алгоритма Евклида-Шенхаге как $O(M(n)\log n)$, а с помощью обычного алгоритма Евклида — как $O(n^2)$.

Упражнение 3.7.43 Докажите, что

$$\frac{f + ((f \pmod R)q \pmod R)p}{R} = fQ \pmod p.$$

Операция перехода от f к

$$fQ \pmod p = \frac{f + ((f \pmod R)q \pmod R)p}{R}$$

называется *редуцированием Монтгомери*, а операция

$$Mont(f,g) = fgQ \pmod p = \frac{fg + ((fg \pmod R)q \pmod R)p}{R}$$

— *умножением Монтгомери* (его можно производить над любыми многочленами f, g степени меньше n). Так как $R = x^{n-1}$, то в этих операциях деление фактически не производится, а производятся только умножения.

Упражнение 3.7.44 Докажите, что сложность выполнения умножения Монтгомери не превосходит $3M(n)+O(n)$, а сложность возведения в квадрат не превосходит $K(n)+2M(n)+O(n)$, где $K(n)$— сложность обычного возведения многочлена в квадрат, если не учитывать сложность предварительного вычисления многочленов Q и q, зависящих только от $p(x)$.

Поясним как используется умножение Монтгомери для возведения в степень. Для этого сначала вычисляется

$$f_M = Mont(f, R^2) = fR \bmod p,$$

а потом он возводится в степень m с помощью кратчайшей аддитивной цепочки, но вместо обычных умножений используется умножение Монтгомери. Так как $Mont(gR, hR) = fgR$, то в результате получается $f_M^m = f^m R \bmod p$, после чего остается вычислить

$$Mont(f_M^m, 1) = f^m \bmod p,$$

т.е. выполнить редукцию по Монтгомери.

Упражнение 3.7.45 Докажите, что сложность возведения в данную степень m по модулю многочлена $p(x)$ произвольного многочлена степени меньшей n методом Монтгомери оценивается как

$$(l(m)+1)(3M(n)+O(n))+6M(n).$$

Все предыдущие результаты переносятся почти без изменений и на *модулярную числовую арифметику.*

Пусть p и R — произвольные взаимно простые числа. Будем рассматривать операции умножения по модулям p и R. Так как R и p взаимно просты, то можно заранее вычислить число q такое, что $qp+1$ делится на R. Пусть Q таково, что $QR-1$ делится на p.

Упражнение 3.7.46 Докажите, что для любого числа $0 \le x < pR$

$$\frac{x + ((x \bmod R)q \bmod R)p}{R}$$

равно либо $xQ \bmod p$, либо больше этого числа на p.

Если для записи чисел используется позиционная система счисления с основанием b и применяются как стандартные так и быстрые арифметические алгоритмы, то алгоритм проведения редукции по Монтгомери удобнее выполнять для $R = b^n$.

Пусть $x = (x_{2n-1} \dots x_0)_b < pR$, $p = (m_{n-1} \dots m_0)_b, m_0 \neq 0$, «цифра» Q такова, что $Qm_0 + 1$ кратно b. Тогда $xQ \bmod p$ можно вычислить следующим образом.

Упражнение 3.7.47 Положим вначале $a = x$, и, обозначая всегда далее

$$a = (a_{2n-1} \dots a_1)_b,$$

для каждого i от нуля до $n-1$ вычисляем «цифру» $u_i = a_i Q \bmod b$, а потом прибавляем к текущему значению a число $u_i p b^i$, в конце этого вычисления младшие n цифр числа a оказываются нулевыми и их просто «стираем», полагая a равным a/b^n, и в завершение, если полученное значение a будет не меньше p, отнимаем от него p. Докажите, что полученный результат всегда равен $xQ \bmod p$.

В указанном выше случае умножение по Монтгомери чисел x и y $Mont(x, y) = xyQ \bmod p$ удобно выполнять следующим алгоритмом. Пусть $0 \le x = (x_{n-1} \dots x_1)_b < p$, $0 \le y = (x_{n-1} \dots x_1)_b < p$.

Упражнение 3.7.48 Положим вначале $a = 0$, и, обозначая всегда далее

$$a = (a_{n-1} \dots a_1)_b,$$

для каждого i от нуля до $n-1$ вычисляем «цифру» $u_i = (a_0 + x_i y_0)Q \bmod b$, а потом прибавляем к текущему значению a числа $u_i p$ и $x_i y$, и так как после этого младшая цифра числа a оказывается нулевой, ее «стираем», полагая a равным a/b, и в завершение, если полученное значение a будет не меньше p, отнимаем от него p. Докажите, что полученный результат всегда равен $xyQ \bmod p$.

Упражнение 3.7.49 Докажите, что для чисел сложность выполнения умножения Монтгомери не превосходит $3M(n) + O(n)$, если не учитывать сложность предварительного вычисления чисел Q и q, зависящих только от числа p.

Упражнение 3.7.50 Докажите, что сложность возведения в данную степень m по модулю данного n-значного числа p произвольного n-значного числа методом Монтгомери оценивается как

$$(l(m) + 1)(3M(n) + O(n)) + 6M(n).$$

3.7.16 Пример реализации метода Монтгомери логическими схемами

Кратко опишем этот пример, пользуясь работой [163]. Разумеется, можно его имплементировать и программным образом. Как известно при схемной реализации арифметики в поле $GF(2^n)$ в стандартных полиномиальных базисах удобно в качестве неприводимого многочлена выбирать трехчлен $f(x) = x^m + x^k + 1$, если он существует. Далее рассматривается только этот случай. Можно при этом предполагать, что $k \geq m/2$. Вычислим заранее многочлены $r(x), r(x)', f'(x)$ такие, что

$$r(x)r'(x) + f(x)f'(x) = 1.$$

Многочлены a, b рассматриваем как элементы поля $GF(2^m)$. Ясно, что $r' = r^{-1} \bmod f$. Умножение по Монтгомери — это вычисление многочлена $c(x) = a(x)b(x)r^{-1}(x) \bmod f(x)$. Оно выполняется следующим образом: находим $t(x) = a(x)b(x)$, потом $u(x) = t(x)f'(x) \bmod r(x)$, и наконец $c(x) = (t(x) + u(x)f(x))/r(x)$. Для того, чтобы деление по модулю r стало тривиальным, выбирается $r = x^k$. В случае $k \leq m-1$ и $\deg c(x) \geq m$ многочлен $c(x)$ надо еще привести по модулю $f(x)$. Вычисляем $r'(x) = r^{-1}(x) = 1 + x^{m-k}$, $f' = 1$. При перемножении $a(x) = \sum\limits_{i=0}^{m-1} a_i x^i$, $b(x) = \sum\limits_{i=0}^{m-1} b_i x^i$, $a_i, b_i = 0, 1$ получаем $t(x) = \sum\limits_{i=0}^{2m-2} t_i x^i$, где при $i < m$

$$t_i = \sum_{j=0}^{i} a_j b_{i-j},$$

а при $i \geq m$

$$t_i = \sum_{j=i-m+1}^{m-1} a_j b_{i-j}.$$

Далее находим

$$u(x) = t(x)f'(x) \bmod r(x) = t(x) \bmod x^k = \sum_{i=0}^{k-1} t_i x_i.$$

Положим $t_L(x) = \sum\limits_{i=0}^{k-1} t_i x_i$, $t_R(x) = \sum\limits_{i=0}^{2m-k-2} t_{i+k} x_i$, тогда $t(x) = t_L(x) + x^k t_R(x), u(x) = t_L(x)$,

$$c(x) = (t(x) + u(x)f(x))/r(x) = t_R(x) + (x^{m-k} + 1)t_L(x).$$

При $k = m - 1$ $\deg c(x) \le m - 1$, приведение по модулю f не требуется поэтому при $k = m - 1$ окончательно имеем

$$c_0 = t_0 + t_{m-1}, c_{m-1} = t_{m-2} + t_{2m-2}, c_i = t_{i-1} + t_i + t_{m-1+i}, 0 < i < m - 1.$$

При $m/2 \le k < m - 1$ положим

$$t_R^{(1)}(x) = \sum_{i=0}^{m-1} t_{k+i} x^i,$$

$$t_R^{(2)}(x) = \sum_{i=m}^{2m-k-2} t_{k+i} x^i,$$

тогда имеем

$$c(x) \bmod f(x) = t_R(x) + (x^{m-k} + 1)t_L(x) \bmod f(x) =$$

$$= t_R^{(1)}(x) + t_R^{(2)}(x) + (x^{m-k} + 1)t_L(x) \bmod f(x) =$$

$$(t_R^{(1)}(x) + x^{m-k}t_L(x) + t_L(x)) + t_R^{(2)}(x) \bmod f(x).$$

Так как

$$t_{k+i+m} x^{i+m} \bmod f(x) = t_{k+i+m}(x^i + x^{i+k}), i = 0, \ldots, m - k - 2,$$

то

$$t_R^{(2)}(x) \bmod f(x) = t_R^{(2,1)}(x) + t_R^{(2,2)}(x),$$

где

$$t_R^{(2,1)}(x) = \sum_{i=0}^{m-k-2} t_{m+k+i} x^i,$$

$$t_R^{(2,2)}(x) = \sum_{i=k}^{m-2} t_{m+i} x^i.$$

Следовательно,

$$c(x) = t_R^{(1)}(x) + x^{m-k}t_L(x) + t_L(x)) + t_R^{(2,1)}(x) + t_R^{(2,2)}(x).$$

При $(m+1)/2 < k < m-1$ отсюда имеем формулы для коэффициентов в явном виде

$$c_i = t_i + t_{i+k} + t_{i+k+m}, i \le m-2-k, c_{m-k-1} = t_{m-k-1} + t_{m-1},$$

$$c_i = t_i + t_{i-m+k} + t_{i+k}, m-k \le i \le k-1,$$

$$c_i = t_{i+k-m} + t_{i+k} + t_{i+m}, k \le i \le m-2, c_{m-1} = t_{k-1} + t_{m+k-1}.$$

Следовательно вычисление c_i по t_j выполняется схемой, состоящей из $2m-2$ операций сложения по модулю 2 (операций XOR) и имеющей глубину 2.

Упражнение 3.7.51 Проверьте, что в случае $k = (m+1)/2$ схема имеет такие же характеристики.

Упражнение 3.7.52 Проверьте, что в случае $k = m/2$ схема имеет сложность $3m/2-1$ и глубину 2.

Учитывая, что схема для вычисления t_i состояла из m^2 конъюнкторов и m^2-1 XOR, и имела глубину $\lceil \log_2 m \rceil$, окончательно имеем, что построенный мультиплер Монтгомери при $k > m/2$ состоит из m^2 конъюнкторов и $(m+1)^2-3$ элементов XOR и имеет глубину $\lceil \log_2 m \rceil + 2$. В случае $k = m/2$ экономится $m-1$ XOR.

Упражнение 3.7.53 Постройте обычный стандартный мультиплер и сравните его характеристики с мультиплером Монтгомери.

Разницы почти никакой.

3.7.17 Одновременное вычисление нескольких степеней в конечном поле

Допустим, что надо вычислить в поле $GF(2^n)$ k разных степеней $f^{n_i}, i = 1, \ldots, k, maxn_i = m$. Используя теорему Яо, можно получить оценку сложности этого вычисления

$$K(n)\log m + k(1+o(1))\frac{M(n)\log m}{\log\log m},$$

где $K(n)$— сложность возведения в квадрат в поле $GF(2^n)$, а $\mathbf{M}(n)$ — сложность умножения в этом поле.

Оценка $K(n)$ имеет вид $D(n)+O(n)$, где $D(n)$ сложность деления многочлена степени $2n$ на многочлен степени n, так как возведение в квадрат двоичного многочлена делается со сложностью $O(n)$. В случае, если $q(x)$ является малочленом, имеем оценку $K(n) = O(n)$. В общем случае оценка имеет вид $K(n) = 5M(n)+O(n)$, где $M(n)$ — сложность умножения двоичных многочленов степени n. Сложность умножения в поле $GF(2^n)$ оценивается как $D(n)+M(n)$, и в случае, когда $q(x)$ является малочленом, имеем оценку $M(n)+O(n)$, а в общем случае — оценку $6M(n)+O(n)$, где $M(n)$ — сложность умножения двоичных многочленов степени n.

В случае, когда $q(x)$ является малочленом, оценка сложности совместного вычисления нескольких степеней имеет вид

$$O(n)\log m + k(1+o(1))\frac{M(n)\log m}{\log\log m}.$$

В общем случае оценка получается несколько хуже, но если применить метод Монтгомери, то она улучшается до асимптотической оценки

$$2M(n)\log m + 3k\frac{M(n)\log m}{\log\log m}.$$

3.7.18 Оценка сложности теста на примитивность в конечном поле

Для проверки, является ли элемент x примитивным элементом поля $GF(2^n)$, как известно, достаточно проверить для любого простого делителя s числа 2^n-1 что $x^{(2^n-1)/s} \bmod q(x)$ не равен 1, где $q(x)$ — неприводимый многочлен, определяющий рассматриваемую реализацию поля. Другой вариант этой задачи — тестирование на примитивность данного многочлена $p(x)$ (как известно, многочлен степени n примитивен тогда и только тогда, когда все его корни в поле $GF(2^n)$ являются примитивными элементами). Тогда в качестве модуля $q(x)$ нужно просто взять $p(x)$.

Количество простых делителей у числа 2^n-1 не превосходит, как известно, по порядку $n/\log n$, а как правило, гораздо меньше. Предположим, что они нам известны, и оценим сложность остальной процедуры.

Используя оценку сложности совместного вычисления степеней, получаем оценку сложности вычисления упомянутой системы

$$O(n^2) + k(1+o(1))\frac{M(n)n}{\log_2 n},$$

в случае, когда $q(x)$ — малочлен, и в общем случае — асимптотическую оценку

$$2M(n)n + 3k\frac{M(n)n}{\log_2 n},$$

где величину k можно оценить как $n/\log n$, но в среднем она гораздо меньше, однако есть примеры, когда она не меньше $\log n/\log\log n$. В итоге замечаем, что рассматриваемая часть алгоритма имеет менее чем кубическую сложность, а во многих случаях она может иметь вид

$$2M(n)n + O(\frac{M(n)n}{\log\log n})$$

и быть даже меньше, вплоть до

$$2M(n)n + O(\frac{M(n)n}{\log n}),$$

когда $2^n - 1$ является простым числом (числом Мерсенна) или имеет малое число простых делителей.

3.7.19 Быстрое экспоненцирование через модулярную композицию

Оценка сложности возведения многочлена степени меньшей n в степень m по модулю данного многочлена степени n, полученная выше в виде

$$(l(m)+1)(3M(n)+O(n))+6M(n) = 3l(m)M(n)+9M(n)+O(nl(m))$$

пригодна для любых полей, но для полей конечной характеристики p и для модулей, содержащих малое число слагаемых, слишком груба. Для этих полей модулярное возведение в степень p, как многократно отмечалось ранее, выполняется быстрее, чем с очевидной оценкой $O(logp)M(n)$. Действительно, простое возведение в степень p делается бесплатно, а приведение многочлена степени pn по модулю данного малочлена выполняется с линейной сложностью. Пользуясь этим соображением, ранее была получена оценка сложности возведения в степень m многочленов над полем $GF(q)$ по модулю данного «малочлена» степени n в виде

$$E(n,m) = O(M(GF(q)))M(n)\log_2 m/\log_2\log_2 m).$$

В [100] в случае $q = 2$ фактически та же оценка была доказана без предположения о «малочленности» многочлена f, по модулю которого вычисляется g^m. Мы ее изложим, подсчитав все мультипликативные константы. Так как ее применение ориентировано на экспоненцирование в конечных полях, далее предполагаем, что $m < q^n, \log_q m \to \infty, n \to \infty$. Идея алгоритма состоит в использовании модулярной композиции. А именно, операция $g^{q^m} \bmod f$ может быть выполнена как $g(h) \bmod f$, где $h = x^{q^m} \bmod f$. Действительно, если

$$g(x) = \sum_{i=0}^{s} a_i x^i,$$

то

$$g^{q^m}(x) = \sum_{i=0}^{s} a_i^{q^m} x^{q^m i} = \sum_{i=0}^{s} a_i x^{q^m i} \bmod f = \sum_{i=0}^{s} a_i h^i \bmod f = g(h) \bmod f.$$

Положим $k = \lceil \log_q m/\log_q\log_q m\rceil = \log_2 m/\log_2\log_q m + O(1)$ и разложим m в системе счисления по основанию q^k, а именно

$$m = \sum_{i=0}^{s-1} m_i q^{ki}, 0 \le m_i < q^k, s = \lceil \log_q\log_q m\rceil.$$

Вычислим $h = x^{q^k} \bmod f$ путем k-кратного модулярного возведения в степень q с применением метода Монтгомери со сложностью

$$k(3l(q)M(n)+9M(n)+O(nl(q))M(GF(q)) =$$

$$= (3 + o(1))M(GF(q))M(n)\log_2 q \log_2 m/\log_2\log_2 m.$$

Применяя s-кратно изложенный выше алгоритм Brickell et al с вычисленными заранее степенями

$$g^{q^{ia}} \bmod f, i = 0, \ldots b-1, b = \lceil k/a \rceil, a = \lfloor \log_q h \rfloor, h = \frac{k\log_2 q}{\log_2^2(k\log_2 q)}$$

вычисляем $g_i = g^{m_i} \bmod f, i < s$ выполнив

$$s(b+h) = s\left(\frac{k\log_2 q}{(1+o(1))\log_2(k\log_2 q)} + h\right) = s\left(\frac{(1+o(1))k\log_2 q}{\log_2(k\log_2 q)}\right) =$$

$$(1+o(1))\log_2 m/log_2\log_2 m$$

модулярных умножений многочленов над полем $GF(q)$, т.е. со сложностью $(1+o(1))6M(n)M(GF(q))\log_2 m/log_2\log_2 m$. Применяя метод Монтгомери, оценку можно улучшить до $(3+o(1))M(n)M(GF(q))\log_2 m/log_2\log_2 m$.

Сложность предварительных вычислений $g^{q^{ia}} \bmod f, i = 0, \ldots b-1$ путем $a(b-1) < k$-кратного модулярного возведения в степень q с применением метода Монтгомери оценивается как

$$(3 + o(1))M(GF(q))M(n)k =$$

$$= (3 + o(1))M(GF(q))M(n)\log_2 q \log_2 m/\log_2\log_2 m.$$

Теперь вычисляем $c = g^m \bmod f$, применяя схему Горнера. Для этого полагаем вначале $c = 1$, а потом в цикле повторяем

$$c = c^{q^k} g_i = c(h)g_i \bmod f, i = s-1, \ldots, 0.$$

При этом выполняется s модулярных умножений со сложностью $s(6M(n) + O(n)M(GF(q))$, но если заранее вычислить частное от деления x^{2n} на f, то можно упоминавшимся ранее приемом уменьшить сложность до

$$3sM(n)M(GF(q)) = 3M(n)M(GF(q))\log_q\log_q m.$$

Но основная сложность приходится на s модулярных композиций, которая равна

$$s\left(6M(n)n^{1/2} + O(n^{(\omega+1)/2})\right)M(GF(q)).$$

Окончательная оценка имеет вид

$$\log_q\log_q m\left(6M(n)n^{1/2}M(GF(q)) + O(n^{(\omega+1)/2})\right) +$$

$$+(3 + o(1))(\log_2 2q^2)M(GF(q))M(n)\log_2 m/\log_2\log_2 m$$

Если применить вместо наилучшей оценки сложности умножения матриц алгоритм «четырех русских» с оценкой $O(\log_2 q)n^3/\log_q n$, имеем оценку сложности

$$(3 + o(1))(\log_2 2q^2)M(GF(q))M(n)\log_2 m/\log_2\log_2 m)+$$

$$+O(\log_2 q)n^2 + 6M(n)n^{1/2}M(GF(q))\log_q \log_q m.$$

Так как $m < q^n$, то отсюда следует оценка

$$(3 + o(1))(\log_2 q \log_2 2q^2)M(GF(q))M(n)n/\log_2(n\log_2 q).$$

Если число ненулевых цифр в q- ичной записи числа $m < q^n$ равно $\nu_q(m)$, то оценку сложности можно получить в виде

$$O(M(GF(q)\log_2 q)n^{1.85} + (3\nu_q(m) + O(q))M(n)M(GF(q)).$$

Для этого выберем $k = \lceil(\log_q m)^{1-\alpha}\rceil \leq \lceil n^{1-\alpha}\rceil$, разложим, как и выше, m в системе счисления по основанию q^k, вычислим все степени $g^{q^i a} \bmod f, a = 0, \ldots q, i = 0, \ldots k-1$ со сложностью

$$O(q + kl(q))M(n)M(GF(q)),$$

потом вычислим все степени $g_i, i = 0, \ldots s-1, s = \lceil(\log_q m)/k\rceil = \lceil(\log_q m)^\alpha\rceil \leq \lceil n^\alpha\rceil$ просто путем перемножения «содержащихся в них» степеней вида $g^{q^i a} \bmod f$ со сложностью $3\nu_q(m)M(n)M(GF(q))$, и закончим работу алгоритма также, как и выше. Оценка сложности тогда принимает вид

$$3\nu_q(m)M(n)M(GF(q))+$$

$$+O(M(n))M(GF(q))\left(q + k\log_2 q + s\left(n^{1/2} + n^{(\omega+1)/2}\right)\right) \leq$$

$$3\nu_q(m)M(n)M(GF(q))+$$

$$+O(M(n))M(GF(q))\left(q + n^{1-\alpha}\log_2 q + n^{1/2+\alpha} + n^{(\omega+1)/2+\alpha}\right).$$

Выбирая $\omega = 2.376$ (см. [?]vonGGer) и оптимизируя выбор параметра α, имеем при $M(n) = O(n^{1+\epsilon_n})$ оценку

$$O(M(GF(q))\log_2 q)n^{1.85} + (3\nu_q(m) + O(q))M(n)M(GF(q)).$$

3.7.20 Быстрое инвертирование в стандартном базисе через модулярную композицию

Напомним, что для инвертирования в поле $GF(q^n)$ можно вычислить $y = x^{(q^n-1)/(q-1)} \in GF(q)$ и применить формулу

$$x^{-1} = (x^{(q^{n-1}-1)/(q-1)})^q y^{-1},$$

Вычисление y^{-1} по определению выполняется со сложностью $I(q)$, а умножение на него в поле $GF(q^n)$ выполняется со сложностью $nM(GF(q))$. Для вычисления $x^{(q^{n-1}-1)/(q-1)}$ используется $l(n-1)$ умножений в поле $GF(q^n)$, а для

вычисления y нужна еще одна операция умножения, более простая, чем обычное умножение. Но кроме умножений, используется еще не более $\lambda_2(n-1)+1$ операций вычисления преобразований Фробениуса, т.е. операций возведения в степени $q^m, m \le n$. Используя указанную выше оценку сложности такой операции, получаем оценку сложности всех использованных этих операций в виде

$$O(M(n)\log_2 q + (n^{(\omega+1)/2} + n^{1/2}M(n)))M(GF(q))\log_2^2 n),$$

где $M(n)$ — число операций в поле $GF(q)$, используемых для умножения многочленов степени меньше n над этим полем. Покажем, что ее можно улучшить до

$$O(M(n)\log_2 q) + O(n^{(\omega+1)/2} + n^{1/2}M(n))M(GF(q)l(n-1),$$

и получить оценку сложности инвертирования в виде

$$I(q^n) \le (l(n-1)+1)M(GF(q^n)) + nM(GF(q)) + I(q) +$$

$$+O(M(n)\log_2 q) + O(n^{(\omega+1)/2} + n^{1/2}M(n))l(n-1).$$

Главным членом в ней при малых q и $M(n) = o(n^{\omega/2})$ будет

$$O((n^{(\omega+1)/2})l(n-1) = O(n^{1.85}).$$

Выше отмечалось, что с учетом новейших результатов о матричном умножении, окончательная асимптотическая оценка принимает вид $O(n^1.668)$. Далее для краткости сложность модулярной композиции обозначим

$$S(n) = O(n^{(\omega+1)/2}) + 6n^{1/2}(M(n) + O(n)).$$

Для доказательства указанной выше оценки надо взять минимальную (не обязательно линейную) цепочку

$$1 = a_0, \ldots, a_r = n-1, r = l(n-1),$$

и построить цепочку для $(q^{n-1}-1)/(q-1)$, состоящую из чисел

$$(q^{a_i}-1)/(q-1),$$

в промежутки между которыми вставлены операции умножения на числа q^{a_j}, в используемой аддитивной записи соответствующие преобразованиям Фробениуса, так как

$$(q^{a_i}-1)q^{a_j}/(q-1) + (q^{a_j}-1)/(q-1) = (q^{a_i+a_j}-1)/(q-1).$$

Значит, для вычисления $x^{(q^{n-1}-1)/(q-1)}$ нужны операции Фробениуса для показателей $a_0, \ldots, a_r$ (может быть, не все). Выше было показано, что операция Фробениуса $g^{q^m} \bmod f$ может быть выполнена как модулярная композиция

$g(h) \bmod f$, где $h = x^{q^m} \bmod f$. Всего нужно не более $l(n-1)$ модулярных композиций, если уже вычислены многочлены $h_i = x^{q^{a_i}} \bmod f, i = 1, \ldots, r$. Но если $a_i = a_j + a_k$, то

$$h_i = x^{q^{a_i}} \bmod f = x^{q^{a_j + a_k}} \bmod f = (x^{q^{a_j}})^{q^{a_k}} \bmod f = (x^{q^{a_j}} \bmod f)^{q^{a_k}} \bmod f = h_j^{q^{a_k}} \bmod f = h_j(h_k) \bmod f,$$

поэтому для вычисления многочленов $h_i = x^{q^{a_i}} \bmod f, i = 1, \ldots, r$ достаточно вычислить $x^q \bmod f$ и выполнить $r = l(n-1)$ модулярных композиций. Поэтому сложность выполнения всех преобразований Фробениуса оценивается как

$$2l(n-1)S(n) + O(M(n)\log_2 q).$$

В случае $q < n$ последнее слагаемое отсутствует.

3.7.21 Некоторые уточнения в случае $q = 2$

Выбирая вместо минимальной цепочки обычную линейную бинарную цепочку, состоящую из $\lambda(n)$ удвоений и $\nu(n) - 1$ прибавлений единицы, можно чуть усилить полученную выше оценку до оценки

$$I(2^n) \le (\lambda(n-1)+\nu(n-1))M(GF(2^n))+2\nu(n-1)K(GF(2^n))+n+2\lambda(n-1)S(n),$$

где $K(GF(2^n))$ — сложность возведения в квадрат в поле $GF(2^n)$. Известно, что $K(GF(2^n)) \le 2M(n)+O(n), M(GF(2^n)) \le 3M(n)+O(n)$ в общем случае, а если базис задан неприводимым малочленом, то $K(GF(2^n)) = O(n)$.

Действительно, если

$$1 = a_0, \ldots, a_r = n-1, r = \lambda(n-1) + \nu(n-1) - 1$$

такая цепочка, то действуя также, как и выше, можно выполнить инвертирование с помощью $\lambda(n-1) + \nu(n-1) - 1$ умножений и $\lambda(n-1) + \nu(n-1) - 1$ операций Фробениуса, из которых $\nu(n-1) - 1$ являются просто возведениями в квадрат. Для выполнения остальных $\lambda(n-1)$ операций Фробениуса используем $\lambda(n-1)$ операций модулярного умножения, одновременно вычисляя необходимые для этого многочлены $x^{2^{a_i}} \bmod f$, где a_i — такие элементы цепочки, которые в ней на следующем шаге удваиваются. Эти вычисления тоже делаются с помощью $\lambda(n-1) - 1$ операций модулярного умножения, из которых $\nu(n)$ операций являются возведениями в квадрат. Заметим, что при этом уменьшается используемая память, так как вычисление последовательности $x^{2^{a_i}} \bmod f$ требует запоминания только одного многочлена во время всего вычисления, а другой многочлен надо помнить во время вычисления промежуточных экспонент $g^{2^{a_i}-1} \bmod f$. Так как возведения в квадрат выполняются отдельно, то все используемые операции Фробениуса являются возведениями в степени 2^{a_i}, где последовательность a_i растет как геометрическая прогрессия: $a_{i+1} \le 2a_i$. Воспользуясь этим, укажем еще возможность уменьшения сложности вычислений в случае использования базиса, заданного неприводимым малочленом.

Пусть сложность операции модулярной композиции оценивается как $S(n) = O(n^\beta), \beta < 2$. Пусть k таково, что $a_k < n^{\beta-1} \leq a_{k+1}$. Тогда возведения в степени $2^{a_i}, i \leq k$ можно выполнить путем повторного возведения в квадрат со сложностью

$$\sum_{i=1}^{k} O(a_i n) = O(a_k n) = O(n^\beta)),$$

а остальные $(2-beta)\log_2 n$ возведений в степени $2^{a_i}, i \geq k$ выполняем также, как описано выше. Тогда получаем следующую оценку сложности инвертирования

$$I(2^n) \leq (\lambda(n-1)+\nu(n-1))M(GF(2^n))+n+2\nu(n-1)K(GF(2^n))+(2(2-beta)\lambda(n-1)+O(1))S(n).$$

В случае схемной реализации множители 2 в последних слагаемых можно отбросить, так как вычисления заранее заданных многочленов $x^{2^{a_i}} \bmod f$ выполнять не нужно.

Заметим, что в случае программной имплементации и использования для матричного умножения алгоритма «четырех русских», получаем для сложности инвертирования всего лишь квадратичную оценку и только при использовании для достаточно быстрого алгоритма умножения многочленов с оценкой $M(n) = O(n^{1.4})$, что не совсем реально при $n < 1000$. Аналогичная оценка для тех же методов получается и в случае схемной реализации, причем глубина полученной схемы будет равна $O(\log_2 n)^2$. Оба этих результата легко получить и без использования модулярной композиции, применяя вместо нее схему сложности $O(n^2/\log_2 n)$ и глубины $O(\log_2 n)$ для возведения в любую степень 2^k. Однако в случае программной имплементации это требует большого расхода памяти, так матрицы каждого используемого линейного преобразования Фробениуса надо запоминать, а уже для одного из них это требует $O(n^2)$ памяти. Для выполнения модулярной композиции требуется только $O(n^{3/2}$ памяти. Некоторого уменьшения сложности можно добиться, также как и выше, используя схемы возведения в квадрат по модулю малочлена с членами малых степеней сложности $O(n)$ и константной глубины.

Упражнение 3.7.54 Как построить такие схемы?

Сложность схемы инвертирования можно уменьшить с квадратичной до

$$\frac{O(n^2)\log\log n}{\log n},$$

используя указанный выше трюк. А именно, возведения в степени $2^{a_i}, a_i \leq n/\log_2 n$ выполняем путем повторного возведения в квадрат со сложностью

$$\frac{O(n^2)}{\log n},$$

а остальные $\log_2\log_2 n$ возведения в степени 2^{a_i} вычисляются со сложностью

$$\frac{O(n^2)(\log\log n}{\log n}.$$

Эта оценка справедлива и для схемной сложности, но глубина схемы получается $O(n/log_2 n)$ даже если использовать в качестве f малочлен.

Так как умножение методом Штрассена имеет глубину $O(\log_2 n)$, то можно построить схему для модулярного умножения на фиксированный многочлен сложности $S(n) = O(n^{\log_2 \sqrt{14}}/(\log_2 n)^{log_2 7/4})$ и глубины $O(\log_2 n)$, поэтому можно построить схему для инвертирования глубины $O(\log_2^2 n)$ и сложности

$$O(S(n)\lambda(n)) = O(n^{\log_2 \sqrt{14}}(\log_2 n)^{log_2 8/7}).$$

Упражнение 3.7.55 Докажите это.

Для уменьшения глубины схемы инвертирования вместо модулярного умножения лучше использовать упоминавшуюся выше схему глубины $\log_2 n + O(1)$ и сложности $O(n^2/\log_2 n)$, а вместо стандартной бинарной аддитивной цепочки использовать аддитивную цепочку для $n-1$ минимальной глубины $\lceil \log_2(n-1) \rceil$.

Упражнение 3.7.56 Докажите по индукции, что существует цепочка для $n \neq 2^k$ глубины $\lambda(n)+1$ и сложности $\lambda(n)(n)+\nu(n)-1$, в которой на каждом шаге выполняется прибавление некоторой степени двойки.

Тогда получается схема глубины $(\lambda(n-1)+1)(D(M(GF(2^n)))+\lambda(n-1)+1)$ и сложности

$$I(2^n) \leq (\lambda(n-1)+\nu(n-1)-1)M(GF(2^n))+n+(\lambda(n-1)+\nu(n-1)-1)S(n).$$

Кроме умножений, в ней будут использоваться только операции возведения в степени $2^k, k = 1, \ldots \lambda(n-1)$. Если используется в качестве f малочлен, для которого модулярное возведение в квадрат имеет сложность $O(n)$ и глубину $O(1)$, то сложность и глубину схемы инвертирования можно немного уменьшить, заменив первые $O(\log\log n)$ операций возведения в степени повторным возведением в квадрат.

Глава 4

Схемы операций в нормальных базисах

4.1 Арифметические операции в полях $GF(2^n)$

4.1.1 Пример построения схемы для умножения и инвертирования в оптимальных нормальных базисах второго типа

Пусть $n = 23$. Можно проверить, что это случай базиса третьего типа, степени двойки порождают все 23 квадратичных вычета по модулю $p = 2n + 1 = 47$, и $2^{23} = 1 \bmod 47$. Укажем без доказательства как выглядит матрица T, определяющая умножение в этом базисе (подобно рассмотренным случаям $n = 10, 28$. Сначала вычислим последовательность $\pi(k) = 2^k \bmod p, k = 0, \ldots, n-1$. Она имеет вид

$$1, 2, 4, 8, 16, 32, 17, 34, 21, 42, 37, 27, 7, 14, 28, 9, 18, 36, 25, 3, 6, 12, 24.$$

Далее определим последовательности $1 \le \sigma(k) \le n, k = 1, \ldots, n$ и $1 \le \mu(k) \le n, k = 1, \ldots, n$ так, чтобы

$$1 + 2^k = \pm\pi(\sigma(k+1)) \bmod p, k = 0, \ldots, n-1,$$

$$1 - 2^k = \pm\pi(\mu(k)) \bmod p, k = 1, \ldots, n.$$

Тогда элементы матрицы T определяются равенствами $t_{i,j} = \delta_{\sigma(i+1),j} + \delta_{\mu(i),j}$, где $\delta_{k,k} = 1, \delta_{k,l} = 0, k \ne l$. Можно проверить, что всегда $t_{i,j} = 0, 1$ и общее число единиц в матрице T равно $2n - 1$. Алгоритм Месси-Омура умножения в этом базисе естественно такой же, как был описан выше. Его сложность равна $n(3n-2) = 1541$, а глубина равна $2 + \lceil \log_2 n \rceil = 7$.

Опишем как можно построить схему перехода от нормального базиса

$$\{\alpha, \alpha^2, \alpha^4, \alpha^8, \alpha^{16}, \ldots, \alpha^{2^{n-1}}\}$$

к стандартному базису

$$\{\alpha, \alpha^2, \dots \alpha^n\}.$$

Так как $\alpha = \zeta + \zeta^{-1}$, где $\zeta \neq 1$ любой элемент поля $GF(2^{2n})$, такой, что $\zeta^p = 1$ (его конкретный вид не будет использоваться), то

$$\alpha^{2^k} = \zeta^{2^k \bmod p} + \zeta^{-2^k \bmod p} = \zeta^l + \zeta^{-l} = \alpha_l, 1 \leq l \leq n$$

поэтому базис

$$\{\alpha, \alpha^2, \alpha^4, \alpha^8, \alpha^{16}, \dots, \alpha^{2^{n-1}}\}$$

является перестановкой базиса

$$\{\alpha_1, \alpha_2, \dots \alpha_n\},$$

а именно, совпадает с базисом

$$\{\alpha_1, \alpha_2, \alpha_4, \alpha_8, \alpha_{16}, \alpha_{15}, \alpha_{17}, \alpha_{13}, \alpha_{21}, \alpha_5, \alpha_{10}, \alpha_{20}, \alpha_7, \alpha_{14}, \alpha_{19}, \alpha_9, \alpha_{18}, \alpha_{11},$$

$$\alpha_{22}, \alpha_3, \alpha_6, \alpha_{12}, \alpha_{23}\}.$$

Поэтому для перехода от нормального базиса к базису

$$\{\alpha_1, \alpha_2, \dots \alpha_n\},$$

достаточно выполнить соответствующую перестановку координат, а для обратного перехода — обратную перестановку.

Опишем теперь, как перейти от базиса

$$\{\alpha_1, \alpha_2, \dots \alpha_n\},$$

к базису

$$\{\alpha^1, \alpha^2, \dots \alpha^n\}.$$

Для этого надо установить связь между $n-$мерными булевскими векторами $\mathbf{x}, \mathbf{y}$, такими, что

$$\sum_{i=1}^{n} x_i \alpha^i = \sum_{i=1}^{n} y_i \alpha_i.$$

Эта связь задается линейным преобразованием $y = F_n(x)$. Его можно вычислить рекуррентно. Покажем, как это сделать при $n = 23$. Воспользуемся тождествами

$$\alpha_i \alpha_j = \alpha_{i+j} + \alpha_{i-j}, i > j, \alpha_i^2 = \alpha_{2i}, \alpha_{2^i} = \alpha^{2^i} = \alpha^{2^i},$$

$$\alpha_i \alpha^{2^k} = \alpha_{2^k+i} + \alpha_{2^k-i}, i = 1, \dots, 2^k - 1.$$

Применяя их, получаем, что $\alpha_1 \alpha^2 = \alpha_3 + \alpha_1$, откуда

$$\sum_{i=1}^{4} y_i \alpha_i = y_1 \alpha_1 + y_2 \alpha_2 + \alpha^2 (y_3 \alpha_1) + y_3 \alpha_1 + y_4 \alpha_4 =$$

$$= (y_1 + y_3)\alpha_1 + y_2\alpha_2 + \alpha^2(y_3\alpha_1 + y_4\alpha_2) = \sum_{i=1}^{n} x_i\alpha^i,$$

значит

$$x_1 = y_1 + y_3, x_2 = y_2, x_3 = y_3, x_4 = y_4,$$

поэтому сложность преобразования F_4 (и совпадающего с ним обратного) равна 1, и глубина тоже 1. Такие же утверждения справедливы и для F_3.

Преобразование F_8 выражается через F_4, а преобразование F_7 — через F_4, F_3 и то же верно для обратных преобразований. Для этого воспользуемся тождествами

$$\alpha_1\alpha^4 = \alpha_5 + \alpha_3, \alpha_2\alpha^4 = \alpha_6 + \alpha_2, \alpha_3\alpha^4 = \alpha_7 + \alpha_1\alpha_4^2 = \alpha_8$$

и вытекающим из них тождеством

$$\sum_{i=1}^{8} y_i\alpha_i = \sum_{i=1}^{4} y_i\alpha_i + \sum_{i=1}^{3} y_{8-i}\alpha_i\alpha^4(\sum_{i=1}^{4} y_{4+i}\alpha_i) =$$

$$\sum_{i=1}^{3}(y_i + y_{8-i})\alpha_i + y_4\alpha_4 + \alpha^4(\sum_{i=1}^{4} y_{4+i}\alpha_i) = \sum_{i=1}^{8} x_i\alpha^i.$$

Из этого тождества следует, что

$$(x_1, \ldots, x_8) = F_8(y_1, \ldots, y_8) = (F_4(y_1 + y_7, y_2 + y_6, y_3 + y_5, y_4), F_4(y_5, \ldots, y_8)),$$

$$(y_1, \ldots, y_8) = F_8^{-1}(x_1, \ldots, x_8) = (S(F_4^{-1}(x_1, \ldots, x_4), y_5, \ldots, y_8), F_4^{-1}(x_5, \ldots, x_8)),$$

где схема S с 8-ю входами и выходами реализует линейное преобразование

$$S(u_1, \ldots, u_8) = (u_1 + u_7, u_2 + u_6, u_3 + u_5, u_4, \ldots, u_8).$$

Поэтому сложность преобразования F_8 равна 5, а глубина — 2 и то же верно для обратного преобразования F_8^{-1}. Аналогично получаем такие же оценки сложности и глубины для F_7. В общем случае рекуррентные оценки имеют вид

$$L(n) \le L(2^k) + L(n - 2^k) + n - 2^k, 2^k < n < 2^{k+1}, L(2^{k+1}) \le 2^k - 1 + 2L(2^k),$$

$$D(n) \le D(2^k) + 1, D(2^k) \le D(2^{k-1}) + 1,$$

откуда

$$L(2^k) = 2^{k-1}(k-2) + 1, D(2^k) = k - 1,$$

$$L(16) = 17, D(16) = 3, L(23) = 17 + 5 + 23 - 16 = 29, D(23) = 4.$$

Таким образом, умножение в оптимальном нормальном базисе 2-го или 3-го типа

$$\{\alpha, \alpha^2, \alpha^4, \alpha^8, \alpha^{16}, \ldots, \alpha^{2^{n-1}}\}$$

сводится к умножению в стандартном базисе

$$\{\alpha, \alpha^2, \ldots \alpha^n\}$$

с дополнительной сложностью не больше $3n/2\log_2 n+3n$ (при $n=23$ не больше 87) и дополнительной глубиной $\lceil\log_2 n\rceil-1$ (при $n=23$ с дополнительной глубиной 4). Умножение в стандартном базисе сводится к обычному умножению двоичных многочленов степени меньшей n, в результате которого получается многочлен f степени не выше $2n-2$, умножению его на x (оно делается бесплатно) и вычислению остатка от его деления на минимальный многочлен g_n указанного базиса. Для вычисления вектора коэффициентов этого остатка можно применить стандартный алгоритм быстрого деления: сначала найти частное q от этого деления (оно имеет степень не больше $n-1$), потом найти младшие n коэффициентов произведения q на g_n, и их сложить попарно с младшими коэффициентами многочлена f. Для вычисления q сначала находим частное r_n от деления x^{2n+1} на g_n (оно не зависит от f), а потом вычислить произведение f на r_n, причем нужны только коэффициенты при члене x^{2n+1} и выше. Оба умножения имеют один переменный и один постоянный множитель с небольшим числом одночленов, и могут быть реализованы схемами из одних элементов XOR, моделирующими стандартный алгоритм умножения столбиком. Основная проблема — вычислить эти множители g_n и r_n, но это можно делать программным образом.

Теоретически ответ можно иногда получить и в явном виде, например в случае $n=3\cdot 2^k-1$. При $n=5$ получается базис второго,а при $n=11,23,191$ получаются базисы 3-го типа. Достаточным условием существования такого базиса является простота чисел n и $2n+1$. Ранее в одном примере было вычислено, что

$$g_{23}=1+x^4+x^6+x^7+x^8+x^{16}+x^{20}+x^{22}+x^{23},$$

и вообще при $n=3\cdot 2^k-1$ получится $2k+3-$член

$$g_n=1+x^{2^{k-1}}+x^{2^{k-1}+2^{k-2}}+x^{2^{k-1}+2^{k-2}+2^{k-3}}+$$

$$+\ldots+x^{2^k-1}+x^{2^k}+x^{2^{k+1}+2^{k-1}}+x^{2^{k+1}+2^{k-1}+2^{k-2}}+x^{2^{k+1}+2^{k-1}+2^{k-2}+2^{k-3}}+\ldots+x^{2^{k+1}+2^k-1}=$$

$$(1+x^{2^{k+1}})h_k+x^{2^k},h_k=1+x^{2^{k-1}}+x^{2^{k-1}+2^{k-2}}+x^{2^{k-1}+2^{k-2}+2^{k-3}}+\ldots+x^{2^k-1}$$

Известно при любом k непосредственно проверяемое тождество над полем $GF(2)$

$$h_k(h_k+x^{2^k})=1+x^{2^{k+1}-1}.$$

Из него следует тождество

$$g_nr_n=((1+x^{2^{k+1}})h_k+x^{2^k})((1+x^{2^{k+1}})h_k+x^{2^{k+1}+2^k})=$$

$$=(x^{2^{k+1}}h_k+(h_k+x^{2^k}))(x^{2^{k+1}}(h_k+x^{2^k})+h_k)=$$

$$=(1+x^{2^{k+2}})h_k(h_k+x^{2^k})+(h_k^2+(h_k+x^{2^k})^2)x^{2^{k+1}}=$$

$$=(1+x^{2^{k+2}})(1+x^{2^{k+1}-1})+(x^{2^k})^2)x^{2^{k+1}}=(1+x^{2^{k+2}})(1+x^{2^{k+1}-1})+x^{2^{k+2}}=$$

$$=x^{2^{k+2}+2^{k+1}-1})+x^{2^{k+1}-1})+1=x^{2n+1}+x^{2^{k+1}-1})+1,$$

откуда видно, что частное от деления x^{2n+1} на g_n равно

$$r_n = (1 + x^{2^{k+1}})h_k + x^{2^{k+1}+2^k} = (1 + x^{2^{k+1}})h_k + x^{n+1}.$$

В частности,

$$r_{23} = 1 + x^4 + x^6 + x^7 + x^{16} + x^{20} + x^{22} + x^{23} + x^{24}.$$

Умножение f на r_n (с вычислением только старших коэффициентов начиная с члена x^{2n+1}) в случае $n = 23$ можно реализовать с помощью восьми векторных сложений по модулю два схемой глубины 3 и сложности 2+4+5+14+18+20+21. После этого полученное частное степени не выше $n-1$ умножаем подобным же образом на g_n схемой глубины 3 и сложности $(23-4)+(23-6)+(23-7)+(23-8)+(23-16)+(23-20)+(23-22)$. После этого производится сложение полученного вектора с вектором младших коэффициентов f. Общая глубина будет равна 7, а общая сложность $7(23-2)+23-8+23 = 185$.

В случае $n = 3 \cdot 2^k - 1$ глубина будет

$$2\lceil \log_2(2k+2) \rceil + 1 = 2\log_2 \log_2 n + O(1),$$

а сложность равна

$$(n-2)(2k+3) - 2^k + 4 < 2n \log_2 n + n.$$

Если обозначит $m(n)$ сложность умножения двоичных многочленов степени меньшей n, а через $d(n)$ — соответствующую глубину, то сложность и глубина умножения в оптимальном нормальном базисе 3-го типа при $n = 3 \cdot 2^k - 1$ оцениваются сверху как

$$M(n) \le m(n) + 3L(n) + 2n \log_2 n + n \le m(n) + 7n/2 \log_2 n + 4n,$$

$$D(n) \le d(n) + 2\log_2 n + 2\log_2 \log_2 n + O(1).$$

При $n = 23$

$$M(n) \le m(n) + 87 + 185,$$

$$D(n) \le d(n) + 4 + 7.$$

Если для умножения взять стандартную схему, то $m(n) \le n^2 + (n-1)^2, d(n) \le \lceil \log_2 n \rceil$, откуда

$$M(n) \le n^2 + (n-1)^2 + 7n/2 \log_2 n + 4n,$$

$$D(n) \le 3\log_2 n + 2\log_2 \log_2 n + O(1).$$

При $n = 23$

$$M(n) \le 1285,$$

$$D(n) \le 17.$$

Для сравнения сложность мультиплера Месси-Омура при $n = 23$ равна $n(3n - 2) = 1541$, а глубина равна $2 + \lceil \log_2 n \rceil = 7$.

Но если для умножения взять схему Карацубы, то уже при $n = 23$ получаются оценки

$$m(23) \leq 755, d(23) \leq 10.$$

Действительно,

$$m(23) = 3m(11)+4\cdot 23, m(11) = 3m(5)+4\cdot 11, m(5) = 3m(2)+20 = 3\cdot 13+20 = 59,$$

а мультипликативная сложность равна $3^5 = 243$, откуда

$$M(23) \leq 1027, D(n) \leq 21,$$

а мультипликативная сложность по прежнему равна 243.

При использовании этой схемы умножения в качестве подсхемы схемы инвертирования на самом деле достаточно одной схемы F_n и одной схемы F_n^{-1}, поэтому оценка сложности будет на 29 меньше, т.е.

$$M(23) \leq 998.$$

По теореме Ферма обратный элемент можно вычислить по формуле $\xi^{-1} = \xi^{2^n-2}$. Для возведения в степень надо построить аддитивную цепочку для числа $2^n - 2$, содержащую минимальное число сложений. Для этого сначала надо построить минимальную линейную аддитивную цепочку для числа $n - 1 = 22$. Например, такой будет цепочка

$$1, 2, 4, 8, 16, 20, 22.$$

После этого выписываются в ряд числа

$$2^1 - 1, 2^2 - 1, 2^4 - 1, 2^8 - 1, 2^{16} - 1, 2^{20} - 1, 2^{22} - 1,$$

и между ними вставляются последовательности удвоений. Так как удвоения реализуются путем соответствующей коммутации входов, то сложность инвертирования в поле $GF(2^{23})$ оценивается как

$$I(23) \leq 6M(23),$$

где $M(23)$ сложность умножения в нормальном базисе. Аналогично глубина оценивается как

$$DI(23) \leq 6DM(23).$$

Если использовать указанную выше схему для умножения в нормальном базисе, то сложность инвертирования будет $I(23) = 6\cdot 998 = 5988$, а глубина $6\cdot 21 = 126$.

Можно взять минимальную по глубине линейную аддитивную цепочку для числа $n - 1 = 22$. Например, такой будет цепочка

$$1, 2, 4, 6, 8, 16, 20, 22.$$

Ее глубина будет 5, но сложность 7 После этого выписываются в ряд числа

$$2^1-1, 2^2-1, 2^4-1, 2^6-1, 2^8-1, 2^{16}-1, 2^{20}-1, 2^{22}-1,$$

и между ними вставляются последовательности удвоений. Так как удвоения реализуются путем соответствующей коммутации входов, то сложность инвертирования в поле $GF(2^{23})$ оценивается как

$$I(23) \le 7M(23),$$

Аналогично глубина оценивается как

$$DI(23) \le 5DM(23).$$

Если использовать указанную выше схему для умножения в нормальном базисе, то сложность инвертирования будет $7 \cdot 998 = 6972$, а глубина $5 \cdot 21 = 105$. Если использовать схему $M(23) = 1288, D(23) = 7$ получим

$$I(23) \le 9016, DI(23) \le 35.$$

4.1.2 О произведениях базисов

Известна следующая конструкция (кронекерова) произведения базисов (см., например [45, 109]). Пусть m и n – взаимно простые числа, а $B_1 = \{\alpha_1, \ldots, \alpha_n\}$, $B_2 = \{\beta_1, \ldots, \beta_m\}$ – произвольные базисы в полях $GF(q^n)$ и $GF(q^m)$ соответственно. Тогда пересечение этих полей совпадает с полем $GF(q)$, и оба поля содержатся в поле $GF(q^{nm})$, базисом в котором (над полем $GF(q)$) будет произведение базисов

$$B_1 \otimes B_2 = \{\alpha_1\beta_1, \ldots, \alpha_n\beta_m\}.$$

Произведение стандартных базисов не будет стандартным базисом и нетривиальной оценки меры близости этого базиса к стандартному авторам неизвестно. Однако для этого произведения базисов сложность умножения в поле $GF(q^{nm})$ легко оценивается как

$$M_{B_1\otimes B_2}(GF(q^{nm})) \le M_{B1}(GF(q^n))M^s_{q^n,g}(m) \le M^s_{q,f}(n)M^s_{q^n,g}(m), \qquad (4.1)$$

где g – неприводимый полином степени m над полем $GF(q)$, порождающий базис B_2, а f – неприводимый полином степени n над полем $GF(q)$, порождающий базис B_1.

Однако, как известно [45, 109], произведение $B^\alpha \otimes B^\beta$ нормальных базисов $B^\alpha \subset GF(q^n)$ и $B^\beta \subset GF(q^m)$ с точностью до перестановки элементов совпадает с нормальным базисом $B^\gamma \subset GF(q^{nm})$, $\gamma = \alpha\beta$.

Рассмотрим также произведение стандартных базисов $B_\alpha \otimes B_\beta$. Легко видеть, что сложность перехода от базиса $B^\alpha \otimes B^\beta$ к базису $B_\alpha \otimes B_\beta$ оценивается сверху как

$$nC^\beta_{q,\beta}(m) + mC^\alpha_{q,\alpha}(n). \tag{4.2}$$

Действительно, если произвольный элемент поля $GF(q^{nm})$ записывается в обоих базисах как

$$\sum_{i=0}^{n-1}\sum_{j=0}^{m-1} x_{i,j}\alpha^{q^i}\beta^{q^j} = \sum_{i=0}^{n-1}\sum_{j=0}^{m-1} x'_{i,j}\alpha^i\beta^j,$$

то для преобразования матрицы $(x_{i,j})$ над полем $GF(q)$ в матрицу $(x'_{i,j})$ над тем же полем достаточно n раз выполнить (в соответствующую сторону) преобразование $C^\beta_{q,\beta}(m)$, над строками матрицы $(x_{i,j})$, чтобы получить равенство

$$\sum_{i=0}^{n-1}\alpha^{q^i}\sum_{j=0}^{m-1} x_{i,j}\beta^{q^j} = \sum_{i=0}^{n-1}\alpha^{q^i}\sum_{j=0}^{m-1} x''_{i,j}\beta^j,$$

а потом m раз выполнить преобразование $C^\alpha_{q,\alpha}(n)$ над столбцами полученной матрицы $(x''_{i,j})$ над тем же полем $GF(q)$, чтобы получить равенство

$$\sum_{i=0}^{n-1}\alpha^{q^i}\sum_{j=0}^{m-1} x''_{i,j}\beta^j = \sum_{j=0}^{m-1}\beta^j\sum_{i=0}^{n-1} x''_{i,j}\alpha^{q^i} = \sum_{i=0}^{n-1}\sum_{j=0}^{m-1} x'_{i,j}\alpha^i\beta^j.$$

Из (4.1) и (4.2) следует, что при подходящем выборе элементов α и β (или, что то же самое, многочленов f и g) и $\gamma = \alpha\beta$ сложность умножения в нормальном базисе в поле $GF(q^{nm})$ оценивается как

$$M^O(GF(q^{nm})) \le M_{B^\gamma}(GF(q^{nm})) \le M^s_{q,f}(n)M^s_{q^n,g}(m) + nC^\beta_{q,\beta}(m) + mC^\alpha_{q,\alpha}(n)$$

$$\le M^s_{q,f}(n)M^s_{q^n,g}(m) + nC_q(m) + mC_q(n).$$

Учитывая оценку Шенхаге [147], отсюда имеем

$$M^O(GF(q^{nm})) =$$

$$O(nm\log n\log m\log\log n\log\log m) + O(nm)(m+n)/log_q(n+m).$$

Для получения оценки битвой сложности нужно приведенные оценки, естественно, умножить на $M(GF(q))$.

В случае, например когда для n и m существуют оптимальные нормальные базисы, последнее слагаемое можно заменить на меньшее, например в последнем случае его можно заменить на $O(nm)log_q(n+m)$. Последнее утверждение очевидным образом распространяется и на случай кронекерова произведения нескольких базисов. В указанном случае последнее слагаемое очевидно поглощается первым и может быть опущено.

Кронекерово произведение базисов можно определить и в случае, когда m и n не обязательно взаимно простые числа, но тогда базис $B_2 = \{\beta_1, \ldots, \beta_m\}$ выбирается в поле $GF(q^{nm})$ и над полем $GF(q^m)$.

Для произведения стандартных базисов сложность умножения в поле $GF(q^{nm})$ опять оценивается как

$$M_{B_1 \otimes B_2}(GF(q^{nm})) \le M^s_{q,f}(n) M^s_{q^n,g}(m), \tag{4.3}$$

но здесь g – неприводимый полином степени m над полем $GF(q^n)$, порождающий базис B_2.

Рассмотрим произведение $B^\alpha \otimes B^\beta$ нормальных базисов $B^\alpha \subset GF(q^n)$ и $B^\beta \subset GF(q^{mn})$, последний из которых – над полем $GF(q^n)$. Этот базис не является нормальным, но очевидно, что возведение в степень q в этом базисе тоже сводится лишь к перестановке координат, но уже не циклической. Поэтому схемная сложность возведения в степень q в этом базисе, как и в нормальном равна нулю и такой базис тоже можно использовать для уменьшения сложности экспоненциирования и инвертирования в поле $GF(q^{nm})$, если получить для него алгоритм умножения невысокой сложности.

Для этого, как и выше, рассмотрим произведение соответствующих стандартных базисов $B_\alpha \otimes B_\beta$. В рассматриваемом случае сложность перехода от базиса $B^\alpha \otimes B^\beta$ к базису $B_\alpha \otimes B_\beta$ оценивается сверху как

$$C^\beta_{q^n,\beta}(m) + mC^\alpha_{q,\alpha}(n). \tag{4.4}$$

Действительно, если произвольный элемент поля $GF(q^{nm})$ записывается в обоих базисах как

$$\sum_{i=0}^{n-1}\sum_{j=0}^{m-1} x_{i,j}\alpha^{q^i}\beta^{q^j} = \sum_{i=0}^{n-1}\sum_{j=0}^{m-1} x'_{i,j}\alpha^i\beta^j,$$

то для преобразования матрицы $(x_{i,j})$ над полем $GF(q)$ в матрицу $(x'_{i,j})$ над тем же полем достаточно m раз выполнить преобразование $C^\beta_{q,\beta}(n)$, над строками матрицы $(x_{i,j})$, чтобы получить равенство

$$\sum_{i=0}^{n-1}\alpha^{q^i}\sum_{j=0}^{m-1} x_{i,j}\beta^{q^j} = \sum_{j=0}^{m-1}\beta^{q^j}\sum_{i=0}^{n-1} x''_{i,j}\alpha^i,$$

а потом один раз выполнить преобразование $C^\alpha_{q^n,\alpha}(m)$ над столбцами полученной матрицы $(x''_{i,j})$ над тем же полем $GF(q)$, чтобы получить равенство

$$\sum_{j=0}^{m-1}\beta^{q^j}\sum_{i=0}^{n-1} x''_{i,j}\alpha^i = \sum_{i=0}^{n-1}\sum_{j=0}^{m-1} x'_{i,j}\alpha^i\beta^j.$$

Из (4.3) и (4.4) следует, что при подходящем выборе элементов α и β (или, что то же самое, многочленов f и g) сложность умножения в поле $GF(q^{nm})$ в

некотором базисе с «бесплатным» возведением в q-ю степень оценивается сверху при малых q как

$$M^s_{q,f}(n)M^s_{q^n,g}(m) + O(m^2n) + O(mn^2)/\log_q n.$$

Учитывая оценку из [147] отсюда имеем

$$M^O(GF(q^{nm})) =$$

$$O(nm\log n\log m\log\log n\log\log m) + +O(m^2n) + O(mn^2)/\log_q n.$$

Для получения оценок битовой сложности нужно приведенные выше оценки умножить на $M(GF(q))$.

4.1.3 Пример построения схем для умножения и инвертирования в базисах низкой сложности

Базис низкой сложности при $n = 20$ можно построить перемножением оптимальных нормальных базисов поля $GF(2^5)$

$$B_\alpha = \{\alpha, \alpha^2, \alpha^4, \alpha^8, \alpha^{16}\}$$

и поля $GF(2^4)$

$$B_\beta = \{\beta, \beta^2, \beta^4, \beta^8\}.$$

Первый из них является базисом второго типа, так как $2^5 = -1 \bmod 11$, и согласно указанной выше теореме его минимальный многочлен равен $g_5 = x^5 + x^4 + x^2 + x + 1$, а второй является базисом первого типа, так как $2^2 = -1 \bmod 5$, и его минимальный многочлен равен $x^4 + x^3 + x^2 + x + 1$. Число единиц в его таблице умножения равно 63.

Непосредственным произведением базисов B_α, B_β является базис

$$B_\alpha \otimes B_\beta = \{\alpha\beta, \alpha\beta^2, \alpha\beta^4, \alpha\beta^8, \alpha^2\beta, \alpha^2\beta^2, \alpha^2\beta^4, \alpha^2\beta^8,$$

$$\alpha^4\beta, \alpha^4\beta^2, \alpha^4\beta^4, \alpha^4\beta^8, \alpha^8\beta, \alpha^8\beta^2, \alpha^8\beta^4, \alpha^8\beta^8, \alpha^{16}\beta, \alpha^{16}\beta^2, \alpha^{16}\beta^4, \alpha^{16}\beta^8.\}$$

Так как $2^i = 2^j \bmod 2^k - 1, 2^i = 2^l \bmod 2^m - 1$, то непосредственно проверяется, что этот базис является перестановкой нормального базиса

$$\{\alpha\beta, \alpha\beta^2, \alpha\beta^4, \alpha\beta^8, \alpha\beta^{16}, \ldots, \alpha\beta^{2^{19}}\}.$$

Чтобы найти координаты элемента поля в базисе $B_\alpha \otimes B_\beta$ нужно сначала найти его координаты, например, в базисе B_β, а потом полученные четыре элемента поля $GF(2^5)$ разложить в базисе B_α.

Как указывалось выше, базис B_β, как базис первого типа, является просто перестановкой стандартного базиса B^β, (так как $\beta^8 = \beta^3$). Поэтому переход от

базиса $B_\alpha \otimes B_\beta$ к базису $B_\alpha \otimes B^\beta$ тоже сводится к перестановке координат, и поэтому схемно делается бесплатно.

Также, как и выше, можно проверить, что переход от нормального базиса B_α к стандартному базису B_α и обратно можно сделать схемой из одних элементов XOR сложности $L(5) = L(4) + L(1) + 1 = 2$ и глубины 2.

Применяя этот переход к каждой из четырех координат данного элемента в базисе B^β, получаем, что переход от базиса $B_\alpha \otimes B^\beta$ к базису $B^\alpha \otimes B^\beta$ схемно делается со сложностью 8 и глубиной 2. Значит, с той же сложностью и глубиной делается переход от нормального базиса $B_{\alpha\beta}$ к базису $B^\alpha \otimes B^\beta$, являющемуся произведением стандартных базисов (и обратно тоже).

Для умножения двух произвольных элементов поля $GF(2^{20})$ в этом базисе можно разложить их по базису B^β, при этом координаты будут принадлежать полю $GF(2^5)$ и выполнить умножение в этом базисе. Как объяснялось выше, оно сводится к перемножению двух многочленов степени три над этим полем, умножению результата на x (бесплатному), а потом к делению с остатком полученного многочлена степени 7 на многочлен $x^4+x^3+x^2+x+1$. Также как и в предыдущем тексте получаем, что указанное деление выполняется с помощью 6 операций сложения в поле $GF(2^5)$, и соответствующая схема имеет сложность 30 и глубину 2. Деление многочлена $z_1x + \ldots z_7x^7$ на $x^4 + x^3 + x^2 + x + 1$ выполняется по формулам

$$z_5 + z_4, (z_1 + z_6) + z_4, (z_2 + z_5) + z_4, z_3 + z_4.$$

Если при перемножении многочленов число понадобившихся операций умножения в поле $GF(2^5)$ обозначить m, а число операций сложения — через a, то сложность умножения в базисе $B^\alpha \otimes B^\beta$ будет равна $mM(5) + 5a$, где $M(5)$— сложность умножения в поле $GF(2^5)$ в базисе B^α. Для уменьшения величины m можно применить схему Карацубы. Она умножает два многочлена степени один с тремя умножениями и четырьмя сложениями и глубиной 3, а два многочлена степени 3 девятью умножениями, 24 сложениями и глубиной 5, причем элемент умножения в любой цепи от входов к выходам только один. Поэтому можно взять $m = 9, a = 24$.

Умножение в поле $GF(2^5)$ в базисе B^α сводится к умножению двух многочленов степени четыре над полем $GF(2)$, умножению результата на x (бесплатному), а потом к делению с остатком полученного многочлена степени 9 на многочлен $g_5 = x^5+x^4+x^2+x+1$. Умножение многочленов степени четыре над полем $GF(2)$ имеет сложность $5^2 + 4^2 = 41$ и глубину 4. Деление многочлена $z_1x + \ldots z_9x^9$ на g_5 выполняется по непосредственно проверяемым формулам

$$(z_5 + z_7) + z_6, (z_5 + z_7) + (z_2 + z_9), z_1 + z_5 + z_8, z_4 + z_5 + z_6 + z_9$$

и имеет сложность 9. Общая сложность схемы умножения в поле $GF(2^5)$ равна $41 + 9 = 50$, а глубина равна 6.

Поэтому сложность умножения в базисе $B^\alpha \otimes B^\beta$ равна $9 \cdot 50 + 24 \cdot 5 + 30 = 600$, а глубина равна $6 + 4 + 2 = 12$, а если более аккуратно посмотреть, то

кажется даже 11. С учетом формул перехода от нормального базиса к базису $B^\alpha \otimes B^\beta$ общая сложность умножения в рассматриваемом нормальном базисе поля $GF(2^{20}$ равна 624, а глубина не более 16.

По теореме Ферма обратный элемент можно вычислить по формуле $\xi^{-1} = \xi^{2^n-2}$. Для возведения в степень надо построить аддитивную цепочку для числа $2^n - 2$, содержащую минимальное число сложений . Для этого сначала надо построить минимальную линейную аддитивную цепочку для числа $n - 1 = 19$. Например, такой будет цепочка

$$1, 2, 4, 8, 16, 18, 19.$$

После этого выписываются в ряд числа

$$2^1 - 1, 2^2 - 1, 2^4 - 1, 2^8 - 1, 2^{16} - 1, 2^{18} - 1, 2^{19} - 1,$$

и между ними вставляются последовательности удвоений. Так как удвоения реализуются путем соответствующей коммутации входов, то сложность инвертирования в поле $GF(2^{20})$ оценивается как

$$I(20) \le 6M(20),$$

где $M(20)$ сложность умножения в нормальном базисе. Аналогично глубина оценивается как

$$DI(20) \le 6DM(20).$$

Если использовать указанную выше схему для умножения в нормальном базисе, то сложность инвертирования будет $6 \cdot 616 = 3696$, а глубина $6 \cdot 16 = 96$.

Подобным же образом можно построить схему умножения и инвертирования в поле $GF(2^{22})$. Базис низкой сложности при $n = 22$ получим перемножением оптимальных нормальных базисов поля $GF(2^{11})$

$$B_\alpha = \{\alpha, \alpha^2, \alpha^4, \alpha^8, \ldots, \alpha^{1024}\}$$

и поля $GF(2^2)$

$$B_\beta = \{\beta, \beta^2\}.$$

Первый из них является базисом третьего типа, так как $2^{11} = 1 \bmod 23$, и согласно указанной в предыдущем тексте теореме его минимальный многочлен равен $g_{11} = x^{11} + x^{10} + x^8 + x^4 + x^3 + x^2 + 1$, так как вообще при $n = 3 \cdot 2^k - 1$ получится $2k + 3-$член

$$g_n = 1 + x^{2^{k-1}} + x^{2^{k-1}+2^{k-2}} + x^{2^{k-1}+2^{k-2}+2^{k-3}} +$$
$$+ \ldots + x^{2^k-1} + x^{2^k} + x^{2^{k+1}+2^{k-1}} + x^{2^{k+1}+2^{k-1}+2^{k-2}} +$$
$$x^{2^{k+1}+2^{k-1}+2^{k-2}+2^{k-3}} + \ldots + x^{2^{k+1}+2^k-1}$$

а второй является базисом первого типа и его минимальный многочлен равен x^2+x+1. Число единиц в его таблице умножения равно 63.

Непосредственным произведением базисов B_α, B_β является базис

$$B_\alpha \otimes B_\beta = \{\alpha\beta, \alpha\beta^2, \alpha^2\beta, \alpha^2\beta^2, \alpha^4\beta, \alpha^4\beta^2, \alpha^8\beta, \alpha^8\beta^2, \alpha^{16}\beta, \alpha^{16}\beta^2,$$

$$\ldots\ldots\ldots\ldots$$

$$\alpha^{1024}\beta, \alpha^{1024}\beta^2,\}$$

Непосредственно проверяется, что этот базис является перестановкой нормального базиса

$$\{\alpha\beta, (\alpha\beta)^2, \ldots, (\alpha\beta)^{2^{21}}\}.$$

Чтобы найти координаты элемента поля в базисе $B_\alpha \otimes B_\beta$ нужно сначала найти его координаты, например, в базисе B_β, а потом полученные два элемента поля $GF(2^{11})$ разложить в базисе B_α.

Базис B_β просто совпадает со стандартным базисом B^β.

Также, как в прошлом тексте, можно проверить, что переход от нормального базиса B_α к стандартному базису B_α и обратно можно сделать схемой из одних элементов XOR сложности $L(11) = L(8) + L(3) + 3 = 5 + 1 + 3 = 9$ и глубины 3.

Применяя этот переход к каждой из двух координат данного элемента в базисе B^β, получаем, что переход от базиса $B_\alpha \otimes B^\beta$ к базису $B^\alpha \otimes B^\beta$ схемно делается со сложностью 18 и глубиной 3. С той же сложностью и глубиной делается переход от нормального базиса $B_{\alpha\beta}$ к базису $B^\alpha \otimes B^\beta$, являющемуся произведением стандартных базисов (и обратно тоже).

Для умножения двух произвольных элементов поля $GF(2^{22})$ в этом базисе можно разложить их по базису B^β, при этом координаты будут принадлежать полю $GF(2^{11})$ и выполнить умножение в этом базисе. Оно сводится к перемножению двух линейных двучленов над этим полем, умножению результата на x (бесплатному), а потом к делению с остатком полученного многочлена степени 3 на многочлен x^2+x+1.

Нахождение остатка от деления многочлена $z_1x + z_2x^2 + z_3x^3$ на x^2+x+1 выполняется по формулам

$$z_3 + z_2 + (z_1 + z_2)x.$$

Для умножения двухчленов можно применить схему Карацубы. В результате умножение двух элементов $x_1\beta + x_2\beta^2$ и $y_1\beta + y_2\beta^2$ можно выполнить по формулам

$$\beta(\beta x_1y_1 + \beta^2(x_1y_2 + x_2y_1) + \beta^3x_2y_2) =$$

$$= \beta(x_1y_2 + x_2y_1 + x_2y_2 + \beta(x_1y_2 + x_2y_1) + x_1y_1)) =$$

$$= \beta((x_1 + x_2)(y_1 + y_2) + x_1y_1 + \beta((x_1 + x_2)(y_1 + y_2) + x_2y_2))$$

с помощью трех умножений и четырех сложений в поле $GF(2^{11})$.

Умножение в поле $GF(2^{11})$ в базисе B^α как и в предыдущем тексте сводится к умножению двух многочленов степени десять над полем $GF(2)$, умножению результата на x (бесплатному), а потом к делению с остатком полученного многочлена степени 21 на многочлен g_{11}.

Умножение многочленов степени десять над полем $GF(2)$ имеет сложность $11^2 + 10^2 = 221$ и глубину 5. Чуть увеличив глубину, можно уменьшить сложность, применяя метод Карацубы. Для этого применяем тождество

$$(f_0x^5 + f_1)(g_0x^5 + g_1) = f_0g_0x^{10} + ((f_0 + f_1)(g_0 + g_1) + (f_0g_0 + f_1g_1))x^5 + f_1g_1,$$

где $\deg f_1, g_1 \le 4, \deg f_0, g_0 \le 5$. Тогда оценка сложности будет $m(10) \le 2M(5) + M(4) + 10 + 10 + 9 + 4 + 5$, так как $\deg((f_0 + f_1)(g_0 + g_1) + (f_0g_0 + f_1g_1) \le 9$. Применяя для умножения многочленов степени 5 и 4 стандартный алгоритм, получаем что $m(5) = 6^2 + 5^2 = 61, m(4) = 5^2 + 4^2 = 41$, а глубина каждой из схем равна 4. Тогда $m(10) \le 2M(5) + M(4) + 38 = 201$, а глубина схемы равна 7, причем 5 самых старших и 5 самых младших коэффициентов имеют глубину не более 4. Если заменить стандартную схему умножения многочленов f_0, g_0 на схему, построенную методом Карацубы, то ее сложность будет равна $m(5) = 3m(2)+20 = 3\cdot 13+20 = 59$, то есть на 2 меньше стандартной, но глубина возрастет на 2 и будет равной 6. Число элементов умножения уменьшится и будет равно $3 \cdot 9 = 27$. Но глубина всей схемы умножения многочленов степени 10 от этого не увеличится. Таким образом глубина этой схемы равна $d(10) = 7$, а сложность равна $m(10) = 199$, причем мультипликативная сложность (число умножений) равна $2 \cdot 27 + 25 = 79$.

Выше было показано что при $n = 3 \cdot 2^k - 1$ дополнительная глубина, используемая для приведения произведения многочленов по модулю многочлена g_n будет не больше

$$2\lceil \log_2(2k + 2)\rceil + 1 = 2\log_2 \log_2 n + O(1),$$

а дополнительная сложность не больше

$$(n - 2)(2k + 3) - 2^k + 4 < 2n \log_2 n + n.$$

В рассматриваемом случае эти оценки равны 7 и 63.

В случае $n = 11$ их можно улучшить, если остаток от деления многочлена $z_1x + \ldots + z_{21}x^{21}$ на g_{11} вычислить по непосредственно проверяемым формулам

$$(z_{20}+z_{19})+((z_{12}+z_{14})+z_{11})+z_{10})x^{10}+(z_{21}+z_{20})+((z_{12}+z_{13})+z_{14})+z_{18}+z_9)x^9+$$

$$+(z_{21}+z_{17})+(z_{20}+z_{19})+z_8+((z_{12}+z_{13})+z_{11}))x^8+(z_{18}+z_{16}+z_{14}+z_7)x^7+$$

$$+(z_{21}+z_{17})+((z_{15}+z_{13})+z_6))x^6+(z_{21}+z_{20})+(z_{12}+z_{14})+z_{16}+z_5)x^5+$$

$$(z_{20}+z_{11})+(z_{15}+z_{13})+z_4+z_{19})x^4+((z_{20}+z_{11})+z_{18}+z_3)x^3+$$

$$+(z_{21}+z_{20})+((z_{12}+z_{14})+z_{11})+z_{17}+z_2)x^2+(z_{21}+z_{16}+((z_{12}+z_{13})+z_{14})+z_1)x+$$

$$(z_{21} + z_{20}) + ((z_{12} + z_{13}) + z_{11}) + z_{15}.$$

Сложность построенной на основе этой формулы схемы для приведения по модулю g_{11} равна 38, а глубина равна 4.

Глубину 4 имеет только выход, реализующий коэффициент

$$(z_{21} + z_{17}) + (z_{20} + z_{19}) + z_8 + ((z_{12} + z_{13}) + z_{11}))x^8.$$

Но если заметить, что глубина входа z_8 равна 7, а глубина входов z_{19}, z_{20} не больше 4, то глубина формулы $(z_{20}+z_{19})+z_8$ равна 8, поэтому глубина формулы $(z_{21}+z_{17})+(z_{20}+z_{19})+z_8$ равна 9, и очевидно такую же глубину имеет формула $(z_{12} + z_{13}) + z_{11}$, значит глубина коэффициента при x^8 равна 10 и такую же глубину имеет вся схема умножения в поле $GF(2^{11})$. Сложность же ее равна $M(11) = 199 + 38 = 237$, а мультипликативная сложность по-прежнему равна 79.

Поэтому сложность умножения в поле $GF(2^{22})$ в базисе $B^\alpha \otimes B^\beta$ равна $3 \cdot 237 + 4 \cdot 11 = 755$, глубина равна $10 + 1 = 11$, а мультипликативная сложность равна $3 \cdot 79 = 237$. С учетом формул перехода от нормального базиса к базису $B^\alpha \otimes B^\beta$ общая сложность умножения в рассматриваемом нормальном базисе поля $GF(2^{22}$ равна $755+3\cdot 18 = 809$, глубина равна $11+6 = 17$, а мультипликативная сложность по прежнему равна $3 \cdot 79 = 237$.

По теореме Ферма обратный элемент вычисляем по формуле $\xi^{-1} = \xi^{2^n-2}$. Для возведения в степень надо построить аддитивную цепочку для числа 2^n-2, содержащую минимальное число сложений . Для этого сначала надо построить минимальную линейную аддитивную цепочку для числа $n - 1 = 21$. Например, такой будет цепочка

$$1, 2, 4, 8, 16, 20, 21.$$

После этого выписываются в ряд числа

$$2^1 - 1, 2^2 - 1, 2^4 - 1, 2^8 - 1, 2^{16} - 1, 2^{20} - 1, 2^{21} - 1,$$

и между ними вставляются последовательности удвоений. Так как удвоения реализуются путем соответствующей коммутации входов, то сложность инвертирования в поле $GF(2^{22})$ оценивается как

$$I(22) \leq 6M(22),$$

где $M(22)$ сложность умножения в нормальном базисе. Аналогично глубина оценивается как

$$DI(22) \leq 6DM(22).$$

Если использовать указанную выше схему для умножения в нормальном базисе, то сложность инвертирования будет $6 \cdot 809 = 4854$, а глубина $6 \cdot 17 = 102$.

4.2 Операции в композитных полях

4.2.1 Схемы для инвертирования в композитных полях $GF(2^n)$ при использовании нормальных базисов

Выше было показано, что если в поле $GF(2^n)$ выбран нормальный базис для которого умножение выполняется со сложностью $M(n)$, то инвертирование в этом базисе можно выполнить со сложностью $O(M(n)\log_2 n$.

Допустим $n = km, (k,m) = 1$ и представим поле $GF(2^n)$ как расширение степени m поля $GF(2^k)$. Допустим, что битовая сложность умножения в поле $GF(2^k)$ равна $M(k)$. Как известно, произвольный, в частности нормальный базис $\{\alpha_1, \ldots, \alpha_m\}$ в поле $GF(2^m)$ является также базисом поля $GF(2^n)$ над подполем $GF(2^k)$. Для инвертирования произвольного элемента x воспользуемся формулой

$$x^{-1} = \frac{x^q x^{q^2} \ldots x^{q^{m-1}}}{x x^q \ldots x^{q^{m-1}}}, q = 2^k.$$

Так как $N(x) = x x^q \ldots x^{q^{m-1}}$ удовлетворяет равенству $N(x)^q = N(x)$, то $N(x)$ принадлежит подполю $GF(q)$, значит инвертирование $N(x)$ проводится в поле размерности k, поэтому его сложность можно обозначить $I(k)$. Умножение элемента $K(x) = x^q x^{q^2} \ldots x^{q^{m-1}}$ на элемент $N(x)^{-1}$ подполя размерности k выполняется со сложностью $mM(k)$ так как сводится к m умножениям в этом подполе. Возведение в степень q произвольного элемента $x = x_1\alpha_1 + \ldots x_m\alpha_m$ сводится к циклическому сдвигу на k позиций, так как $x_i^q = x_i, \alpha_1^q = \alpha_{k+1}$. Поэтому оно реализуется бесплатно. Аналогичное утверждение справедливо для возведения в любую степень вида q^i. Так как

$$K(x) = x^{q\frac{q^{m-1}-1}{q-1}}, x^{q\frac{q^{2i}-1}{q-1}} = \left(x^{q\frac{q^i-1}{q-1}}\right)^{q^i+1},$$

$$x^{q\frac{q^{2i+1}-1}{q-1}} = \left(x^{q\frac{q^{2i}-1}{q-1}}\right)^q x^q, N(x) = xK(x),$$

то $N(x), K(x)$ можно одновременно вычислить, выполнив $\lambda_2(m) + \nu_2(m) - O(1)$ умножений в поле $GF(2^n)$. Так как умножение в этом поле имеет сложность не больше $M(m)M(k)$, где $M(m)$ — сложность умножения в рассматриваемом нормальном базисе поля $GF(2^m)$, окончательно имеем оценку сложности инвертирования

$$I(n) \le I(k) + O(\log_2 m)M(m)M(k).$$

Если M(m) имеет такую же степенную оценку сложности в нормальном базисе, что и $M(k)$, то очевидно что $M(m)M(k) = O(M(mk)) = O(M(n)$. Так при этом $I(k) = O(\log kM(k)) = O(\log k)M(n)/m$, то при $\log k = O(m\log m)$ имеем

$$I(n) = O(\log_k)M(n)/m + O(M(n))\log m = O(M(n))\log m.$$

Если же обе оценки имеют вид $M(n) = O(n \log n \log\log n)$, а для этого достаточно, например, чтобы в полях $GF(2^k), GF(2^m)$ существовали и были выбраны нормальные базисы низкой сложности, то $M(m)M(k) = O(M(n)) \log m \log\log m$ и при $\log k = O(m \log^2 m \log\log m)$ имеем

$$I(n) = O(\log k)M(n)/m + O(M(n)) \log^2 m \log\log m = O(M(n)) \log^2 m \log\log m.$$

В обоих случаях $\log m = O(\log\log k) = O(\log\log n)$. Поэтому в первом из них для указанного поля и базиса в нем имеем оценку сложности инвертирования

$$I(n) = O(M(n)) \log\log n = n^{1+\epsilon},$$

а во втором случае имеем оценку

$$I(n) = O(M(n))(\log\log n)^2 \log\log\log n.$$

Итерируя эти оценки, можно получить примеры полей, для которых

$$I(n) = O(M(n))(\log\log\log\log n)^2 \log\log\log\log\log n$$

и т.д.

4.2.2 Примеры схем для инвертирования в полях $GF(2^n)$, где $n = mk$, $(m, k) = 1$

В первом примере возьмем $k = 2$ и положим $q = 2^m$.

В поле $GF(2^2)$ обозначим α, β элементы, отличные от нуля и единицы. Очевидно $\alpha + \beta = 1, \alpha\beta = 1$. Представим элементы поля $GF(2^n) = GF(q^2)$ в виде $x\alpha + y\beta, x, y \in GF(q)$. Положим

$$N(x\alpha+y\beta) = (x\alpha+y\beta)(x\beta+y\alpha) = xy(\alpha^2+\beta^2)+(x^2+y^2)\alpha\beta(x^2+y^2)+xy \in GF(q),$$

тогда

$$(x\alpha + y\beta))^{-1} = (x\beta + y\alpha)/N = xN^{-1}\beta + yN^{-1}\alpha.$$

Отсюда получается рекуррентная оценка сложности для инверсии

$$I(2m) \le I(m) + 2M(m) + N(n),$$

где $N(n)$ — сложность вычисления нормы, а $M(m)$ — сложность умножения в поле $GF(2^m)$. Так как

$$N(x\alpha + y\beta) = (x^2 + y^2) + xy = (x + y)^2 + xy,$$

то при выборе в поле $GF(q)$ нормального базиса имеем $N(n) \le M(m) + m$. Поэтому

$$I(2m) \le I(m) + 3M(m) + m.$$

Умножение в поле $GF(2^n)$ имеет вид

$$(x_1\alpha + x_2\beta)(y_1\alpha + y_2\beta) = x_1y_1\alpha^2 + (x_2y_1 + x_1y_2)\alpha\beta + x_2y_2\beta^2 =$$

$$= x_1y_1\beta + (x_2y_1 + x_1y_2)(\alpha+\beta) + x_2y_2\alpha = (x_2y_2 + (x_2y_1 + x_1y_2))\alpha + (x_1y_1 + (x_2y_1 + x_1y_2))\beta =$$

$$(x_2y_1 + (x_2 + x_1)y_2)\alpha + (x_2y_1 + x_1(y_1 + y_2))\beta,$$

откуда имеем $M(2n) \le 3M(n) + 4n$.

Рассмотрим второй пример, в котором, в частности, дадим применение полученным выше схемам. Пусть $n = 22 \cdot 23 = 506$. Возьмем кратчайшую линейную аддитивную цепочку

$$1, 2, 3, 4, 8, 11, 22, 23,$$

из нее строим цепочку степеней x с показателями

$$1, \frac{q^2-q}{q-1}, \frac{q^3-q}{q-1}, \frac{q^4-q}{q-1}, \frac{q^8-q}{q-1},$$

$$\frac{q^{11}-q}{q-1}, \frac{q^{22}-q}{q-1}, \frac{q^{23}-q}{q-1}, \frac{q^{23}-1}{q-1},$$

значит для вычисления $N(x), K(x)$ достаточно сделать 7 умножений в поле $GF\left(2^{506}\right)$. Оценим $M(506)$. Можно для умножения использовать две схемы. Первая из них сводит умножение в поле $GF(2^{506})$ к умножению в оптимальном нормальном базисе поля $GF\left(2^{23}\right)$, которое в свою очередь может быть сведено к вычислению некоторой системы билинейных форм с мультипликативной сложностью 243 и аддитивной сложностью 784, а так как переменные этих форм принадлежат полю $GF\left(2^{22}\right)$, в котором умножение делается со сложностью $M(22) = 809$ а сложение со сложностью $A(22) = 22$, то отсюда имеем оценку

$$M(506) \le 243 \cdot 809 + 784 \cdot 22 = 213835.$$

Аналогичным образом, рассматривая поле $GF(2^{506})$ как расширение 22-й степени поля $GF(2^{23})$, имеем худшую оценку

$$M(506) \le 237 \cdot 1027 + 772 \cdot 23 = 261155.$$

Тогда

$$I(506) \le I(22) + 23M(22) + 7M(506) \le 4854 + 23 \cdot 809 + 7 \cdot 213835 = 1520306.$$

Но можно выбрать представление $n = 23 \cdot 22$. Тогда для вычисления $N(x), K(x)$ надо взять линейную аддитивную цепочку

$$1, 2, 3, 4, 8, 11, 22,$$

и из нее построить цепочку степеней x с показателями

$$1, \frac{q^2-q}{q-1}, \frac{q^3-q}{q-1}, \frac{q^4-q}{q-1}, \frac{q^8-q}{q-1},$$

$$\frac{q^{11}-q}{q-1}, \frac{q^{22}-q}{q-1}, \frac{q^{22}-1}{q-1},$$

и поэтому для вычисления $N(x), K(x)$ достаточно 6 умножений в поле $GF\,(2^{506})$. Тогда получается лучшая оценка

$$I(506) \le I(23) + 22M(23) + 6M(506) \le 5988 + 22 \cdot 1027 + 6 \cdot 213835 = 1311592.$$

Если непосредственно применить для инвертирования в поле $GF\,(2^{506})$ метод аддитивных цепочек, то схема получится примерно в два раза сложнее.

Конечно, сложность указанной схемы слишком высока для ее реальной реализации в виде чипа. Но уже для поля размерности вдвое меньшей, а именно 253, подобным же образом можно построить схемы со сложностью

$$M(253) \le 243 \cdot 237 + 784 \cdot 11 = 66215,$$

$$I(253) \le I(23) + 11M(23) + 5M(253) \le 5988 + 11 \cdot 1027 + 5 \cdot 66215 = 348360.$$

Теперь представим поле $GF\,(2^{506})$ в виде квадратичного расширения поля $GF\,(2^{253})$ и применяя указанный выше метод построения схем для умножения и инвертирования в квадратичном расширении, построим эти схемы для поля $GF\,(2^{506})$ сложности

$$M(506) \le 3M(253) + 4 \cdot 253 = 199657.$$

$$I(506) \le I(253) + 3M(253) + 253 \le 348360 + 199657 - 759 = 547258.$$

Такую схему вполне можно реализовать в виде чипа одновременно со схемами умножения и сложения. Схемы для полей приблизительно вдвое меньших размерностей используются в эллиптических криптосистемах.

Использование подобных же методов позволяет для других размерностей строить и менее сложные схемы. Например, можно построить схемы с оценками сложности

$$I(330) \le I(11) + 521L(M(11)) + 1417 \cdot 11 = 140249,$$

$$I(690) \le I(23) + 521L(M(23)) + 1417 \cdot 23 = 573646.$$

В общем случае оценка сложности сильно зависит от структуры разложения на множители размерности данного поля и от выбора порядка сомножителей при построении соответствующей башни расширений. Выгоднее всего порядок выбирать монотонно убывающий.

4.2.3 Схемы для быстрого умножения в полях $GF(2^n)$ размерности $n = m^s$ при растущем s

Пусть $k < s$ параметр, значение которого укажем позднее. Выберем наименьшее r такое, что $2^{m^r} \geq 2m^k - 1$, и $r = s \bmod k$. Очевидно, что $r = O(k)$. Представим поле $GF\left(2^{m^s}\right)$ как $GF\left(q^{m^{s-r}}\right)$ — расширение поля $GF(q), q = 2^{m^r}$. Далее, поле

$$GF\left(2^{m^s}\right) = GF\left(q^{m^{s-r}}\right) = GF\left(q^{m^{kl}}\right)$$

представим в виде башни расширений

$$GF(q) \subset GF\left(q^{m^k}\right) \subset GF\left(\left(q^{m^k}\right)^{m^k}\right) = GF\left(q^{m^{2k}}\right) \subset \ldots \subset GF\left(q^{m^{kl}}\right).$$

Для каждого этажа башни

$$GF(q_i) = GF\left(q^{m^{ik}}\right) \subset GF\left(\left(q^{m^{ik}}\right)^{m^k}\right) = GF\left(q^{m^{(i+1)k}}\right) = GF(q_{i+1})$$

выберем стандартный базис $\{1, \alpha, \ldots, \alpha^{m^k-1}\}$, определяемый неприводимым над полем $GF\left(q_i\right)$ многочленом $p_i(x)$ степени m^k. Тогда произвольный элемент поля $GF(q_{i+1})$ можно представить в виде k-мерного вектора с компонентами из поля $GF(q_i)$. В результате произвольный элемент поля $GF(q_i)$ можно представить в виде m^{ki}—мерного вектора с компонентами из поля $GF(q)$. Умножение в поле $GF(q_{i+1})$ можно свести к умножению по модулю многочлена p_i двух произвольных многочленов степени $t = m^k - 1$ над полем $GF(q_i)$. Для приведения полученного в результате этого умножения многочлена $f(x)$ степени $2t-1$ по модулю данного неприводимого многочлена $p(x)$ степени t, как известно (см., например [25]) достаточно вычислить заранее многочлен $R(x)$ степени $t-1$ такой, что $x^{2t-1} = R(x)p(x) + g(x)$, где степень $g(x)$ меньше t, представить $f(x)$ в виде $f_1x^t + f_2$, где f_2 и f_1 имеют степень меньше t (это делается бесплатно), потом найти частное $h(x)$ от деления $f(x)$ на $p(x)$ по формуле $h(x)x^{t-1} + h_1(x) = f_1(x)R(x)$, где h_1 имеет степень меньше $t-1$ и h имеет степень меньше t, т.е. с помощью умножения многочленов степени $t-1$. Тогда остаток от деления $f(x)$ на $p(x)$ находится по формуле $f \bmod p = f - ph$, то есть с помощью еще одного умножения многочленов степени t. Действительно, $f - ph = f_1x^t + f_2 - ph$ имеет степень меньше t так как многочлен

$$(f_1x^t - ph)x^{t-1} = f_1(x)x^{2t-1} - p(x)h(x)x^{t-1} =$$

$$= f_1(x)(R(x)p(x) + g(x)) - p(x)h(x)x^{t-1} =$$

$$(h(x)x^{t-1} + h_1(x))p(x) + f_1(x)g(x) - p(x)h(x)x^{t-1} = h_1(x)p(x) + f_1(x)g(x)$$

имеет степень меньше $2t-1$.

Таким образом, умножение в поле $GF(q_{i+1})$ сводится к трем умножениям многочленов степени t над полем $GF(q_i)$ и t сложениям в этом поле. Для

умножения двух произвольных многочленов f, g степени t над полем $GF(q_i)$ можно сначала вычислить значения $f(a_i), g(a_i)$ на произвольных $2t+1$ элементах его подполя $GF(q)$ (ведь $2t+1 = 2m^k - 1 \leq 2^{m^r} = q$,) потом выполнить $2t+1$ умножение в поле $GF(q_i)$, а потом, используя интерполяционную формулу, восстановить по значениям $h(a_i) = f(a_i)g(a_i)$ коэффициенты произведения $h(x) = f(x)g(x)$. Для выполнения всех этих операций с помощью схемы Горнера и формулы Лагранжа требуется, как известно, $O(t^2)$ операций сложения и умножения на элементы подполя $GF(q)$ в поле $GF(q_i)$. Используя метод быстрой интерполяции [5], указанную оценку можно уменьшить до $O(t)\log^2 t$. В некоторых случаях ее можно еще усилить, например применяя преобразование Фурье порядка $q-1$ над полем $GF(q)$ или метод [90]. Для теоретических целей нас устраивает простейшая оценка, из которой следует, что сложность умножения элементов поля $GF(q_{i+1})$ оценивается как

$$M(GF(q_{i+1})) \leq 3(2m^k - 1)M(GF(q_i)) + O(m^{2k})m^{ik}(\log_2 q)^{\log_2 3},$$

так как сложность сложения в поле $GF(q_i)$ равна его размерности $\log_2 q_i = m^{ik}\log_2 q$, а сложность умножения на элементы подполя $GF(q)$ равна $M(GF(q))m^{ik} = O(\log_2 q)^{\log_2 3}m^{ik}$, так как это умножение сводится к m^{ik} умножениям в поле $GF(q)$. Если обозначить $M(GF(q_i))$ через $M(i)$, а $3(2m^k-1)$ через a, полученное рекуррентное неравенство переписывается в виде

$$M(i+1) \leq aM(i) + bm^{(i+1)k},$$

где

$$b = O(a)(m^k m^{r\log_2 3}) = O(a)m^{O(k)},$$
$$M(0) = M(GF(q)) = O(\log_2 q)^{\log_2 3}.$$

Применяя индукцию, отсюда имеем

$$M(l) \leq a^l M(0) + b\left(a^{l-1}m^k + a^{l-2}m^{2k} + \ldots + m^{lk}\right),$$

следовательно

$$M(l) \leq a^l M(0) + a^l b\frac{1-(m^k/a)^{l+1}}{1-m^k/a} \leq a^l M(0) + \frac{a^l b}{1-m^k/a} \leq$$

$$\leq a^l M(0) + 3a^l b/2 = O(a^{l+1})m^{O(k)} = O(a)m^{O(k)}2^{l\log_2 3(2m^k-1)}.$$

Поэтому, так как $q_l = q^{m^{kl}}$, то $l = (\log_2\log_q q_l)/\log_2 m^k$ и при $\log_2 q_l = n$

$$M(GF(2^n)) = M(GF(q_l)) = O(m^{r\log_2 3})2^{l\log_2 3(2m^k-1)} = m^{O(k)}2^{\frac{\log_2\log_q q_l\log_2 3(2m^k-1)}{\log_2 m^k}} =$$

$$m^{O(k)}(\log_q q_l)^{\log_{m^k} 3(2m^k-1)} = m^{O(k)}(\log_2 q_l)^{\log_{m^k} 3(2m^k-1)} = m^{O(k)}n^{\log_{m^k} 3(2m^k-1)}.$$

Так как $\log_{m^k} 3(2m^k-1)$ стремится к 1 при растущем m^k, то из доказанного следует, что для любого $\epsilon > 0$ при любом m, $n = m^s$ и $s \geq s_\epsilon$ имеем

$$M\left(GF(2^n)\right) = M\left(GF(2^{m^s})\right) = n^{1+\epsilon}.$$

4.2.4 Схемы для быстрого инвертирования в полях $GF(2^n)$ размерности $n = m^s$ при растущем s

Как и в предыдущей секции пусть $k < s$ параметр, значение которого укажем позднее. Выберем наименьшее r такое, что $q = 2^{m^r} \geq 2m^k - 1$, и $r = s \bmod k$. Поле

$$GF\left(2^{m^s}\right) = GF\left(q^{m^{s-r}}\right) = GF\left(q^{m^{kl}}\right)$$

представим в виде башни расширений

$$GF(q) \subset GF\left(q^{m^k}\right) \subset GF\left(\left(q^{m^k}\right)^{m^k}\right) = GF\left(q^{m^{2k}}\right) \subset \ldots \subset GF\left(q^{m^{kl}}\right).$$

Для каждого этажа башни

$$GF(q_i) = GF\left(q^{m^{ik}}\right) \subset GF\left(\left(q^{m^{ik}}\right)^{m^k}\right) = GF\left(q^{m^{(i+1)k}}\right) = GF(q_{i+1})$$

выберем стандартный базис $\{1, \alpha, \ldots, \alpha^{m^k-1}\}$, определяемый неприводимым над полем $GF\left(q_i\right)$ многочленом $p_i(x)$ степени m^k. Тогда произвольный элемент поля $GF(q_{i+1})$ можно представить в виде k-мерного вектора с компонентами из поля $GF(q_i)$ и произвольный элемент поля $GF(q_i)$ можно представить в виде m^{ki}—мерного вектора с компонентами из поля $GF(q)$. Для инвертирования элемента $x \in GF(q_i)$ применим формулу

$$x^{-1} = \frac{K_{q_{i-1}}(x)}{N_{q_{i-1}}(x)}, N_Q(x) = xK_Q(x), K_Q(x) = x^Q x^{Q^2} \ldots x^{Q^{m^k-1}}, Q = q_{i-1}.$$

Так как $N_Q(x)^q = N_Q(x)$, то $N_Q(x) \in GF(Q) = GF(q_{i-1}$, значит вычисление $N_Q(x)^{-1}$ делается со сложностью

$$I(\log_2 Q) = I(\log_2 q_{i-1}) = I\left(m^{(i-1)k+r}\right).$$

Умножение $K_Q(x)N_Q(x)^{-1}$ выполняется со сложностью $m^k M\left(m^{(i-1)k+r}\right)$ так как элемент $K_Q(x)$ поля $GF(q_i)$ представляется m^k-мерным вектором над полем $GF(q_{i-1})$. Умножение $xK_Q(x)$ выполняется со сложностью $M(\log_2 q_i) = M\left(m^{ik+r}\right)$. Умножение $x^Q x^{Q^2} \ldots x^{Q^{m^k-1}}$ выполняется со сложностью $(m^k - 2)M\left(m^{ik+r}\right)$. Остается оценить сложность вычисления степеней x^{Q^i} в поле $GF(q_i)$. Каждая из этих операций является линейным оператором над подполем $GF(Q) = GF(q_{i-1})$, поэтому ее сложность можно оценить как $O(m^{2k})M\left(m^{(i-1)k+r}\right)$. Значит, сложность вычисления всей системы степеней $x^Q, x^{Q^2}, \ldots, x^{Q^{m^k-1}}$ оценивается как $O(m^{3k})M\left(m^{(i-1)k+r}\right)$. Предполагая справедливым естественное неравенство $aM(n) \leq M(an)$, имеем окончательную оценку

$$I(\log_2 q_i) = I\left(m^{ik+r}\right) \leq I(\log_2 q_{i-1}) + O(m^{2k})M(\log_2 q_i).$$

Из этой оценки по индукции с помощью неравенства $aM(n) \leq M(an)$ выводим, что

$$I(m^s) = I\left(m^{ik+r}\right) = I(\log_2 q_l) \leq I(m^r) + O(m^{2k})M(\log_2 q_l) =$$

$$= I(m^r) + O(m^{2k})M(m^s) = m^{O(k)} + O(m^{2k})M(m^s) = O(m^{2k})M(m^s).$$

При s стремящемся к бесконечности и любых фиксированных m, k имеем

$$I(m^s) = O(m^{2k})M(m^s) = O(M(m^s)).$$

Для любого $\epsilon > 0$, выбирая k при любом фиксированном m также, как и в предыдущем разделе, имеем

$$M\left(GF(2^n)\right) = M\left(GF(2^{m^s})\right) < n^{1+\epsilon/2},$$

$$I(n) = I(m^s) = O(m^{2k})M(m^s) = 2^{O(1)/\epsilon} n^{1+\epsilon/2} < n^{1+\epsilon}.$$

4.2.5 Некоторые усовершенствования схем предыдущей секции

Хотя метод предыдущей секции теоретически является в некотором смысле почти оптимальным, при применении к полям небольших размерностей (меньше тысячи) он не эффективен. Усовершенствуем его.

Для каждого этажа башни

$$GF(q_i) = GF\left(q^{m^{ik}}\right) \subset GF\left(\left(q^{m^{ik}}\right)^{m^k}\right) = GF\left(q^{m^{(i+1)k}}\right) = GF(q_{i+1})$$

выберем стандартный базис $\{1, \alpha, \ldots, \alpha^{m^k-1}\}$, определяемый неприводимым над полем $GF\left(q_i\right)$ многочленом $p_i(x)$ степени m^k таким образом, чтобы элемент α порождал нормальный базис $\{\alpha, \alpha^Q \ldots, \alpha^{Q^{m^k-1}}\}$, где $Q = q_i$. Если есть возможность, элемент α можно выбрать так, чтобы нормальный базис имел минимальную сложность. Если она будет, например, линейна относительно m^k, тогда для умножения в поле $GF(q_{i+1})$ можно использовать метод Месси-Омуры.

Другой метод основан на переходе к стандартному базису, выполнении умножения в нем, и возвращении опять к нормальному базису. Для его использования естественно выбрать элемент α так, чтобы минимизировать сложность перехода к стандартному базису и обратно. Грубая универсальная оценка этой сложности равна

$$m^{2k}M(GF(q_i)) + (m^{2k} - m^k)m^{ik+r},$$

так как для выполнения как прямого, так и обратного преобразования координат требуется не более m^{2k} умножений и не более $m^{2k} - m^k$ сложений в поле $GF(q_i)$, имеющем размерность m^{ik+r}.

В предыдущих разделах было показано, что умножение в стандартном базисе поля $GF(q_{i+1})$ оценивается сверху как

$$M(GF(q_{i+1})) \leq 3M(m^k)M(GF(q_i)) + 3A(m^k)m^{ik}r^{\log_2 3},$$

где $M(n), A(n)$ есть соответственно мультипликативная и аддитивная сложности умножения многочленов степени n над данным полем. Эту оценку в практическом применении иногда можно немного уточнить, если большинство коэффициентов многочлена p_i будут нули или единицы. Тогда для приведения по модулю этого многочлена будет эффективен обычный школьный алгоритм деления с остатком. В общем случае ранее была получена асимптотически более точная оценка

$$M(GF(q_{i+1})) \leq 3(2m^k - 1)M(GF(q_i)) + O(m^{2k})m^{ik}r^{\log_2 3}.$$

Для вычисления $K_{q_i}(x), N_{q_i}(x)$ переходим от стандартного базиса к нормальному со сложностью

$$m^{2k}M(GF(q_i)) + (m^{2k} - m^k)m^{ik+r}$$

и бесплатно с помощью циклических сдвигов получаем в нормальном базисе систему степеней

$$x^Q, x^{Q^2}, \ldots, x^{Q^{m^k-1}}.$$

Далее вычисляем $K(x) = K_{q_i}(x), N(x) = N_{q_i}(x)$ как и в предыдущих разделах по формулам

$$K(x) = x^{Q\frac{Q^{t-1}-1}{Q-1}}, x^{Q\frac{Q^{2i}-1}{Q-1}} = \left(x^{Q\frac{Q^i-1}{Q-1}}\right)^{Q^i+1},$$

$$x^{Q\frac{Q^{2i+1}-1}{Q-1}} = \left(x^{Q\frac{Q^{2i}-1}{Q-1}}\right)^{Q} x^Q, N(x) = xK(x),$$

где $t = m^k$выполнив $\lambda_2(t) + \nu_2(t) - O(1)$ операций умножения в нормальном базисе. Сложность каждого такого умножения оценивается как

$$M(GF(q_{i+1})) \leq 3(2m^k - 1)M(GF(q_i)) + O(m^{2k})m^{ik}r^{\log_2 3} +$$

$$3m^{2k}M(GF(q_i)) + 3(m^{2k} - m^k)m^{ik+r} =$$

$$= 3(m^{2k} + 2m^k - 1)M(GF(q_i)) + O(m^{2k})m^{ik}r^{\log_2 3},$$

так как оно выполняется с помощью двухкратного перехода к стандартному базису, умножения в нем и перехода опять к нормальному базису. Оценку $\lambda_2(t)+\nu_2(t)-O(1)$ для числа операций умножения в нормальном базисе можно иногда чуть уточнить, используя, как и в предыдущих разделах кратчайшие линейные аддитивные цепочки.

Возьмем кратчайшую линейную аддитивную цепочку для числа $t = m^k$

$$a_0 = 1, a_1 = 2, a_2, \ldots, a_L = t$$

длины $L = L(t)$. По ней можно построить цепочку, содержащую степени x с показателями

$$\frac{Q^{a_1}-Q}{Q-1}, \frac{Q^{a_2}-Q}{Q-1}, \ldots, \frac{Q^{t}-Q}{Q-1},$$

пользуясь формулами

$$\frac{Q^{a_i}-Q}{Q-1} = \frac{Q^{a_j+a_h}-Q}{Q-1} = Q^{a_h}\frac{Q^{a_j}-Q}{Q-1} + \frac{Q^{a_h}-Q}{Q-1}.$$

Так как возведение в степень Q^n в нормальном базисе делается бесплатно, а

$$x^{\frac{Q^{a_1}-Q}{Q-1}} = x^Q,$$

то для вычисления

$$K(x) = x^{\frac{Q^{t}-Q}{Q-1}}$$

требуется только $L-1 = L(t)-1 = L(m^k)-1$ операций умножения. Еще одно умножение требуется для вычисления $N(x) = xK(x)$. В итоге сложность совместного вычисления $K(x), N(x)$ оценивается как

$$L(m^k)M(GF(q_{i+1})) \le L(m^k)\left(3(m^{2k}+2m^k-1)M(GF(q_i)) + O(m^{2k})m^{ik}r^{\log_2 3}\right).$$

Используя формулу $x^{-1} = K(x)/N(x)$, получаем рекуррентную оценку сложности инвертирования

$$I(\log_2 q_{i+1}) = I\left(m^{(i+1)k+r}\right) \le I\left(m^{ik+r}\right) + m^k M(GF(q_i)) +$$

$$+L(m^k)\left(3(m^{2k}+2m^k-1)M(GF(q_i)) + O(m^{2k})m^{ik}r^{\log_2 3}\right).$$

Очевидно

$$L(t) \le \lambda_2(t) + \nu_2(t) - 1 \le 2\log_2 t \le 2k\log_2 m.$$

Используя метод множителей для построения линейных аддитивных цепочек, можно по индукции доказать, что $L(m^k) \le kL(m)$.

Из полученных выше оценок по индукции с помощью неравенства $aM(n) \le M(an)$ выводим, лучшую оценку, чем полученная ранее.

$$I(m^s) = I\left(m^{ik+r}\right) = I(\log_2 q_l) \le I(m^r) + O(km^{2k}\log_2 m)M(\log_2 q_{l-1}) =$$

$$= I(m^r) + O(km^k \log_2 m)M(m^s) = m^{O(k)} + O(m^{2k})M(m^s) = O(km^k\log_2 m)M(m^s).$$

4.2.6 Примеры построения схем для полей $GF(2^{2^k})$

Рассмотрим пример $m = 2, n = m^s$. Выберем $k = 8$, тогда $\log_{m^k} 3(2m^k - 1) = \log_{256} 1533 < 1.33$ Тогда при растущем n имеем оценку сложности умножения в поле $GF(2^{2^n})$

$$M(2^n) = O(2^{n \log_{m^k} 3(2m^k - 1)}) = O(2^{n \log_{256} 1533}),$$

и оценку сложности инвертирования $I(2^n) = O(M(2^n))$. Эти оценки асимптотически лучше полученных в [141], которые имеют вид

$$M(2^n) = O(2^{n \log_2 3}), I(2^n) = O(M(2^n)).$$

Рассмотрим еще более конкретный пример, выбрав $s = 10$. Тогда положим $r = 3$ (так как $2^{2^3} > 2^8 - 1$) и поле $GF(2^{1024})$ представляем в виде расширения степени 2^8 подполя $GF(2^8)$. Элементы поля представляем в виде многочленов по модулю некоторого неприводимого над полем $GF(2^8)$ многочлена степени 2^7. Обычное умножение двух таких многочленов f, g указанным выше методом сводится к вычислению в поле $GF(2^8)$ значений $f(\alpha), g(\alpha)$ для всех ненулевых элементов поля, попарному перемножению полученных значений $f(\alpha)g(\alpha)$ и восстановлению многочлена $h = fg$ степени $2^8 - 2$ интерполяционным методом. Так как узлы интерполяции совпадают с корнями $2^8 - 1$ степени из единицы, то вычисление значений $f(\alpha), g(\alpha)$ совпадает с двукратным вычислением преобразования Фурье порядка $2^8 - 1 = 255$, а вычисление коэффициентов многочлена h совпадает с вычислением обратного преобразования Фурье того же порядка порядка.

Обозначим через $FM(n)$ мультипликативную, а через $FA(n)$ аддитивную сложность преобразования Фурье порядка n. Согласно теореме Гуда-Томаса (см., напр. [10] или [53]) имеем

$$FM(255) = 15FM(17) + 17FM(15), FA(255) = 15FA(17) + 17FA(15),$$

$$FM(15) = 3FM(5) + 5FM(3), FA(15) = 3FA(5) + 5FA(3).$$

Так как преобразование Фурье третьего порядка выполняется по формулам

$$f_1 = x_1 + (x_2 + x_3), f_2 = x_1 + x_2\omega + x_3\omega^2 = x_1 + (x_2 + x_3)\omega + x_3,$$

$$f_3 = x_1 + x_2\omega^2 + x_3\omega^4 = x_1 + x_2(\omega + 1) + x_2\omega = x_1 + (x_2 + x_3)\omega + x_2, \omega \in GF(2^2),$$

очевидно $FM(3) = 1, FA(3) = 6$, причем во всех операциях умножения один из сомножителей (скаляр) принадлежит подполю $GF(2^2)$.

Согласно теореме Райдера (см., напр. [10]) при простом m справедливо неравенство $FM(m) \leq CM(m-1)$, $FA(m) \leq CA(m-1) + 2(m-1)$, где $CM(m-1), CA(m-1)$ — мультипликативная и аддитивная сложности циклической свертки порядка $m - 1$, то есть операции умножения многочленов степени $m - 2$ по модулю $x^{m-1} + 1$.

Циклическая свертка четного порядка m может быть вычислена по формулам Карацубы

$$(a_0 + a_1 x^{m/2})(b_0 + b_1 x^{m/2}) \bmod x^m + 1 =$$
$$= a_0 b_0 + a_1 b_1 + ((a_0 + a_1)(b_0 + b_1) + a_0 b_0 + a_1 b_1) x^{m/2} \bmod x^m + 1 =$$
$$= a_0 b_0 + a_1 b_1 + c_1 + c_0 x^{m/2},$$

где

$$(a_0 + a_1)(b_0 + b_1) + a_0 b_0 + a_1 b_1 = c_0 + c_1 x^{m/2},$$

a_i, b_i, c_i— многочлены степени $m/2-1$. Из них вытекает следующая оценка сложности циклической свертки

$$CM(m) \le 3M(m/2), CA(m) \le 3A(m/2) + 4m - 4,$$

где $M(n), A(n)$ — мультипликативная и аддитивная сложности умножения многочленов степени $n-1$. Из приведенной выше оценки получается, что $CM(4) \le 3 \cdot M(2) = 9, CA(4) = 24$, значит $FM(5) \le CM(4) = 9$, $FA(5) \le CA(4) + 8 = 32$.

Тривиальный способ вычисления преобразования Фурье 5-го порядка требует 16 умножений и 20 сложений, зато умножения в нем выполняются на скаляры из подполя $GF(2^4)$. Использование алгоритма Блюстейна [10] при нечетных m позволяет уменьшить число умножений до $m(m-1)/2$. Можно проверить, что при $m = 5$ среди 10 умножений 6 выполняется на скаляры из поля $GF(2^2)$, а остальные умножения выполняются на скаляры из поля $GF(2^4)$. Действительно, первообразный корень ω пятой степени из единицы можно выбрать в поле $GF(2^4)$, а в алгоритме Блюстейна используется деление многочленов четвертой степени на трехчлены

$$(x + \omega)(x + \omega^4) = x^2 + \alpha x + 1, (x + \omega^2)(x + \omega^3) = x^2 + \beta x + 1,$$

где $\beta = \alpha^2, \alpha\beta = 1, \alpha + \beta = 1$ принадлежат подполю $GF(2^2)$.

Поэтому для преобразования Фурье порядка 15 можно построить две разные схемы. У одной из них

$$FM(15) = 3FM(5) + 5FM(3) = 32, FA(15) = 3FA(5) + 5FA(3) = 126,$$

причем 6 умножений выполняются на скаляры из поля $GF(2^2)$, а у другой

$$FM(15) = 3FM(5) + 5FM(3) = 35, FA(15) = 3FA(5) + 5FA(3) = 90,$$

причем 23 умножения выполняются на скаляры из поля $GF(2^2)$, а остальные 12 умножений выполняются на скаляры из поля $GF(2^4)$.

Оценка Карацубы сложности умножения многочленов четной степени, основанная на формуле

$$(a_0 + a_1 x^{m/2})(b_0 + b_1 x^{m/2}) =$$
$$= a_0 b_0 + a_1 b_1 x^m + ((a_0 + a_1)(b_0 + b_1) + a_0 b_0 + a_1 b_1) x^{m/2},$$

также имеет вид $M(m) \leq 3M(m/2), A(m) \leq 3A(m/2) + 4m - 4$ [101], [25]. Применяя ее, имеем

$$M(4) \leq 3M(2) = 9, A(4) \leq 3A(2) + 12 = 24,$$

$$M(8) \leq 3M(4) = 27, A(8) \leq 3A(4) + 28 = 3 \cdot 24 + 28 = 100,$$

откуда следует, что

$$CM(16) \leq 3M(8) \leq 81, CA(16) \leq 3A(8) + 60 \leq 360,$$

значит

$$FM(17) \leq CM(16) \leq 81, FA(17) \leq CA(16) + 32 \leq 392.$$

Окончательно имеем схему для преобразования Фурье 255 порядка содержащую в одном варианте $FM(255) = 15FM(17) + 17FM(15) = 1810$ умножений, причем из них 391 умножений на скаляр из поля $GF(2^2)$ и 204 умножений на скаляр из поля $GF(2^4)$ и $FA(255) = 15FA(17) + 17FA(15) = 7410$ сложений, а в другом варианте $FM(255) = 15FM(17) + 17FM(15) = 1759$ умножений, причем из них 85 умножений на скаляр из поля $GF(2^2)$ и $FA(255) = 15FA(17) + 17FA(15) = 8022$ сложений, Каждое из сложений имеет битовую сложность 8. В поле $GF(2^8)$ можно выбрать такую башню расширений, что сложность умножения в нем будет равна 106, сложность умножения в подполе $GF(2^4)$ равна 29 а сложность умножения в подполе $GF(2^2)$ равна 7. Тогда сложность умножения на скаляр из подполя $GF(2^4)$ будет равна $2 \cdot 29 = 58$, а сложность умножения на скаляр из подполя $GF(2^2)$ будет равна $4 \cdot 7 = 28$. Заметим, что сложность схемы умножения в поле $GF(2^8)$ в стандартном базисе будет больше 106 даже при применении метода Карацубы.

Тогда общая сложность $F(255)$ схемы для преобразования Фурье порядка 255 в первом варианте равна $391 + \cdot 28 + 204 \cdot 58 + 1215 \cdot 106 + 7410 \cdot 8 = 210850$. а во втором варианте $85 \cdot 28 + 1674 \cdot 106 + 8022 \cdot 8 = 244000$.

Оценка сложности построенной схемы для умножения многочленов над полем $GF(2^8)$ равна $3 \cdot 210850 + 255 \cdot 106 = 659580$. Заметим, что два раза преобразование Фурье применяется к векторам, наполовину состоящим из нулей. При применении тривиального алгоритма для его выполнения, или алгоритма Блюстейна, в этом случае сложность уменьшилась бы в два раза. Однако при применении указанного быстрого алгоритма этот факт не имеет места.

Убедимся, что даже для такого большого поля метод Карацубы все еще лучше. Оценим сложность умножения многочленов степени 2^7-1 над полем $GF(2^8)$ методом Карацубы. Оцениваем отдельно число умножений и число сложений в этом поле. Ранее было получено, что $M(8) \leq 27, A(8) \leq 100$, откуда

$$M(16) \leq 81, A(16) \leq 360, M(32) \leq 243, A(32) \leq 1204,$$

$$M(64) \leq 729, A(64) \leq 3864, M(128) \leq 2187, A(128) \leq 12100,$$

$$M(256) \le 6561, A(256) \le 37320, M(512) \le 19683, A(512) \le 114004,$$

$$M(1024) \le 59049, A(1024) \le 346104.$$

Поэтому сложность умножения многочленов степени $2^7 - 1$ над полем $GF(2^8)$ методом Карацубы равна $2187 \cdot 106 + 12100 \cdot 8 = 328622$. Как видим, она вдвое меньше. Для сравнения укажем, что схема, основанная на стандартном алгоритме умножения многочленов степени $2^7 - 1$, имеет сложность почти 2000000.

Но оценка сложности умножения многочленов степени $2^7 - 1$ над полем $GF(2^8)$ по модулю произвольного неприводимого многочлена степени 2^7 может быть в три раза больше, чем просто сложность умножения многочленов. Реально такого увеличения сложности может и не быть, если удастся найти неприводимый многочлен $p(x)$ над полем $GF(2^8)$ с малым числом ненулевых и неединичных коэффициентов. Тогда схема приведения произвольного многочлена степени $2^8 - 2$ по модулю $p(x)$ может быть построена с малой сложностью с помощью школьного алгоритма деления с остатком.

Рассмотрим еще вариант построения схемы, основанный на умножении многочленов степени $2^{10} - 1$ по модулю многочлена $p(x) = x^{1024} + x^{19} + x^9 + x^6 + 1$ (неприводимых трехчленов такой степени не существует). В этом случае схема приведения произвольного многочлена степени $2^{11} - 2$ по модулю $p(x)$ может быть построена с сложностью $\le 1023 \cdot 4 = 4092$ с помощью школьного алгоритма деления с остатком. Сложность же умножения методом Карацубы $\le 346104 + 59049 = 405153$. Возможно, эта схема окажется лучшей, чем указанная выше. Но ее глубина будет чрезвычайно большой.

Заметим, что схема умножения из [141] в случае поля $GF(2^{10})$ имеет сложность 377675. Но она работает не в стандартном, а в некотором специальном базисе. Схема для инвертирования из [141] имеет сложность 567559.

Применяя указанный выше метод, можно построить схему умножения в поле $GF(2^{2^{17}})$ сложности не более

$$3(255M(GF(2^{1024})) + 3FA(255) \cdot 1024 + 3FM(255) \cdot M(GF(2^8)) \cdot 2^7) =$$

$$= 3(255 \cdot 3 \cdot 328622 + 3 \cdot 8022 \cdot 1024 + 3 \cdot 1759 \cdot 106 \cdot 128 <$$

$$< 105 \cdot 10^7.$$

Эта схема все еще возможно чуть хуже схемы [141], у которой сложность примерно равна $8.4 \cdot 10^8$. Но у схемы для поля $GF(2^{2^{24}}$ сложность возрастет не более чем в 800 раз по сравнению со сложностью схемы умножения для поля $GF(2^{2^{17}}$, а в схеме [141] она возрастает в $3^7 = 2187$ раз при увеличении размерности в 2^7 раз. Далее указанная схема будет все больше и больше обгонять как схему [141], так и схему, основанную на методе Карацубы. Но, разумеется, практического значения схемы таких больших размеров не имеют.

4.2.7 Примеры построения схем для полей $GF(2^{2\cdot 3^k})$.

Положим $q_n = 2^{2\cdot 3^n}$ и рассмотрим башню полей

$$GF(q_2) \subset GF(q_3) \subset \ldots \subset GF(q_n).$$

В каждом из кубических расширений

$$GF(q_{i-1}) \subset GF(q_i)$$

выберем базис $\{1, \alpha_i, \alpha_i^2\}$ соответствующий неприводимому двучлену $x^3 + \omega_i$, где ω_i любой из примитивных элементов поля $GF(q_{i-1})$. Неприводимость этого двучлена следует из одной общей теоремы, которую можно найти в [45], но в данном случае ее легко проверить непосредственно. Действительно, этот двучлен неприводим над полем $GF(q_{i-1})$, так как в нем он не имеет корней, в противном случае возводя его корень в степень $(q_{i-1}-1)/3$, получим равенство

$$1 = x^{q_{i-1}-1} = (x^3)^{(q_{i-1}-1)/3} = \omega^{(q_{i-1}-1)/3},$$

противоречащее определению примитивного элемента.

Для умножения двух квадратных трехчленов на полем $GF(q_{i-1})$ применяем метод Карацубы в виде:

$$(a_0 + a_1x + a_2x^2)(b_0 + b_1x + b_2x^2) = c_0 + c_1x + c_2x^2 + c_3x^3 + c_4,$$

$$c_0 = a_0b_0, c_4 = a_2b_2, c_1 = (a_0 + a_1)(b_0 + b_1) + (a_0b_0 + a_1b_1),$$
$$c_2 = (a_0b_0 + a_1b_1) + (a_0 + a_2)(b_0 + b_2) + a_2b_2,$$
$$c_3 = (a_1 + a_2)(b_1 + b_2) + a_1b_1 + a_2b_2.$$

При этом используется 6 умножений и 12 сложений. Операция умножения в рассматриваемом расширении сводится к рассмотренному выше умножению трехчленов и приведению полученного в результате многочлена четвертой степени по модулю многочлена $x^3 + \omega_i$. Последняя операция требует двух умножений и двух сложений. Полагая $n_i = 2 \cdot 3^i, M(GF(q_i)) = M(n_i)$ отсюда имеем

$$M(n_i) \leq 8M(n_{i-1}) + 14n_{i-1}.$$

Для оценки $M(n_2) = M(18)$ можно воспользоваться указанным выше методом умножения в квадратичных расширениях, а в поле $GF(2^9)$ выбрать существующий в нем оптимальный нормальный базис. Тогда можно построить схемы с оценками

$$M(9) \leq 189, I(9) \leq 2M(9)+2M(3) = 414, M(18) \leq 603, I(18) \leq I(9)+3M(9)+9 = 990.$$

Отсюда следуют оценки

$$M(54) \leq 8M(18) + 14 \cdot 18 = 5076, M(162) \leq 8M(54) + 14 \cdot 54 = 41364,$$

$$M(486) \le 8M(162) + 14 \cdot 162 = 333180,$$

и в общем случае по индукции имеем

$$M(n_k) \le 8^{k-2}M(18) + 28(3^{k-1} + 8 \cdot 3^{k-2} + \ldots + 8^{k-3} \cdot 3^2) <$$

$$603 \cdot 8^{k-2} + \frac{28 \cdot 9}{5}8^{k-2} = 653.4 \cdot 8^{k-2} < 2.75n_k^{\log_3 8} < 2.75n_k^{1.9}.$$

Для инвертирования в поле $GF(q_i)$ пользуемся формулой

$$x^{-1} = K(x) \cdot N(x)^{-1}, K(x) = x^{q_{i-1}}x^{q_{i-1}^2}, N(x) = xK(x).$$

Так как

$$N(x)^{q_{i-1}} = x^{q_{i-1}}x^{q_{i-1}^2}x^{q_{i-1}^3} =$$

$$x^{q_{i-1}}x^{q_{i-1}^2}x^{q_i} = x^{q_{i-1}}x^{q_{i-1}^2}x = N(x),$$

то $N(x) \in GF(q_{i-1})$, поэтому для инвертирования нужно выполнить возведение в степени q_{i-1}, q_{i-1}^2 потом два раза выполнить умножение в поле $GF(q_i)$, потом выполнить инвертирование в подполе $GF(q_{i-1})$, и еще три умножения в этом же подполе. Так как $q_{i-1} - 1$ кратно трем, то

$$\alpha_i^{q_{i-1}} = \alpha_i(\omega_i)^{(q_{i-1}-1)/3} = a_i\alpha_i\alpha_i^{2q_{i-1}} = \alpha_i^2(\omega_i)^{2(q_{i-1}-1)/3} = b_i\alpha_i^2, a_i, b_i \in GF(q_{i-1}),$$

$$\alpha_i^{q_{i-1}^2} = \alpha_i(\omega_i)^{(q_{i-1}^2-1)/3} = c_i\alpha_i\alpha_i^{2q_{i-1}^2} = \alpha_i^2(\omega_i)^{2(q_{i-1}^2-1)/3} = d_i\alpha_i^2, c_i, d_i \in GF(q_{i-1}).$$

Поэтому возведение в степень q_{i-1} произвольного $x = x_0 + x_1\alpha_i + x_2\alpha_i^2, x_i \in GF(q_{i-1})$ выполняется по формуле

$$x^{q_{i-1}} = x_0^{q_{i-1}} + x_1^{q_{i-1}}\alpha_i^{q_{i-1}} + x_2^{q_{i-1}}\alpha_i^{2q_{i-1}} = x_0 + x_1a_i\alpha_i + x_2b_i\alpha_i^2,$$

и сводится к двум умножениям в подполе $GF(q_{i-1})$. Аналогично, возведение в степень q_{i-1}^2 сводится к двум умножениям в подполе $GF(q_{i-1})$. Поэтому для сложности инвертирования имеем рекуррентную оценку

$$I(n_i) \le I(n_{i-1}) + 2M(n_i) + 7M(n_{i-1})$$

Из нее следует оценка

$$I(54) \le I(18) + 2M(54) + 7M(18) = 15363.$$

Ее можно улучшить, если вспомнить, что в поле $GF(2^{18})$ мы выбрали базис, являющийся произведением оптимальных нормальных базисов порядков 2 и 9, для которого

$$M(18) \le 603, I(18) \le 990.$$

В этом базисе возведение в квадрат бесплатное, поэтому можно воспользоваться справедливой для любого такого базиса оценкой

$$I(n_i) \le I(n_{i-1}) + 2M(n_i) + 3M(n_{i-1}).$$

Тогда имеем

$$I(54) \le I(18) + 2M(54) + 3M(18) = 12951.$$

Далее,

$$I(162) \le I(54)+2M(162)+7M(54) = 131211, I(486) \le I(162)+2M(486)+7M(162) = 1087119,$$

и в общем случае по индукции имеем

$$I(n_k) < 2 \cdot 653.4 \cdot (9/8)8^{k-2} + 7 \cdot 653.4 \cdot (9/8)8^{k-3} + 990 <$$

$$< 33.03 \cdot 8^k + 990 < 8.9 n_k^{\log_3 8} + 990 < 9 n_k^{1.9}.$$

Оценку

$$I(486) \le 1087119$$

можно улучшить, если взять в качества базиса индукции поле $GF(2^{162})$. Действительно, оценку

$$M(162) \le 41364$$

можно улучшить, если воспользоваться существующим в поле $GF(2^{162})$ оптимальным нормальным базисом первого типа. Применим доказанное ранее для таких базисов неравенство

$$M(n) \le 4n - 4 + m(n),$$

где $m(n)$ — сложность умножения многочленов степени $n-1$. Применяя для оценки $m(n)$ рекуррентные оценки метода Карацубы, имеем

$$m(3) = 3^2 + 2^2 = 13, m(5) = 4^2 + 5^2 = 41, m(6) = 3m(3) + 4(6-1) = 59,$$

$$m(10) = 3m(5) + 36 = 159, m(11) = 2m(6) + m(5) + 38 = 197,$$

$$m(20) = 3m(10) + 76 = 553, m(21) = 2m(11) + m(10) + 78 = 631,$$

$$m(40) = 3m(20) + 156 = 1815, m(41) = 2m(21) + m(20) + 158 = 1973,$$

$$m(81) = 2m(41) + m(40) + 318 = 6079, m(162) = 3m(81) + 644 = 18881.$$

В результате

$$M(162) \le 644 + m(162) = 19525.$$

Тогда на следующем этаже башни

$$M(486) \le 8M(162) + 14 \cdot 162 = 158468,$$

что в два раза лучше полученного ранее, и, пользуясь тем, что в нормальном базисе поле $GF(2^{162})$ возведение в квадрат делается бесплатно, можно применить оценку сложности инвертирования

$$I(n_i) \le I(n_{i-1}) + 2M(n_i) + 3M(n_{i-1}),$$

откуда имеем

$$I(486) \le I(162) + 2M(486) + 3M(162) = I(162) + 375511.$$

Однако мы не можем здесь воспользоваться старой оценкой для $I(162)$, так как она была получена для другого базиса. Грубую оценку, почти такую же как и полученную ранее, можно получить применяя оценку метода аддитивных цепочек для нормального базиса.

$$I(162) \le l(162)M(162),$$

где $l(162)$ —длина кратчайшей линейной аддитивной цепочки для числа 162. Применяя метод множителей, оцениваем $l(162)$ как $1 + l(81) = 3 + l(27) = \ldots = 7 + l(3) = 9$. Отсюда

$$I(162) \le 9M(162) = 175725.$$

Эту оценку можно уточнить, если вычислять инверсию по формуле

$$x^{-1} = K(x) = x^2 \ldots x^{2^{161}} = x^{2^{81}} y^2 y^4 y^8 \ldots y^{2^{80}}, y = x \cdot x^{2^{81}},$$

где $y^{2^{81}} = y$, значит $y \in GF(2^{81})$. В этом поле существует оптимальный нормальный базис второго типа, порождаемый элементом $\alpha = \zeta + \zeta^{-1} = \zeta + \zeta^{162}$, где $\zeta \in GF(2^{162}), \zeta^{163} = 1$ есть порождающий элемент оптимального нормального базиса первого типа в поле $GF(2^{162})$. Ранее отмечалось, что координаты произвольного элемента подполя $GF(2^{81})$ относительно оптимального базиса просто совпадают с половиной координат этого же элемента относительно оптимального базиса в поле $GF(2^{162})$. Поэтому для вычисления

$$x^{-1} = x^{2^{81}} y^2 y^4 y^8 \ldots y^{2^{80}}, y = x \cdot x^{2^{81}}$$

достаточно выполнить $l(80) = 7$ умножений в поле $GF(2^{81})$ и только два умножения в поле $GF(2^{162})$ (все возведения в степени — бесплатные). Значит,

$$I(162) \le 2M(162) + 7M(81).$$

Для оценки $M(81)$ воспользуемся методом перехода от оптимального к стандартному базису. Сложность этого перехода ранее была оценена как $(n/2)\log_2 n + 3n < 500$. Сложность обычного умножения многочленов степени 80 оценивается как $m(81) \le 6079$. Вычисляя минимальный многочлен рассматриваемого базиса, находим что он имеет не более 25 единичных коэффициентов. Поэтому сложность приведения многочлена степени 160 по модулю этого минимального многочлена оценивается как $81 \cdot 25 = 2025$. Окончательно имеем $M(81) \le 6079 + 2025 + 1500 = 9604$. Поэтому

$$I(162) \le 2M(162) + 7M(81) \le 106278.$$

Эту оценку можно и далее несколько уточнить, но мы не будем здесь этим заниматься. Используя полученную оценку, имеем

$$I(486) \leq I(162) + 375511 = 481789.$$

Таким образом и для полей размерностей 162 и 486 реально на одном чипе разместить одновременно схемы для всех арифметических операций в этих полях.

Для $n_k = 4 \cdot 3^k$ можно получить асимптотически более точные оценки $I(n_k) = O(n_k^{\log_3 7}, M(n_k) = O(n_k^{\log_3 7}$, если для умножения двух квадратных трехчленов на полем $GF(q_{i-1})$ воспользоваться методом 4.2.5. Опишем кратко этот метод в рассматриваемом случае. Для умножения трехчленов f, g вычислим вначале значения $f(\omega^i), g(\omega^i), i = 0, \ldots, 4$, где $\omega^5 = 1, \omega \in GF(2^4)$, т.е. вычислим два раза ДПФ пятого порядка, примененное к укороченным векторам. Тривиальным образом для этого нужно 16 умножений на скаляры из подполя $GF(2^4) \subset GF(q_{i-1})$ и 20 сложений в поле $GF(q_{i-1})$. Если применить алгоритм Блюстейна, то число умножений уменьшится до 12, и из них 4 будут умножениями на скаляры из подполя $GF(2^2)$. Потом с помощью 5 умножений в поле $GF(q_{i-1})$ вычислим $h(\omega^i) = f(\omega^i)g(\omega^i), i = 0, \ldots, 4$. Для того чтобы найти коэффициенты многочлена $h(x) = f(x)g(x)$ применяем обратное ДПФ пятого порядка к вектору с компонентами $h(\omega^i), i = 0, \ldots, 4$. Тривиальным образом для этого нужно 16 умножений на скаляры из подполя $GF(2^4) \subset GF(q_{i-1})$ и 20 сложений в поле $GF(q_{i-1})$. Как уже отмечалось в предыдущей секции, если применить алгоритм Блюстейна, то число умножений уменьшится до 10, причем 6 из них будут умножениями на скаляры из подполя $GF(2^2)$. После того, как вычислено произведение $h = fg$, остается привести его по модулю $x^3 + \omega_i$, выполнив 2 умножения и 2 сложения в поле $GF(q_{i-1})$. В результате имеем

$$M(n_i) \leq 7M(n_{i-1}) + 10M(2)n_{i-1}/2 + 12M(4)n_{i-1}/4 + 42n_{i-1},$$

где $M(2) = 7$ есть сложность умножения в поле $GF(2^2)$, $M(4) = 29$ есть сложность умножения в поле $GF(2^4)$ при выборе базиса, являющегося произведением нормального базиса

$$\{\alpha_1, \alpha_1^2\}, \alpha_1^2 + \alpha_1 = 1,$$

и базиса

$$\{1, \alpha_2, \}\alpha_2^2 + \alpha_2 = \alpha_1.$$

Следовательно

$$M(n_i) \leq 7M(n_{i-1}) + 164n_{i-1}.$$

Для инвертирования в поле $GF(q_i)$ остается справедливой рекуррентная оценка

$$I(n_i) \leq I(n_{i-1}) + 2M(n_i) + 15M(n_{i-1}) + 8n_{i-1}.$$

Приведем пример применения полученных оценок. Выше было указано на существование схем сложности

$$M(9) \leq 189, I(9) \leq 2M(9) + 2M(3) = 414.$$

Заметим без доказательства, что для любого нечетного n можно построить схемы сложности

$$M(4n) \leq 9M(n) + 20n, I(4n) \leq I(n) + 14M(n) + 16n.$$

Поэтому существуют схемы сложности

$$M(36) \leq 1881, I(36) \leq 3204.$$

Применяя указанные выше оценки, получаем что существуют схемы умножения и инвертирования сложности

$$M(108) \leq 7M(36) + 164 \cdot 36 \leq 19071,$$

$$I(108) \leq I(36) + 2M(108) + 15M(36) + 8 \cdot 36 = 69649,$$

$$M(324) \leq 7M(108) + 164 \cdot 108 \leq 151209,$$

$$I(324) \leq I(108) + 2M(324) + 15M(108) + 8 \cdot 108 = 659196.$$

Аналогичным образом можно для $n_k = 8 \cdot 3^k$ получить асимптотически более точные оценки $I(n_k) = O(n_k^{\log_3 5}, M(n_k) = O(n_k^{\log_3 5}$, если рассмотреть башню полей $GF(q_{i-1}) \subset GF(q_i)$, где $q_i = 2^{24 \cdot 9^i}$ или $q_i = 2^{72 \cdot 9^i}$, в которой выбираются на каждом этаже базисы, соответствующие неприводимому многочлену $x^9 + \omega_i$, где ω_i есть произвольный примитивный элемент поля $GF(q_{i-1})$. Неприводимость этого многочлена следует из теоремы 3.75 [45], так как $q_i - 1$ кратно $2^6 - 1$, а значит и 9. Тогда умножение многочленов 8-й степени по модулю $x^9 + \omega_i$ будет выполняться за $17+8 = 25$ умножений в поле $GF(q_{i-1})$ с помощью трехкратного применения ДПФ порядка 17 над подполем $GF(2^8)$.

Для уменьшения мультипликативной константы в этой оценке можно выполнить ДПФ с помощью алгоритма Блюстейна, который вычисление значений $f(\omega^i), i = 0, \ldots, 16$, где $\omega \in GF(2^8), \omega^{17} = 1$ сводит к делению многочлена $f(x)$ с остатком на многочлены

$$p_i = (x + \omega^i)(x + \omega^{17-i}) = x^2 + \beta_i + 1, i = 1, \ldots, 8,$$

где $\beta_i \in GF(2^4)$ так как

$$(\beta_1)^{16} = \beta_1, \beta_2 = \beta_1^2, \beta_4 = \beta_1^4, \beta_8 = \beta_1^8, \beta_3 = \beta_2\beta_1 + \beta_1,$$

$$\beta_6 = \beta_3^2, \beta_5 = \beta_6^2, \beta_7 = \beta_5^2.$$

Можно проверить, что $\beta_i^3 \neq 1, 0, i = 1, \ldots, 8$ значит все эти элементы различны и не принадлежат полю $GF(2^2)$. Деление на многочлены p_i можно свести к делению на многочлены

$$g_1 = p_1p_4 = x^4 + \gamma_1 x^3 + \beta_1^5 x^2 + \gamma_1 x + 1, \gamma_1 = \beta_1 + \beta_1^4 = \beta_1 + \beta_4, g_2 = p_2p_8 = x^4 + \gamma_2 x^3 + \beta_2^5 x^2 + \gamma_2 x + 1, \gamma_2 =$$

$g_3 = p_3p_5 = x^4+\gamma_3x^3+\beta_3^5x^2+\gamma_3x+1, \gamma_3 = \beta_3+\beta_3^4 = \beta_3+\beta_5, g_4 = p_6p_7 = x^4+\gamma_4x^3+\beta_6^5x^2+\gamma_4x+1, \gamma_4 = \beta_6+\beta_6^4 =$
где $\gamma_i \neq 0, \gamma_i^4 = \gamma_i, i = 1,2,3,4$, поэтому $\gamma_i \in GF(2^2), i = 1,2,3,4$. Кроме того, и остальные коэффициенты $\beta_i^5 \in GF(2^2)$, так как $(\beta_i^5)^3 = 1$. Далее, деление на многочлены p_i можно свести к делению на многочлен

$$g_1g_2 = x^8 + \delta_1x^7 + \delta_2x^6 + \delta_3x^5 + \delta_4x^4 + \delta_3x^3 + \delta_2x^6 + \delta_1x^7 + 1,$$

где $\delta_1 = \gamma_1 + \gamma_2 = \gamma_1 + \gamma_1^2, \delta_1^2 = \delta_1 \in GF(2)$,

$\delta_2 = \gamma_1\gamma_2+\beta_2^5+\beta_1^5 = \gamma_1^3+\beta_2^5+\beta_1^5 = 1+\beta_3+\beta_5+\beta_6+\beta_7 = 1+\gamma_3+\gamma_4 = 1+\gamma_3+\gamma_3^2 \in GF(2)$,

$$\delta_3 = \gamma_3 + \gamma_4 + \gamma_3\beta_6^5 + \gamma_4\beta_3^5 \in GF(2)$$

так как

$$\gamma_3\beta_6^5 = (\beta_3 + \beta_3^4)\beta_3^{10} = \beta_3^{11} + \beta_3^{14} == \beta_3^{-4} + \beta_3^{-1} \in GF(2^2)$$

и аналогично $\gamma_4\beta_3^5 =\in GF(2^2)$,

$$\delta_4 = \beta_6^5\beta_3^5 = \beta_3^{15} = 1.$$

Круговой многочлен $1+x+\ldots+x^{16}$ согласно теории должен разлагаться на на два неприводимых множителя степени 8 над полем $GF(2)$. Один из них — это g_1g_2, а второй очевидно g_3g_4. В таблице неприводимых многочленов степени 8 только два возвратных: $1+x^3+x^4+x^5+x^8$ и $1+x+x^2+x^4+x^6+x^7+x^8$. Поэтому один из них равен g_1g_2, а другой равен g_3g_4. Можно подсчитать, что двукратное вычисление прямого ДПФ и однократное вычисление обратного ДПФ с помощью указанного алгоритма требует 470 операций сложения в поле $GF(q_{i-1})$, 48 умножений на скаляры из поля $GF(2^2)$, 144 умножений на скаляры из поля $GF(2^4)$ и 48 умножений на скаляры из поля $GF(2^8)$. После этого выполнение умножения двух многочленов степени 8 по модулю $x^9+\omega_i$ требует еще 25 умножений в поле $GF(q_{i-1})$ и еще 8 сложений. Отсюда имеем оценку

$$M(n_i) \leq 25M(n_{i-1}) + 478n_{i-1} + 48M(2)n_{i-1}/2 + 48M(8)n_{i-1}/8 + 48M(4)n_{i-1}/4.$$

Выбирая в $GF(2^8)$ подходящий базис, имеем $M(2) = 7, M(4) = 29, M(8) = 106$, откуда

$$M(n_i) \leq 25M(n_{i-1}) + 1630n_{i-1}.$$

Можно проверить, что

$$M(24) \leq 882, I(24) \leq 1110.$$

Тогда

$$M(216) \leq 25M(24) + 1630 \cdot 24 = 61170.$$

Двукратное применение предыдущей конструкции дает оценки

$$M(72) \leq 7M(24) + 164 \cdot 24 \leq 10110,$$

$$M(216) \le 7M(72) + 164 \cdot 72 \le 82578.$$

Уточним теперь оценку для сложности инвертирования. В расширении $GF(q_i)$ поля $GF(q_{i-1})$ выполняем инвертирование по формуле

$$x^{-1} = K(x)N(x)^{-1}, K(x) = x^{q_{i-1}}x^{q_{i-1}^2}\dots x^{q_{i-1}^8}, N(x) = xK(x).$$

Так как

$$N(x)^{q_{i-1}} = x^{q_{i-1}}x^{q_{i-1}^2}\dots x^{q_{i-1}^9} =$$

$$= x^{q_{i-1}}x^{q_{i-1}^2}\dots x^{q_{i-1}^8}x = N(x),$$

то $N(x) \in GF(q_{i-1})$, поэтому для инвертирования нужно вычислить $K(x), N(x)$, потом выполнить инвертирование в подполе $GF(q_{i-1})$, и 9 раз выполнить умножение в поле $GF(q_{i-1})$. Для вычисления $N(x), K(x)$ сначала вычислим $y = xx^{q_{i-1}^3}x^{q_{i-1}^6}$, а потом

$$N(x) = yy^{q_{i-1}}y^{q_{i-1}^2}, K(x) = y^{q_{i-1}}y^{q_{i-1}^2}x^{q_{i-1}^3}x^{q_{i-1}^6}.$$

Так как

$$y^{q_{i-1}^3} = x^{q_{i-1}^3}x^{q_{i-1}^6}x^{q_{i-1}^9} = xx^{q_{i-1}^3}x^{q_{i-1}^6} = y,$$

то $y \in GF(q_{i-1}^3)$. Значит, для вычисления $N(x), K(x)$ нужно сделать 2 умножения в поле $GF(q_i)$, 2 умножения в поле $GF(q_{i-1}^3)$ и одно умножение в поле $GF(q_i)$ на элемент подполя $GF(q_{i-1}^3)$, кроме операций возведения в степени вида q^l, две из которых делаются в поле $GF(q_i)$, а две — в поле $GF(q_{i-1}^3)$. Поле $GF(q_{i-1}^3)$ является кубическим расширением подполя $GF(q_{i-1})$, и в нем можно выбрать базис $\{1, \beta_i, \beta_i^2\}$, где $\beta_i^3 = \omega_i$ — примитивному элементу подполя $GF(q_{i-1})$. Умножение в этом базисе совпадает с умножением квадратных трехчленов по модулю неприводимого над полем $GF(q_{i-1})$ многочлена $x^3 + \omega_i$. Напомним, что в расширении девятой степени $GF(q_{i-1}) \subset GF(q_i)$ ранее был выбран базис $\{1, \alpha_i, \dots, \alpha_i^8\}$, где $\alpha_i^9 = \omega_i$. Поэтому можно выбрать $\beta_i = \alpha_i^3$. Тогда произвольный элемент подполя $GF(q_{i-1}^3)$ имеет относительно базиса $\{1, \beta_i, \beta_i^2\}$, координаты, которые совпадают с тремя координатами этого элемента относительно базиса $\{1, \alpha_i, \dots, \alpha_i^8\}$ (а остальные его координаты в этом базисе равны нулю). Поэтому сложность умножения элементов этого подполя оценивается как и выше неравенством

$$M(3n_{i-1}) \le 8M(n_{i-1}) + 14n_{i-1}$$

или неравенством

$$M(3n_{i-1}) \le 7M(n_{i-1}) + 164n_{i-1}.$$

Так как произвольный элемент поля $GF(q_i)$ можно представить в виде

$$X_0 + X_1\alpha_i + X_2\alpha_i^2,$$

где $X_j = x_j + x_{3+j}\alpha_i^3 + x_{6+j}\alpha_i^6 \in GF(q_{i-1}^3), j = 0,1,2$ то умножение в поле $GF(q_i)$ на элемент подполя $GF(q_{i-1}^3)$ сводится к трем умножениям в этом подполе. Поэтому суммарная сложность всех используемых при инвертировании умножений равна

$$2M(n_i) + 5M(3n_{i-1}) + 9M(n_{i-1}).$$

Возведение в степень вида $q_{i-1}^l, l < 3$ в подполе $GF(q_{i-1}^3)$ выполняется, как было показано выше, со сложностью $2M(n_{i-1})$, а делается оно двукратно. Оценим сложность возведения в степени q_{i-1}^3, q_{i-1}^6 в поле $GF(q_i)$. Так как произвольный элемент x поля $GF(q_i)$ можно представить в виде

$$X_0 + X_1\alpha_i + X_2\alpha_i^2,$$

где $X_j \in GF(q_{i-1}^3), j = 0,1,2$, то

$$x^{q_{i-1}^3} = X_0^{q_{i-1}^3} + X_1^{q_{i-1}^3}\alpha_i^{q_{i-1}^3} + X_2^{q_{i-1}^3}\alpha_i^{2\cdot q_{i-1}^3} = X_0 + X_1\alpha_i^{q_{i-1}^3} + X_2\alpha_i^{2\cdot q_{i-1}^3}.$$

Так как

$$\alpha_i^{q_{i-1}^3} = \alpha_i(\omega_i)^{(q_{i-1}^3-1)/9} = a_i\alpha_i, \alpha_i^{2\cdot q_{i-1}^3} = \alpha_i^2(\omega_i)^{2(q_{i-1}^3-1)/9} = b_i\alpha_i^2, a_i, b_i \in GF(q_{i-1}),$$

$$\alpha_i^{q_{i-1}^6} = \alpha_i(\omega_i)^{(q_{i-1}^6-1)/9} = c_i\alpha_i, \alpha_i^{2\cdot q_{i-1}^6} = \alpha_i^2(\omega_i)^{2(q_{i-1}^6-1)/9} = d_i\alpha_i^2, c_i, d_i \in GF(q_{i-1}),$$

то

$$x^{q_{i-1}^3} = X_0 + X_1\alpha_i^{q_{i-1}^3} + X_2\alpha_i^{2\cdot q_{i-1}^3} = X_0 + X_1a_i\alpha_i + X_2b_i\alpha_i,$$

значит возведение в степень q_{i-1}^3 в поле $GF(q_i)$ сводится к двум умножениям в подполе $GF(q_{i-1}^3)$ на элементы подполя $GF(q_{i-1})$. Значит, его сложность оценивается как $6M(n_{i-1})$. Аналогично оценивается сложность возведения в степень q_{i-1}^6. Отсюда следует оценка сложности инвертирования

$$I(n_i) \le I(n_{i-1}) + 2M(n_i) + 5M(3n_{i-1}) + 25M(n_{i-1}),$$

откуда имеем оценки

$$I(n_i) \le I(n_{i-1}) + 2M(n_i) + 65M(n_{i-1}) + 70n_{i-1},$$

или

$$I(n_i) \le I(n_{i-1}) + 2M(n_i) + 60M(n_{i-1}) + 820n_{i-1},$$

и окончательно

$$I(n_i) \le I(n_{i-1}) + 115M(n_{i-1}) + 1700n_{i-1},$$

или

$$I(n_i) \le I(n_{i-1}) + 110M(n_{i-1}) + 2450n_{i-1}.$$

В частности,

$$I(216) \le I(24) + 2M(216) + 5M(72) + 25M(24) = 196050.$$

4.2.8 Еще о построении схем для полей $GF(2^{2\cdot 3^k})$.

Используя ДПФ, приведем пример последовательности полей $GF(2^N)$, для которых можно построить схемы инвертирования и умножения со сложностью

$$M(GF(2^N)) = N(\log_3 N)^{(\log_2 \log_3 N)/2+O(1)}, I(N) = O(M(GF(2^N))).$$

Положим $q_i = 2^{a_i}, a_i = 2 \cdot 3^{b_i}, b_i = 2^i$ и рассмотрим башню полей

$$GF(q_0) \subset GF(q_1) \subset \ldots \subset GF(q_k).$$

В отличие от ранее рассматривавшихся башен в этой степени расширений в ее этажах растут в сверхгеометрической прогрессии, а именно как 3^{2^i}, причем основание i-го этажа башни является полем размерности $a_i = 2 \cdot 3^{2^i}$.

Упражнение 4.2.1 Докажите, что $2^{2\cdot 3^k} - 1$ кратно 3^{k+1}.

Так как $q_i - 1 = 2^{a_i} - 1$ кратно $3^{b_i+1} = 3n_i$, то в поле $GF(q_i)$ найдется элемент порядка $3^{b_i+1} = 3n_i$, и, значит, определено ДПФ порядка $3^{b_i+1} = 3n_i$. Ранее было показано, что многочлены степени меньше $n_i = 3^{b_i} = a_i/2$ над полем $GF(q_i)$ могут быть перемножены с помощью $24n_i \log_3 n_i + O(n_i)$ мультипликативных и $68n_i \log_3 n_i + O(n_i)$ аддитивных операций в этом поле. Если обозначить сложность умножения в поле $GF(q_i)$ через $M(GF(q_i))$, то сложность умножения многочленов степени меньше n_i над полем $GF(q_i)$ будет оцениваться как

$$M(a_{i+1}) = (24n_i \log_3 n_i + O(n_i))M(GF(q_i)) + (68n_i \log_3 n_i + O(n_i))n_i.$$

Выберем в этом поле произвольный примитивный элемент α_i, тогда многочлен $x^{n_i} - \alpha_i$ будет над ним неприводимым согласно , теореме 3.75 [45], так как $n_i = 3^{b_i}$ делит $q_i - 1$, а значит и $q_{i+1} - 1 = 2^{a_{i+1}} - 1$. Выбирая в расширении $GF(q_{i+1})$ поля $GF(q_i)$ стандартный базис, соответствующий этому многочлену (степень расширения равна $a_{i+1}/a_i = n_i$), получаем что

$$M(GF(q_{i+1})) \leq M(a_{i+1}) + n_i M(GF(q_i)) + a_{i+1} \leq$$

$$\leq (24n_i \log_3 n_i + O(n_i))M(GF(q_i)) + (68n_i \log_3 n_i + O(n_i))n_i + a_{i+1} \leq$$

$$\leq (12a_i \log_3 a_i + O(a_i))M(GF(q_i)) + (17a_i^2 \log_3 a_i + O(a_i^2) + ai + 1 \leq$$

$$\leq (12a_i \log_3 a_i + O(a_i))M(GF(q_i)) + (17a_i^2 \log_3 a_i + O(a_i^2) \leq$$

$$\leq (12a_i b_i + O(a_i))M(GF(q_i)).$$

Отсюда по индукции следует, что

$$\log_2 M(GF(q_n)) \leq \sum_{i=1}^{n-1} \log_2 12a_i b_i + O(1) = \sum_{i=1}^{n-1}((\log_2 3)2^i + i \log_2 24) + O(1) \leq$$

$$(\log_2 3)2^n + n^2/2 + (2\frac{1}{2} + \log_2 3)n + O(1),$$

значит

$$M(GF(q_n)) \le O\left(3^{2^n+n}2^{(n^2+5n)/2}\right), q_n = 2^{2\cdot 3^{2^n}}.$$

Обозначая для краткости $a_n = 2\cdot 3^{2^n}$ через N, имеем

$$M(GF(2^N)) \le N(\log_3 N)^{n/2+O(1)} = N(\log_3 N)^{(\log_2\log_3 N)/2+O(1)}.$$

Получим теперь оценку для сложности инвертирования. В расширении $GF(q_{i+1})$ поля $GF(q_i)$ выполняем инвертирование по формуле

$$x^{-1} = K(x)N(x)^{-1}, K(x) = x^{q_i}x^{q_i^2}\dots x^{q_i^{n_i-1}}, N(x) = xK(x).$$

Так как

$$N(x)^{q_i} = x^{q_i}x^{q_i^2}\dots x^{q_i^{n_i}} =$$
$$= x^{q_i}x^{q_i^2}\dots x^{q_i^{n_i-1}}x = N(x),$$

то $N(x) \in GF(q_i)$, поэтому для инвертирования нужно вычислить $K(x), N(x)$, потом выполнить инвертирование в подполе $GF(q_i)$, и n_i раз выполнить умножение в поле $GF(q_i)$.

Для вычисления $N(x), K(x)$ сначала найдем $y = xx^{q_i^{n_i/3}}x^{q_i^{2n_i/3}}$, а потом

$$N(x) = yy^{q_i}\dots y^{q_i^{n_i/3-1}}, K(x) = y^{q_i}\dots y^{q_i^{n_i/3-1}}x^{q_i^{n_i/3}}x^{q_i^{2n_i/3}}.$$

Так как

$$y^{q_i^{n_i/3}} = x^{q_i^{n_i/3}}x^{q_i^{2n_i/3}}x^{q_i^{n_i}} = xx^{q_i^{n_i/3}}x^{q_i^{2n_i/3}} = y,$$

то $y \in GF(q_i^{n_i/3})$. Значит, для вычисления y нужно сделать 2 умножения в поле $GF(q_{i+1})$, и две операции возведения в степени $q_i^{n_i/3}, q_i^{2n_i/3}$ в том же поле, потом нужно вычислить

$$N(x) = yy^{q_i}\dots y^{q_i^{n_i/3-1}},$$

и для вычисления $K(x)$ нужно сделать одно умножение в поле $GF(q_{i+1})$ на элемент подполя $GF(q_i^{n_i/3})$. Так как произвольный элемент поля $GF(q_{i+1})$ можно представить в виде

$$X_0 + X_1\gamma_i + \dots + X_2\gamma_i^2,$$

где

$$X_j = x_j + x_3\gamma_i^{3+j} + x_{3(n_i/3-1)+j}\alpha_i^{3(n_i/3-1)} \in GF(q_i^{n_i/3}), j = 0,1,2$$

то умножение в поле $GF(q_{i+1})$ на элемент подполя $GF(q_i^{n_i/3})$ сводится к трем умножениям в этом подполе.

Поле $GF(q_i^{n_i/3})$ является расширением степени $n_i/3 = 3^{b_i-1}$ подполя $GF(q_i)$, и в нем можно выбрать базис $\{1, \beta_i, \dots, \beta_i^{n_i/3-1}\}$, где $\beta_i^{n_i/3}$ равен α_i — ранее выбранному примитивному элементу подполя $GF(q_i)$. Умножение в этом базисе совпадает с умножением многочленов степени $n_i/3$ по модулю неприводимого над полем $GF(q_i)$ многочлена $x^{n_i/3} + \alpha_i$. Напомним, что в расширении $GF(q_i) \subset GF(q_{i+1})$ ранее фактически был выбран базис $\{1, \gamma_i, \dots, \gamma_i^{n_i-1}\}$,

где $\gamma_i^{n_i} = \alpha_i$. Поэтому можно выбрать $\beta_i = \gamma_i^3$. Тогда произвольный элемент подполя $GF(q_i^{n_i/3})$ имеет относительно базиса $\{1, \beta_i, \ldots, \beta_i^{n_i/3-1}\}$ координаты, которые совпадают с $n_i/3$ координатами этого элемента относительно базиса $\{1, \alpha_i, \ldots, \alpha_i^{n_i-1}\}$ (а остальные его координаты в этом базисе равны нулю). Поэтому сложность умножения элементов этого подполя оценивается подобно тому, как это делалось выше, неравенством

$$M(GF(q_i^{n_i/3})) \le M(a_{i+1}/3) + n_i/3M(GF(q_i)) + a_{i+1}/3 \le$$

$$\le (8n_i \log_3 n_i + O(n_i))M(GF(q_i)) + ((68/3)n_i \log_3 n_i + O(n_i))n_i + a_{i+1}/3 \le$$

$$\le (4a_i b_i + O(a_i))M(GF(q_i)).$$

Оценим сложность возведения в степени $q_i^{n_i/3}, q_i^{2n_i/3}$ в поле $GF(q_{i+1})$. Так как произвольный элемент x поля $GF(q_{i+1})$ можно представить в виде

$$X_0 + X_1\gamma_i + X_2\gamma_i^2,$$

где $X_j \in GF(q_i^{n_i/3}), j = 0, 1, 2$, то

$$x^{q_i^{n_i/3}} = X_0^{q_i^{n_i/3}} + X_1^{q_i^{n_i/3}}\gamma_i^{q_i^{n_i/3}} + X_2^{q_i^{n_i/3}}\gamma_i^{2\cdot q_i^{n_i/3}} = X_0 + X_1\gamma_i^{q_i^{n_i/3}} + X_2\gamma_i^{2\cdot q_i^{n_i/3}}.$$

Так как $q_i^{n_i/3} - 1$ делится на $q_i - 1$, а значит кратно n_i, то

$$\gamma_i^{q_i^{n_i/3}} = \gamma_i(\alpha_i)^{(q_i^{n_i/3}-1)/n_i} = a_i\gamma_i, \gamma_i^{2\cdot q_{i-1}^3} = \gamma_i^2(\alpha_i)^{2(q_{i-1}^3-1)/9} = b_i\gamma_i^2, a_i, b_i \in GF(q_i),$$

поэтому

$$x^{q_i^{n_i/3}} = X_0 + X_1\gamma_i^{q_i^{n_i/3}} + X_2\gamma_i^{2\cdot q_i^{n_i/3}} = X_0 + X_1a_i\gamma_i + X_2b_i\gamma_i,$$

значит возведение в степень $q_i^{n_i/3}$ в поле $GF(q_{i+1})$ сводится к двум умножениям в подполе $GF(q_i^{n_i/3})$ на элементы подполя $GF(q_i)$. Значит, его сложность оценивается как $2n_i/3M(GF(q_i))$. Точно также оценивается сложность возведения в степень $q_i^{2n_i/3}$. Поэтому суммарная сложность всех выполненных операций равна

$$L_i = 2M(GF(q_{i+1})) + 3M(GF(q_i^{n_i/3})) + 4n_i/3M(GF(q_i)) =$$

$$\le (36a_i b_i + O(a_i))M(GF(q_i)).$$

Для вычисления

$$yy^{q_i} \ldots y^{q_i^{n_i/3-1}},$$

где $y \in GF(q_i^{n_i/3})$ применяем тот же прием, вычисляя сначала

$$z = yy^{q_i^{n_i/9}}y^{q_i^{2n_i/9}}.$$

Так как $y^{q_i^{n_i/3}} = y$, то $z^{q_i^{n_i/9}} = z$, значит $z \in GF(q_i^{n_i/9})$. Для вычисления z нужно выполнить два умножения в поле $GF(q_i^{n_i/3})$ и возведения в степени $q_i^{n_i/9}, q_i^{2n_i/9}$ в том же поле. Аналогично предыдущим рассуждениям, оцениваем их сложность как

$$2M(GF(q_i^{n_i/3})) + (4n_i/9)M(GF(q_i)) \leq$$
$$\leq (24a_{i-1}b_i + O(a_{i-1}))M(GF(q_i)).$$

Так как

$$yy^{q_i} \dots y^{q_i^{n_i/3-1}} = zz^{q_i} \dots z^{q_i^{n_i/9-1}},$$

то остается вычислить

$$zz^{q_i} \dots z^{q_i^{n_i/9-1}}, z \in GF(q_i^{n_i/9}).$$

Применяя тот же прием, сводим со сложностью

$$2M(GF(q_i^{n_i/9})) + 4n_i/27M(GF(q_i)) \leq$$
$$\leq (24a_{i-2}b_i + O(a_{i-2}))M(GF(q_i))$$

это вычисление к вычислению

$$ww^{q_i} \dots w^{q_i^{n_i/27-1}}, w \in GF(q_i^{n_i/27})$$

и так далее. Так как $n_i = 3^{b_i}$, этот процесс закончится через b_i шагов. На каждом очередном шаге требуемая сложность уменьшается асимптотически в три раза, поэтому сложность вычисления $N(x)$ оценивается как

$$\leq (24\frac{3}{2}a_{i-1}b_i + O(a_i))M(GF(q_i)),$$

значит сложность вычисления $N(x), K(x)$ оценивается как

$$\leq (44a_ib_i + O(a_i))M(GF(q_i)).$$

Отсюда следует рекуррентная оценка сложности инвертирования

$$I(a_{i+1}) \leq I(a_i) + n_iM(GF(q_i)) + (44a_ib_i + O(a_i))M(GF(q_i)) \leq$$
$$\leq I(a_i) + (44a_ib_i + O(a_i))M(GF(q_i)).$$

Из нее по индукции получаем, что

$$I(a_n) \leq \sum_{i=1}^{n-1}(44a_ib_i + O(a_i))M(GF(q_i)) + I(a_0) =$$
$$(44a_{n-1}b_{n-1} + O(a_{n-1}))M(GF(q_{n-1})).$$

Так как

$$M(GF(q_{i+1})) \leq$$
$$\leq (12a_ib_i + O(a_i))M(GF(q_i)),$$

то предполагая, что

$$M(GF(q_{i+1})) =$$
$$(12a_ib_i + O(a_i))M(GF(q_i)),$$

получаем асимптотическую оценку

$$I(a_n) \leq (11/3)M(GF(q_n)).$$

Используя оценку

$$M(GF(q_n)) = O\left(3^{2^n+n}2^{(n^2+5n)/2}\right),$$

во всяком случае имеем

$$I(a_n) = (44a_{n-1}b_{n-1} + O(a_{n-1}))M(GF(q_{n-1})) = O\left(3^{2^n+n}2^{(n^2+5n)/2}\right).$$

Обозначая для краткости $a_n = 2 \cdot 3^{2^n}$ через N, имеем

$$I(N) \leq N(\log_3 N)^{n/2+O(1)} = N(\log_3 N)^{(\log_2 \log_3 N)/2+O(1)}.$$

Такие же оценки можно получить и для любого $N = 2 \cdot 3^n$. Для этого выберем k так чтобы $2^{k-1} \leq n < 2^k$ и определим последовательность $a_k = N, a_{i-1} = 2 \cdot 3^{\lceil \log_3(a_i/2))/2 \rceil}$, положим $q_i = 2^{a_i}$, и рассмотрим башню полей

$$GF(q_0) \subset GF(q_1) \subset \ldots \subset GF(q_k).$$

4.2.9 Схемы для инвертирования в произвольных полях $GF(2^n)$

Для их построения можно разложить n на множители, равные степеням простых чисел, построить схемы для инвертирования в полях, размерности которых равны этим множителям методом, указанным выше, сводя их построение к построению схем для инвертирования в полях простой размерности, а потом применить описанный выше метод построения схем составной размерности при условии взаимной простоты сомножителей. Для полей простой размерности можно применить метод аддитивных цепочек. Заметим, что в указанном выше методе вместо простых чисел, при возможности, следует применять размерности, для которых существуют оптимальные нормальные базисы, или базисы малой сложности, что мы и делали в некоторых конкретных примерах.

Указанный метод построения схем похож на метод вычисления ДПФ в конечных полях, который состоит в комбинировании метода Кули-Тьюки для вычисления ДПФ размерности m^k и метода Гуда-Томаса для вычисления ДПФ

размерностей, равных произведению взаимно простых чисел. Для *гладких* чисел, т.е. чисел разлагающихся на малые простые множители, сложность ДПФ размерности n будет довольно близка к $n \log^{O(1)} n$, хотя точную оценку в общем случае дать затруднительно. Подобная же оценка будет справедливой в этом случае и для умножения многочленов над малыми полями и для умножения в полях малой характеристики. Для ДПФ негладкого порядка сложность оценивается еще хуже, несмотря на возможное применение разных ухищрений, например метода Райдера сведения ДПФ простого порядка к циклической свертке на единицу меньшего порядка и метода Винограда сведения вычисления циклической свертки к умножению многочленов по модулям, являющимся неприводимыми делителями круговых многочленов над данным конечным полем.

Для умножения многочленов Шенхаге, как уже упоминалось, предложил использовать преобразование Фурье не над подходящим конечным полем, а над некоторым фактор-кольцом кольца многочленов, и получил асимптотически очень хорошую оценку сложности умножения многочленов. Подобный, но чуть более быстрый алгоритм предложен в [91]. Другой, асимптотически чуть менее быстрый алгоритм имеется в [90].

Для сложности инвертирования подобных асимптотически быстрых алгоритмов при любом n, видимо, не известно. Но в случае гладкого числа n оценка сложности инвертирования будет иметь вид $n^{1+o(1)}$, и будет асимптотически достаточно хорошей, а для чисел специального вида, как отмечалось выше, оценка даже становится близкой к сложности умножения многочленов. К сожалению, эти результаты не имеют практического значения для криптографии над конечными полями, потому что для гладкого n число $q^n - 1$, даже если не будет достаточно гладким, но будет иметь много не очень больших множителей, что благодаря известному алгоритму Полига-Сильбера-Хеллмана (см., например, [41]), фактически открытому еще раньше Нечаевым (но не опубликованному в открытой печати, см. [52]), облегчает дискретное логарифмирование в таких полях. Для криптографии на эллиптических кривых верно то же самое, так как порядок группы точек эллиптической кривой над данным полем делится на порядок группы той же кривой над любым подполем этого поля. Поэтому и там и там предпочитают использовать поля простой размерности. К сожалению, для таких полей из описанных выше приемов удается применять только метод аддитивных цепочек, эффективность которого повышается в случае, если в данном поле существует оптимальный нормальный базис, или хотя бы какой-нибудь базис малой сложности, например гауссов нормальный базис. Именно такие базисы, наряду со стандартными полиномиальными базисами, соответствующими «малочленам», рекомендуют применять все криптографические стандарты.

4.2.10 Применение недвоичных полей

В криптографии эллиптических кривых, как и в кодировании, такие поля применяются редко, так как, вообще говоря, они менее удобны для имплементации. Поэтому мы их как правило не рассматриваем. Отметим все же, что в некоторых работах, например в [84], сделаны попытки ускорить программную имплементацию арифметических вычислений в криптографии как раз с помощью удачного применения недвоичных полей. В [84] предлагается например использовать поля $GF(p^m)$ при $p = 2^n - c, \log_2 c \leq n/2$ («псевдо простые Мерсенна»), а в качестве порождающего базис многочлена берется двучлен вида $x^m - \omega$. В этом случае арифметика в поле $GF(p)$ имплеметируется просто и быстро, так как требует нескольких машинных операций. В случае псевдомерсенновских простых с длиной близкой к машинному слову приведение по модулю сводится к сложению или вычитанию и умножению на малое число, например по формуле

$$2^{32}a + b(\mathrm{mod}2^{32} + c = b - ca(\mathrm{mod}2^{32} + c.$$

Так как умножение довольно медленная операция, в случае c с малым числом двоичных единиц его выгоднее заменить на сложение и сдвиг битов. Например, при $p = 2^{31} - 19$ потребуется два сложения и два сдвига.

Приведение по модулю двучлена $x^m - \omega$ выполняется всего лишь с помощью $m - 1$ сложений и $m - 1$ умножений на ω в поле $GF(p)$. Особенно просто эта операция производится при $\omega = 2$, так ка такое умножение сводится к сдвигу битов. Чуть сложнее эта операция делается для n вида $2^k \pm 2^m$, так как кроме сдвига битов требуется еще одно сложение. Поиск неприводимых двучленов облегчается следующей теоремой из [45]

Двучлен $x^m - \omega$ неприводим над полем $GF(p)$ тогда и только тогда, когда каждый простой делитель m делит e, но не делит $(p-1)/e$, где e — порядок элемента ω, и m кратно 4. Например, при $p = 2^{61} - 1$ в случае, если $p = 1 \bmod 4$.

Упражнение 4.2.2 Выведите из этой теоремы, что если ω — примитивный элемент в поле $GF(p)$, и m делит $p-1$, то $x^m - \omega$ неприводим над полем $GF(p)$. В частности, $x^2 - \omega$ неприводим для любого примитивного элемента.

В [84] в качестве простых особенно рекомендуются максимальные числа вида $p = 2^n \pm 1$ (простые Мерсенна и Ферма), умещающиеся в пределах машинного слова, так как для таких полей арифметика имплементируется особенно просто, потому что приведение по модулю и в случае умножения сводится к сложению или вычитанию, а в случае сложения — к инкременции или декременции.

Но 2 не является примитивным элементом по эти модулям, и в случае простых Ферма $x^m - 2$ всегда приводим, а в случае простых Мерсенна многочлен $x^m - 2$ будет неприводимым только если каждый простой делитель m делит n, но не делит $(p-1)/n$, и m не кратно 4. Например, при $p = 2^{61} - 1$ в качестве m годится только минимум 61. Поэтому в качестве неприводимого двучлена в [84] предложен $x^3 - 37$. Там утверждается, что в поле $GF((2^{61} - 1)^3)$ арифметика работает в два раза быстрее, чем в поле $GF(2^{155}$ согласно [150].

Упражнение 4.2.3 Выведите из предыдущего упражнения, что для числа Ферма $p = 2^{2^k}+1$ двучлен $x^m - 3$ неприводим тогда и только тогда, когда $m = 2^l, l \leq 2^k$.

Поэтому возможно также, что быстрая имплементация может быть сделана для $p = 2^{16} + 1$ и многочлена $x^{16} - 3$.

В [85] дано более развернутое изложение, чем в [85], в частности вычислены таблицы с параметрами рекомендуемых расширений, названных там оптимальными. Однако в программе их вычисляющей имеется не замеченная авторами ошибка в проверке условий приведенной выше теремы, и она дает иногда неправильные результаты, попавшие в таблицы. Так, для чисел Ферма 2^8+1 и $2^{16}+1$ неправильно утверждается неприводимость в соотвествующих полях многочленов вида $x^{2^n} - 2$. На самом деле они приводимы, так как очевидно 2 является квадратичным вычетом в этих полях и данные многочлены приводимы как разности квадратов. Двойка является квадратичным вычетом в силу критерия Эйлера и того факта, что ее порядок в этих полях делит 32, а значит и $(p-1)/2$ тоже, поэтому $2^{(p-1)/2} = 1 \bmod p$.

Упражнение 4.2.4 Проверьте это.

Инвертирование в поле $GF(p^n)$ в [85] сводится к инвертированию в поле $GF(p)$ с помощью формулы

$$x^{-1} = (x^{(p^{n-1}-1)/(p-1)})^p y^{-1},$$

где $y = x^{(p^n-1)/(p-1)} \in GF(p)$, а вычисление y и $x^{(p^{n-1}-1)/(p-1)}$ сводится к возведениям в степени p^i и $\lambda(n-1) + \nu(n-1) - 1$ умножениям. Так, например при $n = 8$ проводятся следующие вычисления

$$x^{\frac{p^7-1}{p-1}} = \left(x^{\frac{p^4-1}{p-1}}\right)^{p^3} x^{\frac{p^3-1}{p-1}}, x^{\frac{p^4-1}{p-1}} = \left(x^{p+1}\right)^{p^2} x^{p+1}, x^{\frac{p^3-1}{p-1}} = \left(x^{p+1}\right)^p x, y = \left(x^{\frac{p^7-1}{p-1}}\right)^p x,$$

требующие 4 полных умножения и одно умножение, в котором вычисляется только одна координата, а также 5 возведений в степени p^k. Возведение в степень p^k в поле $GF(p^n)$ сводится к возведению в эту степень одночленов, а оно выполняется по формуле

$$(x^i)^{p^k} = \omega^s x^e (\mathrm{mod} x^n - \omega, e = p^k i \bmod n, s = (p^k i - e)/n,$$

поэтому возведение в степень p^k требует $n-1$ умножений в поле $GF(p)$. В [85] не отмечено, что в случае составных n вычисления можно выполнить быстрее, например следующим образом. Заметим, что $z_1 = x^{p^4} x \in GF(p^4)$, так как $z_1^{p^4} = z_1$, и $z_2 = z_1^{p^2} z_1 \in GF(p^2)$, так как $z_2^{p^2} = z_2$, тогда

$$z_2^p z_2 = x^{\frac{p^8-1}{p^4-1}\frac{p^4-1}{p^2-1}\frac{p^2-1}{p-1}} = x^{\frac{p^8-1}{p-1}} = y \in GF(p),$$

$$z_2^p z_1^{p^2} x^{p^4} = x^{\frac{p^8-p}{p-1}},$$

потому что

$$z_2^p z_1^{p^2} x^{p^4} x = z_2^p z_1^{p^2} z_1 = z_2^p z_2 = y.$$

В этом вычислении использовались только 3 возведения в степени p^k, причем одно в поле $GF(p^8)$, одно в поле $GF(p^4)$, и одно в поле $GF(p^2)$, а также 5 умножений, из которых одно в поле $GF(p^8)$, но его результат принадлежит подполю $GF(p^4)$, одно в поле $GF(p^4)$, но его результат принадлежит подполю $GF(p^2)$, одно в поле $GF(p^2)$, но его результат принадлежит подполю $GF(p)$, еще в одном умножении элемент подполя $GF(p^4)$ умножается на элемент подполя $GF(p^2)$, и полученный элемент $z_2^p z_1^{p^2}$ подполя $GF(p^4)$ умножается на элемент поля $GF(p^8)$. Так как вместе с двучленом $x^8 - \omega$ неприводимыми над полем $GF(p)$ являются двучлены $x^4 - \omega$, $x^2 - \omega$, и если $\alpha\ in GF(p^8)$ корень двучлена $x^8 - \omega$, то α^2 будет корнем $x^4 - \omega$, значит $\alpha^2 \in GF(p^4)$, и $\alpha^4 \in GF(p^2)$ будет корнем $x^2 - \omega$, поэтому базисом подполя $GF(p^4)$ будет «подбазис» $\{1, \alpha^2, \alpha^4, \alpha^6\}$ базиса $\{1, \alpha, \ldots, \alpha^7\}$ поля $GF(p^8)$, а базисом подполя $GF(p^2)$ будет «подбазис» $\{1, \alpha^4\}$. Значит для нахождения координат в подполях нужно просто взять соответствующие координты в поле, игнорируя нули; так как при стандартном умножении в этих базисах формулы для разных координат не имеют общих переменных, то, например, сложность умножения в поле $GF(p^8)$ с результататом из подполя $GF(p^4)$ будет вдвое меньше сложности общего умножения, так как формулы, дающие нулевые координаты можно не вычислять. Для умножения и возведения в степени в подполе естественно использовать только координаты, соответствующие его базису, игнорируя остальные нулевые координаты, поэтому эти операции фактически можно выполнять, как внутренние операции этого подполя. Аналогичное утверждение верно для умножения на элемент подполя, точнее умножение на элемент подполя сводится к двум умножениям в этом подполе.

Если через $M(GF(q))$ обозначить сложность умножения в рассматриваемых базисах данных полей, то общая оценка сложности интвертирования будет

$$M(GF(p^8))/2 + M(GF(p^4))/2 + M(GF(p^2))/2 + 2M(GF(p^4)) + 2M(GF(p^2)) + (7+3+1)M(GF(p)) =$$

$$= M(GF(p^8)) + 3M(GF(p^4))/2 + M(GF(p^2))/2 + 10M(GF(p)).$$

Так как операция умножения в поле $GF(p)$ медленнее сложения, то для умножения многочленов над полем $GF(p)$ в [85] предлагается уменьшать количество операций умножения с помощью введения $n(n-1)/2$ произведений вида $(a_i + a_j)(b_i + b_j)$. Из них с помощью еще n произведений вида $a_i b_i$ можно вычислить все суммы вида

$$a_i b_j + a_j b_i = (a_i + a_j)(b_i + b_j) + a_i b_i + a_j b_j.$$

Мультипликативная сложность основанного на применении этих тождеств алгоритма умножения многочленов степени $n-1$ равна $n(n+1)/2$. Эта идея похожа на использованную нами выше идею ускорения умножения в нормальных

базисах. Но в отличие от нормальных базисов указанный алгоритм дает выигрыш только в мультипликативной сложности, и существенно проигрывает стандартному алгоритму в аддитивной сложности. Тем не менее, рекуррентное применение этого алгоритма при некоторых значениях n улучшает оценки, получаемые непосредственным применением метода Карацубы. Например, для умножения квадратных трехчленов рекуррентое применение метода Карацубы требует 7 умножений, а предложенная в [85] схема требует 6 умножений и 13 сложений. Заметим, однако, что выше нами использовалась еще лучшая схема, имеющая только 12 сложений. Эта схема выглядит следующим образом:

$$(a_0 + a_1x + a_2x^2)(b_0 + b_1x + b_2x^2) = c_0 + c_1x + c_2x^2 + c_3x^3 + c_4,$$

$$c_0 = a_0b_0, c_4 = a_2b_2, c_1 = (a_0 + a_1)(b_0 + b_1) + (a_0b_0 + a_1b_1),$$
$$c_2 = (a_0b_0 + a_1b_1) + (a_0 + a_2)(b_0 + b_2) + a_2b_2,$$
$$c_3 = (a_1 + a_2)(b_1 + b_2) + a_1b_1 + a_2b_2.$$

Применяя ее в методе Карацубы, получаем схему для умножения многочленов пятой степени лучшую по числу сложений, чем схема [85]. Обе эти схемы можно использовать для ускорения операций в поле $GF(p^6)$ методом, подобным приведенному выше.

В приложении Б приведена составленная на основе компьютерных вычислений таблица размерностей полей $GF(2^n)$ при $1000 < n10000$, для которых существуют оптимальные нормальные базисы.

Глава 5

Алгоритмы на эллиптических кривых

5.1 Алгоритм сложения и удвоения

5.1.1 Общая схема алгоритма сложения и правила вывода частных формул

В соответствии с определением операции сложения в группе точек эллиптической кривой общая схема алгоритма сложения точек $P_1 = (x_1, y_1)$ и $P_2 = (x_2, y_2)$ представляется следующим образом:

Алгоритм 5.1.1

Вход: Коэффициенты эллиптической кривой,
точки $P_1 = (x_1, y_1)$ и $P_2 = (x_2, y_2)$.
Выход: $P = (x, y) = P_1 + P_2$.
Вычислить P : если $P_1 = O$ то $P = P_2$,
если $P_2 = O$, то $P = P_1$,
если $P_2 = -P_1$, то $P = O$,
если $x_1 \neq x_2$, то $P = -(x_3, y_3)$,
если $x_1 = x_2$ и $y_1 = y_2$, то $P = 2P_1 = -(x_3, y_3)$.

Примечания. 1. Координаты точек $-(x_3, y_3)$ подсчитываются по формулам, вытекающим из определения 3.1 в главе 1, в зависимости от вида эллиптической кривой, по общим правилам, описанным ниже. Эти формулы выведены в следующем параграфе:

для эллиптических кривых над полем $\mathcal{F}$ характеристики $char\mathcal{F}$, $4 \preceq char\mathcal{F}$ это формулы (1.10) и (1.11);

для эллиптических кривых над полем характеристики 3 – формулы (1.17) и (1.18);

для несуперсингулярных эллиптических кривых над полем характеристики 2 – формулы (1.24) и (1.125);

для суперсингулярных эллиптических кривых над полем характеристики 2 – формулы (1.31) и (1.32).

Эти формулы использованы в Алгоритмах 4.2, 4.3, 4.4 и 4.5 в параграфе 3.4, которые детализируют приведенную общую схему с учетом типа кривой и характеристики поля, над которым она определяется.

2. Точка O имеет определённый идентификатор, например, она может быть представлен не удовлетворяющей уравнению этой кривой парой (a, b) элементов поля, над которым определена эллиптическая кривая.

3. Элемент O никогда не используется в качестве входных данных и не возникает как значениие выходных данных при исполнении двух последних строк схемы алгоритма. Поэтому изучаемые ниже формулы, определяющие операции для этих случаев, не имеют смысла, если один или оба аргумента являются точкой O.

4. Алгоритм включает случай удвоения точки (последняя строка алгоритма), когда слагаемые одинаковы, но не взаимно противоположны.

5. Некоторые кривые имеют взаимно-противоположные точки (такие, что $P = -P$), к таким точкам правило удвоения не применимо, но в этом случае алгоритм заканчивает работу, не доходя до последней строки.

Детализируя построения, лежащие в основе определения (3.1) из главы 1 суммы двух точек, в принципе можно построить формулы, выражающие координаты точки $(x_3, y_3') = -(x_3, y_3)$ через координаты x_1, x_2, y_1, y_2 (или x', y' в случае равенства слагаемых, не являющихся взаимно-противоположными) для суперсингулярной или несуперсингулярной кривой любой характеристики, но эти формулы окажутся крайне громоздкими и неудобными для реализации. В то же время в приложениях используется, как правило, определенная конкретная кривая. Формулы для кривых, описываемых уравнениями частного вида, определяемого характеристикой поля и типом кривой, оказываются весьма компактными. Вывод таких формул осуществляется по общим правилам, конкретизирующим упомянутые построения, лежащие в основе определения суммы двух точек.

Эти правила следующие.

Сначала следует получить координаты точки $R = (x_3, y_3)$.

Для этого нужно привести уравнение кривой к виду

$$F(X, Y) = 0$$

и выполнить следующие действия

а) Если точки $P_1 = (x_1, y_1)$ и $P_2 = (x_2, y_2)$ разные:

Принять $\lambda = \frac{y_2 - y_1}{x_2 - x_1}$, $\beta = y_1 - \lambda x_1$.

Подставить $\lambda X + \beta$ вместо Y в $F(X, Y)$:

$$F(X, \lambda X + \beta)) = 0.$$

Привести левую часть к виду

$$X^3 + \gamma X^2 + aX + b = 0,\ \gamma, a\ b \in \mathcal{F}.$$

Вычислить

$$x_3 = -\gamma - x_1 - x_2. \tag{5.1}$$

Вычислить

$$y_3 = y_1 + \lambda(x_3 - x_1). \tag{5.2}$$

б) Если точки одинаковые, $P_1 = P_2 = P = (x, y);\ 2P = (x_3, y_3)$:

По формальным правилам дифференцирования получить выражение

$$\lambda'(X, Y) = -\frac{dF(X, Y)/dX}{dF(X, Y)/dY}$$

и поэтому выражению при $X = x,\ Y = y$ вычислить элемент

$$\lambda' = \lambda'(x, y)$$

поля $\mathcal{F}$, затем вычислить $\beta = y_1 - \lambda' x_1$.

Подставить $\lambda' X + \beta$ вместо Y в $F(X, Y)$:

$$F(X, \lambda' X + \beta)) = 0.$$

Привести левую часть к виду

$$X^3 + \gamma' X^2 + aX + b = 0, \gamma', a, b \in \mathcal{F}.$$

Вычислить

$$x_3 = -\gamma' - 2x. \tag{5.3}$$

Вычислить

$$y_3 = y_1 + \lambda'(x_3 - x_1). \tag{5.4}$$

Далее с учетом

$$-P = (x, -a_1 x - a_3 - y) \tag{5.5}$$

вычислить $-R = -(x_3, y_3) = (x_3, \tilde{y}_3)$, где $\tilde{y} = -a_1 x - a_3 - y$ при соответствующих уравнению кривой коэффициентах a_1 и a_3, то есть получить результат сложения или удвоения.

Примечание. Окончательные формулы получают конкретизацией выражений (1.1)-(1.5) с учетом уравнения эллиптической кривой.

Заметим, что выражение для γ' можно получить из γ просто заменой константы λ константой λ'.

5.1.2 Частные формулы сложения и удвоения

Сложение и удвоение в $\mathcal{EF}$ над полем $\mathcal{F}$ характеристики $char\mathcal{F}, 4 \preceq char\mathcal{F}$. Уравнение эллиптической кривой в этом случае (характеристика поля $char\mathcal{F} \neq \{2,3\}$) приводится к виду (3.3) (см. главу 1) с ненулевым дискриминантом (3.4) правой части этого уравнения. Например, к этому классу относятся эллиптические кривые с ненулевым дискриминантом над полем R действительных чисел, в каждой точке таких кривых можно провести касательную.

В этом случае условие гладкости кривой состоит в требовании, что кубический многочлен справа не имеет кратных корней. А это выполняется тогда и только тогда, когда его дискриминант не равен нулю.

В соответствии с общим определением (1.3) для данного частного случая кривой ($a_1 = a_3 = 0$) отрицанием точки $P = (x, y) \neq O$ явлется точка $-P = (x, -y)$.

Коэффициент γ в формуле (1.1) в этом случае есть $-\lambda^2$, где

$$\lambda = \frac{y_2 - y_1}{x_2 - x_1}$$

, а коэффициент γ' в формуле (1.4) есть $-(\lambda')^2$, где

$$\lambda' = \left(\frac{3x^2 + a}{2y}\right)$$

(упражнение 3.?.?).

В случае $P \neq Q$ это позволяет получить выражение для x_3 и затем для y_3 и, наконец, для $\tilde{y}_3$ и, тем самым, вычислить сумму $P + Q = (x_3, \tilde{y}_3)$:

$$x_3 = \lambda^2 - x_1 - x_2; \tag{5.6}$$

$$y_3 = y_1 + \lambda(x_3 - x_1),$$

$$\tilde{y}_3 = -y_3 = -y_1 + \left(\frac{y_2 - y_1}{x_2 - x_1}\right)(x_1 - x_3). \tag{5.7}$$

В случае $P = Q$ выражения для x_3 и y_3 и координаты точки, соответствующей сумме $P + Q = (x_3, \tilde{y}_3) = 2P$, то есть удвоенной точки P получаются через координаты x, y точки $P = (x, y)$ следующим образом:

$$x_3 = (\lambda')^2 - 2x; \tag{5.8}$$

$$y_3 = y + \lambda'(x3 - x);$$

$$\tilde{y}_3 = -y + \lambda'(x - x_3). \tag{5.9}$$

Окончательно для эллиптической кривой над полем характеристики $char\mathcal{F}, 4 \preceq char\mathcal{F}$ получаем (используя (1.6) и(1.7),(1.8) и(1.9):)

а)при $P_1 = (x_1, y_1) \neq P_2 = (x_2, y_2)$:

$$P_1 + P_2 = -R = (x_3, -y_3) = (x_3, -y_1 + \lambda(x_1 - x_3)), \tag{5.10}$$

где

$$\lambda = \frac{y_2 - y_1}{x_2 - x_1},\ x_3 = \lambda^2 - x_1 - x_2;$$

б) при $P_1 = P_2 = P = (x, y)$:

$$2P = -R = (x_3, -y_3) = (x_3, -y + \lambda'(x - x_3)). \tag{5.11}$$

где

$$\lambda' = \frac{3x^2 + a}{2y},\ x_3 = (\lambda')^2 - 2x.$$

Пример 5.1.1 Пусть $P = (0, 0)$ на эллиптической кривой

$$Y^2 + Y = X^3 - X^2. \tag{5.12}$$

Найдем $2P = P + P$ и $3P = P + 2P$.

Решение. Прежде всего преобразуем уравнение (1.12) в уравнение вида (3.3) (глава 1) путем замены переменных $Y \to Y - 1/2,\ X \to X + 1/3$. Получим

$$Y^2 = X^3 - \frac{1}{3}x + \left(\frac{1}{4} - \frac{2}{27}\right).$$

На этой кривой точка P становится точкой $Q = (-1/3, 1/2)$. Используя (1.9), получим $2Q = (2/3, -1/2)$. Затем из (1.4) мы имеем $3Q = 2Q + Q = (2/3, 1/2)$. Заметим, что $3Q = -(2Q)$, и следовательно Q является точкой порядка 5, то есть $5Q = O$. Возвращаясь к исходной кривой, имеем $2P = (1, -1),\ 3P = (1, 0) = -2P$.

Сложение и удвоение в $\mathcal{EF}$ над полем $\mathcal{F}$ характеристики 3. В этом случае без ограничения общности мы можем полагать, что в уравнении (3.1) эллиптической кривой общего вида (глава 1) $a_1 = a_3 = 0$, а коэффициент a_2 не обязательно равен 0, и вместо этого уравнения использовать уравнение (3.2) из первой главы, то есть уравнение

$$Y^2 = X^3 + a_2X^2 + a_4X + a_6,\ a_i \in \mathcal{F}.$$

Отметим, что согласно (1.5) при указанных значениях коэффициентов a_1 и a_3 противоположный относительно точки $P = (x, y) \neq O$ элемент $-P = (x, \tilde{y}) = (x, -y)$.

В рассматриваемом случае

$$F(X, Y) = Y^2 - X^3 - a_2X^2 - a_4X - a_6 = 0.$$

Для сложения разных точек $P_1 = (x_1, y_1)$ и $P_2 = (x_2, y_2)$ используем

$$F(X, \lambda X + \beta) = (\lambda X + \beta)^2 - X^3 - a_2X^2 - a_4X - a_6 = 0,$$

где

$$\lambda = \frac{y_2 - y_1}{x_2 - x_1};\ \beta = y_1 - \lambda x_1.$$

Отсюда получим (как коэффициент при X^2 после приведения к полиномиальному виду) константу

$$\gamma = -(\lambda^2 - a_2),$$

и выражения для $x_3,\ y_3$ и $\tilde{y}_3$, определяющие, в конечном итоге, координаты точки $P_1 + P_2 = (x_3, \tilde{y}_3)$ следующие

$$x_3 = -\gamma - x_1 - x_2 = \lambda^2 - a_2 - x_1 - x_2; \tag{5.13}$$

$$y_3 = y_1 + \lambda(x_3 - x_1).$$

$$\tilde{y}_3 = -y_1 + \lambda(x_1 - x_2). \tag{5.14}$$

Для удвоения точки $P = (x, y)$ вычислим

$$\lambda'(X, Y) = -\frac{-3X^2 - 2a_2X - a_4}{2Y},$$

$$\lambda' = \lambda'(x, y) = \frac{3x^2 + 2a_2x + a_4}{2y}$$

и, используя

$$\gamma' = -(\lambda')^2 - a_2)$$

(в выражении для γ заменили λ на λ'), получим координаты точки $2P$:

$$@?x_3 = -\gamma' - 2x = ((\lambda')^2 - a_2) - 2x; \tag{5.15}$$

$$y_3 = y + \lambda'(x_3 - x),$$

$$\tilde{y}_3 = -y + \lambda'(x - x_3), \tag{5.16}$$

Учитывая, что характеристика поля равна 3, выражение для λ' можно упростить

$$\lambda' = -\frac{-3x^2 - 2a_2x - a_4}{2y} = \frac{a_2x - a_4}{y}.$$

Окончательно для эллиптической кривой над полем характеристики 3 получаем (используя (1.13) и(1.14),(1.15) и(1.16))

а)При $P_1 = (x_1, y_1) \neq P_2 = (x_2, y_2)$:

$$P_1 + P_2 = -R = (x_3, -y_3) =$$

$$= (x_3,\ -y_1 + \lambda(x_1 - x_3)). \tag{5.17}$$

где

$$\lambda = \frac{y_2 - y_1}{x_2 - x_1},\ x_3 = \lambda^2 - a_2) - x_1 - x_2;$$

б) При $P_1 = P_2 = P = (x, y)$:

$$2P = -R = (x_3, -y_3) =$$

$$= (x_3, -y + \lambda'(x - x_3)), \tag{5.18}$$

где

$$\lambda' = \frac{a_2 x - a_4}{y},\ x_3 = (\lambda')^2 - a_2) - 2x.$$

Сложение и удвоение в $\mathcal{EF}$ над полем $\mathcal{F}$ характеристики 2 для несуперсингулярных эллиптических кривых. Как отмечено в первой главе, эллиптическая кривая над полем характеристики $char\mathcal{F} = 2$ в несуперсингулярном случае имеет уравнение

$$Y^2 + a_1 XY = X^3 + a_2 X^2 + a_6,\ a_i \in \mathcal{F} \tag{5.19}$$

или

$$F(X, Y) = Y^2 + a_1 XY + X^3 + a_2 X^2 + a_6 = 0.$$

Точка кривой $-(x, y) = (x, \tilde{y})$, противоположная относительно точки кривой (x, y) в этом случае определятся в соответствии с выражением (1.5) как $(x, x+y)$.

Для сложения разных точек $P_1 = (x_1, y_1)$ и $P_2 = (x_2, y_2)$ используем

$$F(X, \lambda X + \beta) = (\lambda X + \beta)^2 + a_1 X(\lambda X + \beta) + X^3 + a_2 X^2 + a_6 = 0,$$

где

$$\lambda = \frac{y_2 - y_1}{x_2 - x_1};\ \beta = y_1 - \lambda x_1.$$

Откуда получим (как коэффициент при X^2 после привеления к полиномиальному виду) константу

$$\gamma = -\lambda^2 + \lambda + a_2$$

и выражения для $x_3,\ y_3$ и $\tilde{y}_3$, определяющие координаты точки $R = -(P_1 + P_2) = (x_3, y_3)$ и суммы $P_1 + P_2) = (x_3, \tilde{y}_3)$:

$$x_3 = -\gamma + x_1 + x_2 = (\lambda^2 + \lambda + a_2) + x_1 + x_2; \tag{5.20}$$

$$y_3 = y_1 + \lambda(x_3 + x_1);$$

$$\tilde{y}_3 = x_3 + y_1 + \lambda(x_3 + x_1) \tag{5.21}$$

Для удвоения точки $P = (x, y)$ вычислим

$$\lambda'(X, Y) = \frac{3X^2 + 2a_2 + Y}{2Y + X} = \frac{X^2 + Y}{X};$$

$$\lambda' = \lambda'(x, y) = \frac{x^2 + x}{x} = x + \frac{y}{x}$$

и, используя

$$\gamma' = (\lambda')^2 + \lambda' + a_2,$$

получим выражения для $x_3,\ y_3$ и $\tilde{y}_3$, определяющие координаты точки $R = -2P$ и удвоенной точки $2P = (x_3, y_3)$:

$$x_3 = \gamma' + 2x = (\lambda')^2 + \lambda' + a_2 \tag{5.22}$$

$$y_3 = y + \lambda'(x_3 + x) = y + \lambda'(x + x_3)$$

$$\tilde{y}_3 = x_3 + y + \lambda'(x_3 + x) = x_3 + y + \lambda'(x + x_3) =$$

$$= x^2 + (\lambda' + 1)x_3. \tag{5.23}$$

Окончательно для несуперсингулярной эллиптической кривой над полем характеристики 2 получаем (используя (1.20) и(1.21),(1.22) и(1.23)):

а)при $P_1 = (x_1, y_1) \neq P_2 = (x_2, y_2)$:

$$P_1 + P_2 = (x_3, x_3 + y_1 + \lambda(x_3 + x_1)), \tag{5.24}$$

где

$$\lambda = \frac{y_2 - y_1}{x_2 - x_1},\ x_3 = (\lambda)^2 + \lambda + a_2 + x_1 + x_2;$$

б) при $P_1 = P_2 = P = (x, y)$:

$$2P = (x_3, x^2 + (\lambda' + 1)x_3), \tag{5.25}$$

где

$$\lambda' = x + \frac{y}{x},\ x_3 = (\lambda')^2 + (\lambda') + a_2.$$

Пример 5.1.2 Группа точек эллиптической кривой над полем $GF(2^2)$ (порождаемом корнем многочлена $x^2 + x + 1$ над $GF(2)$)

$$Y^2 + XY = X^3 + X^2 + 1$$

кроме точки $\mathcal{O}$ включает точки $((0,0),(0,1)), ((0,1),(1,0)), ((0,1),(1,1)),$
$((1,0),(0,1)), ((1,0),(1,1)), ((1,1),(0,1)), ((1,1)(1,0))$. Заметим, что в данном случае $((0,0),(0,1)) = ((0,0),(0,1)$ («самообратная» точка).

Вычислим, например, $2P = 2((0,1),(1,0)) = (x_3, y_3) =$
$((\lambda')^2 + \lambda' + a_2, x^2 + (\lambda' + 1)x_3) =$
$= ((1,1)^2 + (1,1) + (0,1)), (0,1)^2 + ((1,1) + (0,1))x_3)) =$
$= ((1,0) + (1,1) + (0,1), ((0,1) + (1,0)x_3) = ((0,0),(0,1)),$
где $\lambda' = (0,1) + \frac{(1,0)}{(0,1)} = (1,1)$.

Проверим, что $2P + (-P) = P$, где $P = ((0,1),(1,0)),\ -P = ((0,1),(1,1) :\ 2P + (-P) =$
$((0,0),(0,1)) + ((0,1),(1,1)) =$
$= (\lambda^2 + \lambda + a_2 + x_1 + x_2, x_3 + y_1 + \lambda(x_3 + x_1) =$
$= ((1,1) + (1,0) + (0,1) + (0,0) + (0,1), x_3 + (0,1) + (1,0)((0,0) + x_3) =$
$= ((0,1), (0,1) + (0,1) + (1,0)((0,0) + (0,1))) = ((0.1),(1,0)) = P.$
где $\lambda = \frac{(0,1)+(1,1)}{(0,0)+(0,1)} = \frac{(1,0)}{(0,1)} = (1,0) \cdot (0,1)^{-1} = (1,0) \cdot (0,1) = (1,0).$

Сложение и удвоение в $\mathcal{EF}$ над полем $\mathcal{F}$ характеристики 2 для суперсингулярных эллиптических кривых. Если $a_1 = a_2 = 0$ в общем уравнении элиптической кривой (3.1) в главе 1, а $a_3 \neq 0$ и char$\mathcal{F}$=2 (суперсингулярный случай), то имеем уравнение.

$$Y^2 + a_3 Y = X^3 + a_4 X + a_6,\ a_i \in \mathcal{F} \tag{5.26}$$

В этом случае

$$F(X,Y) = Y^2 + a_3 Y + X^3 + a_4 X + a_6.$$

При значениях коэффициентов $a_3 \neq 0$ и $a_1 = a_2 = 0$ противоположная относительно точки $P = (x,y) \neq O$ точка (см. (1.5))есть

$$-P = (x, y+1).$$

Для сложения разных точек x_1, y_1 и x_2, y_2 используем

$$F(X, \lambda X + \beta) = (\lambda X + \beta)^2 + a_3(\lambda X + \beta) + X^3 + a_4 X + a_6 = 0.$$

где

$$\lambda = \frac{y_2 - y_1}{x_2 - x_1};\ \beta = y_1 - \lambda x_1.$$

Отсюда после приведения к полиномиальному виду получим (как коэффициент при X^2) константу

$$\gamma = \lambda^2$$

и выражения для $x_3,\ y_3$ и $\tilde{y}_3$, определяющие координаты точки $R = -(P_1 + P_2) = (x_3, y_3)$ и суммы $P_1 + P_2) = - = (x_3, \tilde{y}_3)$:

$$x_3 = \gamma + x_1 + x_2 = \lambda^2 + x_1 + x_2, \tag{5.27}$$

$$y_3 = \lambda(x_1 + x_3) + y_1.$$

$$\tilde{y}_3 = \lambda(x_1 + x_3) + y_1 + 1. \tag{5.28}$$

Для получения координат удвоенной точки $2P = 2(x,y)$ рассмотрим выражение

$$\lambda'(X,Y) = \frac{3X^2 + a_4}{2Y + a_3} = \frac{X^2 + a_4}{a_3};$$

$$\lambda' = \lambda'(x,y) = \frac{x^2 + a_4}{a_3}.$$

Используем

$$\gamma' = (\lambda')^2$$

(в выражение для γ подставили λ' вместо λ) и получим выражения для $x_3,\ y_3$ и $\tilde{y}_3$, определяющие координаты точки $R = -2P$ и удвоенной точки $2P = (x_3, y_3)$:

$$x_3 = \gamma + 2x = \gamma = (\lambda')^2; \tag{5.29}$$

$$y_3 = \lambda'(x + x_3) + y;$$

$$\tilde{y}_3 = \lambda'(x + x_3) + y + 1. \tag{5.30}$$

Окончательно для суперсингулярной эллиптической кривой над полем характеристики 2 получаем (используя (1.25) и(1.26),(1.27) и(1.28))

а)При $P_1 = (x_1, y_1) \neq P_2 = (x_2, y_2)$:

$$P_1 + P_2 = (x_3, \tilde{y}_3) = (x_3, \lambda(x_1 + x_3) + y_1 + 1), \tag{5.31}$$

где

$$\lambda = \frac{y_1 + y_3}{x_1 + x_3},\ x_3 = \lambda^2 + x_1 + x_2;$$

б) при $P_1 = P_2 = P = (x, y)$:

$$2P = -R = (x_3, \tilde{y}_3) = (x_3, \lambda'(x + x_3) + y + 1), \tag{5.32}$$

где

$$\lambda' = \frac{x^2 + a_4}{a_3},\ x_3 = \ (\lambda')^2.$$

При $a_3 = a_4 = 1$

$$2P = (x^4 + 1, x^4 + y^4).$$

Пример 5.1.3 Группа точек эллиптической кривой

$$Y^2 + Y = X^3 + X$$

над $GF(2)$ кроме точки $\mathcal{O}$ включает точки $(0,0), (0,1), (1,0), (1,1)$.

Вычислим, например, $2P = 2(0,0) = (0^4 + 1, 0^4 + 0^4) = (1,0)$.

Проверим, что $2P + (-P) = P$, где $P = (0,0),\ -P = (0,1)$:
$(1,0) + (0,1) = (\lambda^2 + x_1 + x_2, \lambda(x_1 + x_3) + y_1 + 1) =$
$= (1 + 1 + 0, 1(0 + (1 + 1 + 0)) + 0 + 1) = (0,0),$
где $\lambda = \frac{0+1}{0+1} = \frac{1}{1} = 1 \cdot 1^{-1} = 1 \cdot 1 = 1.$

5.1.3 Алгоритмы сложения и удвоения в группе точек эллиптической кривой

Приведем алгоритмы сложения и удвоения, конрктизирующие общую схему алгоритма с учетом особенностей эллиптической кривой.[1]

Алгоритм сложения и удвоения для эллиптических кривых над полем характеристики $char\mathcal{F}, 4 \preceq char\mathcal{F}$ **.**

[1]Напомним, что точка O имеет определённый идентификатор, например, она может быть представлен не удовлетворяющей уравнению этой кривой парой (a, b) элементов поля, над которым определена эллиптическая кривая.

Алгоритм 5.1.2

Вход: Коэффициент a эллиптической кривой (3.3) (глава 1),
точки $P_1 = (x_1, y_1)$ и $P_2 = (x_2, y_2)$.
Выход: $P = (x, y) = P_1 + P_2$.
Если $P_1 = O$ то вернуть $P = P_2$,
если $P_2 = O$, то вернуть $P = P_1$,
если $P_2 = -P_1$, то вернуть $P = O$,
если $x_1 \neq x_2$, то
вычислить $\lambda = \frac{y_2 - y_1}{x_2 - x_1}$, $x_3 = \lambda^2 - x_1 - x_2$
вернуть $P = (x_3, -y_1 + \lambda(x_1 - x_3))$,
принять $x = x_1,\ y = y_1$, вычислить $\lambda = \frac{3x^2 + a}{2y}$, $x_3 = \lambda^2 - 2x$;
вернуть $(x_3, -y + \lambda(x - x_3))$.

Алгоритм сложения и удвоения для эллиптических кривых над полем характеристики 3 .

Алгоритм 5.1.3

Вход: Коэффициенты a_2, a_3 эллиптической кривой (3.2) (глава 1),
точки $P_1 = (x_1, y_1)$ и $P_2 = (x_2, y_2)$.
Выход: $P = (x, y) = P_1 + P_2$.
Если $P_1 = O$ то вернуть $P = P_2$,
если $P_2 = O$, то вернуть $P = P_1$,
если $P_2 = -P_1$, то вернуть $P = O$,
если $x_1 \neq x_2$, то
вычислить $\lambda = \frac{y_2 - y_1}{x_2 - x_1},\ x_3 = \lambda^2 - a_2 - x_1 - x_2$;
вернуть $P = (x_3, -y_1 + \lambda(x_1 - x_3))$,
принять $x = x_1,\ y = y_1$, вычислить $\lambda = \frac{a_2 x - a_4}{y},\ x_3 = \lambda^2 - a_2 - 2x$;
вернуть $(x_3, -y + \lambda(x - x_3))$.

Алгоритм сложения и удвоения для несуперсингулярных эллиптических кривых над полем характеристики 2.

Алгоритм 5.1.4

Вход: Коэффициент a_2 эллиптической кривой (1.19),
точки $P_1 = (x_1, y_1)$ и $P_2 = (x_2, y_2)$.
Выход: $P = (x, y) = P_1 + P_2$.
Если $P_1 = O$ то вернуть $P = P_2$,
если $P_2 = O$, то вернуть $P = P_1$,
если $P_2 = -P_1$, то вернуть $P = O$,
если $x_1 \neq x_2$, то
вычислить $\lambda = \frac{y_2 - y_1}{x_2 - x_1},\ x_3 = \lambda^2 + \lambda + a_2 + x_1 + x_2$;
вернуть $P = (x_3,\ x_3 + y_1 + \lambda(x_3 + x_1))$,
принять $x = x_1,\ y = y_1$, вычислить $\lambda = x + \frac{y}{x},\ x_3 = \lambda^2 + \lambda + a_2$;
вернуть $(x_3, x^2 + (\lambda + 1)x_3)$.

Алгоритм сложения и удвоения для суперсингулярных эллиптических кривых над полем характеристики 2.

Алгоритм 5.1.5

Вход: Коэффициенты a_3, a_4 эллиптической кривой (1.26),
точки $P_1 = (x_1, y_1)$ и $P_2 = (x_2, y_2)$.
Выход: $P = (x, y) = P_1 + P_2$.
Если $P_1 = O$ то вернуть $P = P_2$,
если $P_2 = O$, то вернуть $P = P_1$,
если $P_2 = -P_1$, то вернуть $P = O$,
если $x_1 \neq x_2$, то
вычислить $\lambda = \frac{y_2 - y_1}{x_2 - x_1},\ x_3 = \lambda^2 + x_1 + x_2$;
вернуть $P = (x_3, \lambda(x_1 + x_3) + y_1 + 1)$,
принять $x = x_1,\ y = y_1$,
если $a_3 = a_4 = 1$, то вернуть $P = (x^4 + 1, x^4 + y^4)$
вычислить $\lambda = \frac{x^2 + a_4}{a_3},\ x_3 = \lambda^2$,
вернуть $P = ((x_3, \lambda(x + x_3) + y + 1)$.

Обратим внимание, что при исполнении любого из приведенных алгоритмов наиболее трудоемкая операция в арифметике конечного поля – мультипликативное обращение элемента поля – выполняется однократно.

5.2 Эллиптические кривые над $GF(2^n)$

В криптографии используются и кривые над полями нечетной характеристики, но чаще все же находят применение кривые над полем $GF(2^n)$ в связи с большим удобством таких полей для программной и схемной имплементации.

Рассмотрим пример кривой над полем $GF(2^4)$.

Пример 5.2.1 Выберем в нем стандартный базис с неприводимым многочленом $x^4 + x + 1$. Элементы поля представляем в виде многочленов степени не больше 3, записываемых в виде

вектора коэффициентов. Например многочлен x записываем как вектор $(0,0,1,0)$, а константу 1 — как вектор $(0,0,0,1)$.

Упражнение 5.2.1 Проверьте, что x является примитивным элементом поля, который далее обозначаем γ. Составьте таблицу его степеней $1=\gamma^0, \gamma=\gamma^1, \gamma^2, \ldots \gamma^{14}$.

Пример 5.2.2 Рассмотрим несуперсингулярную кривую $y^2+xy=x^3+\gamma^4x^2+1$. Проверим, что точка (γ^3,γ^{13}) лежит на кривой. Для этого достаточно проверить равенство:

$$\gamma^{26}+\gamma^3\gamma^{13}=\gamma^9+\gamma^4\gamma^6+1,$$

очевидно равносильное равенству

$$\gamma^{11}+\gamma=\gamma^9+\gamma^{10}+1,$$

которое легко проверить с помощью таблицы степеней.

Упражнение 5.2.2 Проверьте, что эта кривая состоит из 15 конечных точек:

$$(1,\gamma^{13}),(\gamma^3,\gamma^{13}),(\gamma^5,\gamma^{11}),(\gamma^6,\gamma^{14}),(\gamma^9,\gamma^{13}),(\gamma^{10},\gamma^8),(\gamma^{12},\gamma^{12}),$$

$$(1,\gamma^6),(\gamma^3,\gamma^8),(\gamma^5,\gamma^3),(\gamma^6,\gamma^8),(\gamma^9,\gamma^{10}),(\gamma^{10},\gamma),(\gamma^{12},0),(0,1).$$

5.2.1 Суперсингулярные кривые

основное удобство использования суперсингулярных эллиптических кривых в том, что для них легко вычислить порядок соответствующей группы $\mathcal{EK}$ (здесь $\mathcal{K}=GF(2^n)$), в то время, как определение порядка несуперсингулярной кривой проблематично. Суперсингулярные кривые особенно удобны для создания с минимальными усилиями самодельной ECC-криптосистемы. Для их использования можно обойтись без сложного алгоритма вычисления порядка кривой. Все необходимые сведения можно найти в этом разделе.

Произвольная суперсингулярная кривая над полем характеристики два, как отмечалось выше, изоморфна кривой с уравнением

$$Y^2+a_3Y=X^3+a_4X+a_6,\ a_i\in GF(2^n), a_3\neq 0.$$

Заменой переменных $X=a_3x, Y=a_3y$ его можно свести к виду

$$Y^2+Y=a_2X^3+a_4X+a_6,\ a_i\in GF(2^n), a_2\neq 0.$$

Так как при нечетном n в поле $GF(2^n)$ из любого элемента извлекается однозначно кубический корень, то заменой $X=a_2^{-1/3}$ последнее уравнение можно свести к виду

$$Y^2+Y=X^3+a_4X+a_6,\ a_i\in GF(2^n).$$

Упражнение 5.2.3 Докажите это.

Указание. Достаточно доказать существование кубического корня. При нечетном n $2^n - 1 = 3k + 1$, поэтому при m не кратном 3 согласно теореме Ферма

$$\alpha^m = \alpha^{m+3k+1} = \alpha^{m+6k+2} = \alpha^{3l}.$$

Остается воспользоваться существованием примитивного элемента. Единственность вытекает из того, что уравнение $x^3 = 1$ имеет единственный корень.

Дальнейшее упрощения уравнения данной кривой можно получить заменой переменных $Y = Y + kx + l, X = X + k^2$, если выбрать k, l так, чтобы $k^4 + k + a_4 \in GF(2), l^2 + l + k^6 + a_4k^2 + a_6 \in GF(2)$. Тогда уравнение примет вид

$$Y^2 + Y = X^3 + aX + b, a, b \in GF(2).$$

Упражнение 5.2.4 Проверьте это.

Для доказательства существования такой замены используем понятие следа. След элемента $y \in GF(2^n)$ в поле $GF(2)$ определяется формулой

$$Tr(y) = y + y^2 + y^4 + y^8 + \ldots + y^{2^{n-1}},$$

согласно тождеству Фробениуса линеен, удовлетворяет тождеству

$$(Tr(y))^2 = Tr(y^2) = Tr(y)$$

и всегда $Tr(y) \in GF(2)$. (См. 1.2).

Упражнение 5.2.5 Докажите, что уравнение $y^2 + y = a$ разрешимо в поле $GF(2^n)$ если и только если $Tr(a) = 0$.

При нечетном n решение можно найти в виде

$$y = a + a^4 + a^{16} + a^{64} + \ldots + a^{2^{n-1}}.$$

Второе решение получается прибавлением единицы.

Указание. В силу линейности $Tr(a) = Tr(y^2 + y) = Tr(y^2) + Tr(y) = 0$. Согласно тождествам Фробениуса и Ферма

$$y^2 = a^2 + a^8 + a^{32} + a^{128} + \ldots + a^{2^n} = a + a^2 + a^8 + a^{32} + a^{128} + \ldots + a^{2^{n-2}},$$

откуда

$$y^2 + y = a + (a + a^2 + a^4 + a^8 + \ldots + a^{2^{n-1}}) = a + Tr(a) = a.$$

Упражнение 5.2.6 Докажите, что уравнение $y^4 + y = a$ разрешимо в поле $GF(2^n)$ если и только если $Tr(a) = 0$.

Указание. Необходимость получается также, как и выше, если заметить, что $Tr(y^4) = Tr(y^2) = Tr(y)$. Для доказательства достаточности вначале решим уравнение $x^2 + x = a$ и выберем то его решение, у которого $Tr(x) = 0$ (если $Tr(x) = 1$, то $Tr(x+1) = Tr(x) + Tr(1) = 0$, так как при нечетном n очевидно $Tr(1) = 1$). Потом решим уравнение $y+y^2 = x$ и получим $a = y+y^2+(y+y^2)^2 = y + y^4$.

Упражнение 5.2.7 Докажите, что при нечетном n для любого $a \in GF(2^n)$ существуют $y \in GF(2^n)$ такое, что $y^2 + y + a \in GF(2)$ и такое $z \in GF(2^n)$, что $z^4 + z + a \in GF(2)$.

Указание. Так как $Tr(a) = a$ при $a \in GF(2)$ то в случае $Tr(a) = 1$ очевидно $Tr(a+1) = Tr(a) + Tr(1) = 0$.

Теперь уже можно найти такое k, что $k^4 + k + a_4 \in GF(2)$, а потом такое l, что $l^2 + l + k^6 + a_4k^2 + a_6 \in GF(2)$.

Очевидно, что кривые $Y^2 + Y = X^3, Y^2 + Y = X^3 + 1$ изоморфны.

Упражнение 5.2.8 Проверьте, что $Y = Y + X, X = X + 1$ — изоморфизм.

Поэтому при нечетном n имеется 3 класса неизоморфных суперсингулярных эллиптических кривых (согласно [130] при четном n имеется 7 классов), стандартными представителями которых являются кривые

$$\mathcal{E}_1 : Y^2 + Y = X^3, \;\; \mathcal{E}_2 : Y^2 + Y = X^3 + X$$

и

$$\mathcal{E}_3 : Y^2 + Y = X^3 + X + 1.$$

При нечетном n число точек для первой кривой равно 2^n+1 и $2^n \pm \sqrt{2^{n+1}}+1$ для второй и третьей (знак + или − выбирается в зависимости от кривой и от сравнения n по модулю 8):

Кривая	n	Порядок группы
$y^2 + y = x^3$	нечетное	$2^n + 1$
$y^2 + y = x^3 + x$	$n \equiv 1, 7 (\mathrm{mod} 8)$	$2^n + 1 + 2^{(n+1)/2}$
$y^2 + y = x^3 + x$	$n \equiv 3, 5 (\mathrm{mod} 8)$	$2^n + 1 - 2^{(n+1)/2}$
$y^2+y = x^3+x+1$	$n \equiv 1, 7 (\mathrm{mod} 8)$	$2^n + 1 - 2^{(n+1)/2}$
$y^2+y = x^3+x+1$	$n \equiv 3, 5 (\mathrm{mod} 8)$	$2^n + 1 + 2^{(n+1)/2}$

Указанные значения легко вычисляются с использованием теоремы Хассе-Вейля.

Отметим, что группы этих кривых при нечетном n являются циклическими.

Пример 5.2.3 Найдем порядок группы эллиптической кривой в случае $\mathcal{E}_2$. Рассматривая $\mathcal{E}_2$ над полем $GF(2)$, имеем на ней точки (0,0),(0,1),(1,0),(1,1) и еще нулевой элемент O – всего пять элементов. циклической группы. Таким образом, $q = 2, N = 5, t = -2$, и мы находим корни квадратного уравнения $x^2 + 2x + 2 = 0$ в стандартной декартовой форме $\alpha = -1 + \sqrt{i}$ и $\beta = -1 - \sqrt{i}$ или в тригонометрической форме

$$\alpha = \sqrt{2}\left(\cos\frac{3\pi}{4} + i\sin\frac{3\pi}{4}\right)$$

и

$$\beta = \sqrt{2}\left(\cos\frac{5\pi}{4} + i\sin\frac{5\pi}{4}\right).$$

По теореме Хассе-Вейля получим

$$N(n) = 2^n - 2^{\frac{n}{2}}\left(\cos\frac{3\pi n}{4} + i\sin\frac{3\pi n}{4}\right) - 2^{\frac{n}{2}}\left(\cos\frac{5\pi n}{4} + i\sin\frac{5\pi n}{4}\right) \equiv$$

$$2^n + 1 - 2\cdot 2^{\frac{n}{2}}\cos\frac{3\pi n}{4},$$

поскольку

$$\sin\frac{5\pi}{4} = \sin\left(-\frac{3\pi}{4}\right) = -\sin\frac{3\pi}{4}.$$

Для $\cos\frac{3\pi n}{4}$ имеем
$\cos\frac{3\pi n}{4} = \frac{-\sqrt{2}}{2}$, если $n \equiv 1(\text{mod } 8)$ или $n \equiv 7(\text{mod } 8)$,
$\cos\frac{3\pi n}{4} = \frac{\sqrt{2}}{2}$, если $n \equiv 3(\text{mod } 8)$ или $n \equiv 5(\text{mod } 8)$
(напомним, что n – нечетное).

Окончательно получаем

$$N(n) = 2^n + \sqrt{2^{n+1}} + 1,$$

$$\text{если } \ n \equiv 1 \ \text{ или } \ 7 \ (\text{mod } 8),$$

$$N(n) = 2^n - \sqrt{2^{n+1}} + 1,$$

$$\text{если } \ n \equiv 3 \ \text{ или } \ 3 \ (\text{mod } 8).$$

Тот факт, что порядок кривой $Y^2 + Y = X^3$ при нечетном n равен $2^n + 1$, можно доказать и без теоремы Хассе-Вейля.

Упражнение 5.2.9 Докажите это.

Указание. Уравнение $Y^2 + Y = a$ разрешимо только при $Tr(a)$ и имеет 2 различных корня. Ровно половина элементов поля $GF(2^n)$ имеет нулевой след (так как $Tr(x) = 0 \leftrightarrow Tr(x+1) = 1$). При нечетном n каждый элемент поля имеет единственный кубический корень.

Также элементарно можно доказать, что порядок суперсингулярных кривых нечетен, а несуперсингулярных кривых — четен.

Упражнение 5.2.10 Докажите это.

Указание. У несуперсингулярных кривых существует точка порядка два вида $(0, y)$. Поэтому по теореме Лагранжа их порядок четен. В суперсингулярной кривой пары противоположных элементов имеют вид $(x, y), (x, y+1)$ и их элементы всегда различны, значит число конечных точек у нее четно.

Особый интерес будут представлять такие n, для которых соответствующий порядок группы $\mathcal{EK}$ при разложении на простые множители содержит большое простое число (кстати, большое простое с N цифрами в десятичном представлении будем записывать как PN).

Ниже приведены некоторые конкретные значения, которые можно использовать для имплементации.

n	Кривая	Порядок группы
173	$\mathcal{E}_2$	$5 \cdot 13625405957 \cdot P42$
173	$\mathcal{E}_3$	$7152893721041 \cdot P40$
191	$\mathcal{E}_1$	$3 \cdot P58$
191	$\mathcal{E}_2$	$5 \cdot 3821 \cdot 89618875387061 \cdot P40$
191	$\mathcal{E}_3$	$25212001 \cdot 5972216269 \cdot P41$
239	$\mathcal{E}_2$	$5 \cdot 77852679293 \cdot P61$
239	$\mathcal{E}_3$	$P72$
251	$\mathcal{E}_1$	$3 \cdot 238451 \cdot P70$
323	$\mathcal{E}_3$	$137 \cdot 953 \cdot 525313 \cdot P87$

Другим достоинством суперсингулярных кривых является простота формул сложения для них, и особенно формул удвоения. (См. 5.1).

Заметим, что при сложении точек все же выполняется одна инверсия, которая является самой трудоемкой операцией в арифметике конечного поля.

Для координатного представления нулевого элемента $\mathcal{O}$ группы $\mathcal{E}_K$ фиксируем какое-нибудь координатное представление, задающую точку, не лежащую на кривой, например, $(0,0)$ для E_3 (для нулевого элемента E_2 придётся использовать специальное обозначение, например, (a,a), поскольку все точки из $(x,y) \in \{0,1\}^2$ принадлежат этой кривой.)

Отмеченное в начале данного параграфа удобство использования суперсингулярных эллиптических кривых не может скомпенсировать их глпаный недостаток: лля таких кривых известно [?] сведение задачи вычисления дискретного логарифма к аналогичной задаче для конечных полей с повышением размерности поля в некоторую константу k раз, зависящую от класса кривой. Это сведение выполняется вероятностным полиномиальным относительно $\log_2 q$ алгоритмом.

Так, для суперсингулярной эллиптической кривой $\mathcal{E}_1$, заданной над полем порядка q, задача вычисления дискретного логарифма в группе этой эллиптической кривой сводится к соответствующей задаче вычисления в том же поле порядка q, так как $k = 1$. Значит, использование этой кривой не более надежно, чем использование обычной криптосистемы в том же поле. Для кривых $\mathcal{E}_2$ и $\mathcal{E}_3$ такое сведение в случае нечетного n приводит уже к полям порядка q^4. Но четное n использовать нельзя, так как в этом случае константа $k = 1$.

Несмотря на то, что для полей $GF(2^n)$ известны специальные алгоритмы вычисления дискретных логарифмов (алгоритм Купершмита [92] сложности $exp(1.35n^{1/3}(\ln n)^{2/3})$), тем не менее они остаются практически неосуществимыми для полей размерности 800 и более. Это оправдывает использование на практике кривых $\mathcal{E}_4$ и $\mathcal{E}_5$.

Еще одна приятная особенность суперсингулярных кривых заключается в

возможности Быстрого вычисление y-координаты точки такой кривой по x-координате путем решения квадратного уравнения в поле $GF(2^n)$.

Этот «трюк» [130] позволяет экономить объем информации, какой приходится обмениваться сторонам при выполнении криптографических протоколов. Вместо того, чтобы пересылать обе координаты точки кривой, можно послать только x координату и один бит, с помощью которого можно из двух точек (x, y) и $(x, y+1)$ данной кривой выбрать одну. Но для этого нужно уметь быстро вычислять y, т.е. решать квадратное уравнение $y^2 + y = f(x)$ в поле $GF(2^n)$. Вычисление $f(x)$ для любой из указанных выше кривых требует только одного умножения, одного возведения в квадрат и может быть, сложения. Сложение делается очень быстро в любом базисе, а для ускорения возведения в квадрат выбираем нормальный базис, если возможно, низкой сложности. Для решения квадратного уравнения $y^2 + y = a$ применяем полученную в упражнениях формулу

$$y = a + a^4 + a^{16} + a^{64} + \ldots + a^{2^{n-1}}.$$

Это можно делать при нечетном n. В случае четного n квадратное уравнение тоже решить можно, оно в нормальном базисе сводится к решению системы линейных уравнений над полем $GF(2)$, которая решается быстро, но такой простой явной формулы уже не получается. Заметим, что второе решение $y + 1$ получается из первого заменой всех битов на противоположные, так как 1 в нормальном базисе имеет только единичные координаты. Вычисление y по x в нормальном базисе тем самым выполняется с линейной сложностью как при программной, так и при схемной имплементации, причем в случае последней еще и с малой глубиной, не большей $\log_2 n + 1$.

Для несуперсингулярных кривых пришлось бы решать уравнение $y^2 + yx = f(x)$, которое заменой $y = xz$ сводится к уравнению $z^2 + z = f(x)/x^2$, но при этом пришлось бы выполнить дополнительно деление и умножение.

5.2.2 Несуперсингулярные кривые

Несуперсингулярная эллиптическая кривая над полем $GF(2^n)$ имеет уравнение

$$Y^2 + XY = X^3 + a_2X^2 + a_6.$$

Замена переменной $Y = Y + kX$, где $k^2 + k + a_2 = a_2'$ переводит эту кривую в кривую

$$Y^2 + XY = X^3 + a_2'X^2 + a_6.$$

Так как согласно предыдущим упражнениям уравнение $k^2 + k + a_2 = a_2'$ разрешимо если и только если $Tr(a_2) = Tr(a_2')$ то кривые, связанные таким соотношением, изоморфны. Можно доказать, что это условие и необходимо для изоморфизма. В случае нечетного n можно далее считать, что $a_2 \in GF(2)$.

Ранее (См. 5.1) мы видели, что формулы сложения для таких кривых чуть менее удобны для вычислений, а формулы удвоения — существенно менее удобны, чем такие же формулы для суперсингулярных кривых.

Для несуперсингулярной кривой формула вычисления обратной точки тоже чуть сложнее:

$$-P = -(x, y) = (x, y + x).$$

Несуперсингулярные эллиптические кривые представляют больший интерес с криптографической точки зрения, по сравнению с суперсингулярными. Желательность их использования стала ясной после появления работы [?] и со временем была закреплена во всех существующих криптографических стандартах. Для несуперсингулярных кривых не известны методы взлома, основанные на неэкспоненциальных алгоритмах. Однако для обеспечения максимальной безопасности необходимо соблюдать осторожность в выборе коэффициентов. Так, коэффициент a_6 не должен быть нулевым. Эти кривые надо выбирать так, чтобы их порядок не был гладким числом, как в следующих примерах.

Пример 5.2.4 Кривая

$$X^2 + XY = X^3 + 1$$

над полем $GF(2^{163})$ имеет порядок

$$2 \times 5846\ 00654\ 93236\ 11672\ 81474\ 17536\ 98448\ 34832\ 91185\ 74063.$$

Пример 5.2.5 Кривая $Y^2 + XY = X^3 + 1$ над полем $GF(2^{131})$ имеет порядок

$$4 \times 6805\ 64733\ 84187\ 69269\ 32320\ 12949\ 34099\ 85129.$$

Среди несуперсингулярных кривых для практической имплементации можно брать кривые

$$\mathcal{E}_1 : Y^2 + XY = X^3 + X^2 + 1$$

или семейство кривых

$$\mathcal{E}_2 : Y^2 + XY = X^3 + X^2 + \gamma,$$

где $\gamma^3 = \gamma + 1,\ \gamma \in GF(2^n)$.

Так, группа точек кривой $\mathcal{E}_1$ над полем $GF(2^{163})$ имеет порядок $2 \cdot P49$, а группа точек кривой $\mathcal{E}_2$ над $GF(2^{177})$ имеет порядок $10 \cdot P53$. Отметим также, что группы точек кривой $\mathcal{E}_1$ имеют порядки с большим простым числом и для случаев $n = 283, 311, 331, 347$ и 359.

Говорят, что неизоморфные кривые над полем $GF(2^n)$

$$Y^2 + XY = X^3 + a_1X^2 + b, Y^2 + XY = X^3 + a_2X^2 + b$$

являются *скручиванием друг друга*, если $Tr(a_1) \neq Tr(a_2)$.

Для доказанной ранее теоремы о скрученных кривых аналогом для характеристики два является

Теорема 5.2.1 *Сумма порядков двух скрученных кривых равна* $2^{n+1}+2$.

Доказательство. Можно без ограничения общности считать, что $a_1 = 0, Tr(a_2) = 1$. При $x = 0$ оба уравнения имеют по одной точке $(0, \sqrt{b}$, так как в поле $GF(2^n)$ квадратный корень всегда существует и определен однозначно.

Упражнение 5.2.11 Докажите это.

При $x \neq 0$ только одно из этих двух уравнений имеет корни, причем два разных y и $y + x$. Действительно, если бы существовали $x \neq 0, y, z$ такие, что

$$y^2 + xy = x^3 + b, z^2 + xz = x^3 + a_2x^2 + b,$$

то

$$a_2x^2 = z^2 + y^2 + xz + xy = (y + z)^2 + x(y + z),$$

откуда

$$a_2 = ((y + z)/x)^2 + (y + z)/x,$$

что противоречит неразрешимости уравнения $u^2+u = a$ при $Tr(a) = 1$. Поэтому общее число конечных точек у обеих кривых равно 2^{n+1}.

5.2.3 Существующие стандарты о выборе кривых для имплементации ECC криптосистем

Алгоритм цифровой подписи с использованием эллиптических кривых (ECDSA) принят и описан в различных стандартах. Среди них ANSI X9.62, FIPS 186-2 (NIST), IEEE 1363-2000 [110], ISO/IEC 14888-3 [111], ISO/IEC 15946-3 [112], SEC-1[?], SEC-2[?] и др.

Далее мы опишем основные рекомендации стандарта ANSI X9.62 ECDSA следуя статье [109].

К эллиптическим кривым предъявляются следующие требования.

1. Кривые рассматриваются или над простыми полями (порядок q которых равен простому числу p), или над полями характеристики два (у которых $q = 2^m$)

2. Для представления элементов поля используется либо стандартный базис, порождаемый трехчленом или пятичленом, либо гауссов нормальный базис (GNB).

3. Кривая E задается выбором двух элементов a, b поля $GF(q)$. В случае $p > 2$ она имеет вид $y^2 = x^3 + ax + b$, а в случае $p = 2$ вид $y^2 + xy = x^3 + ax^2 + b$. Таким образом, стандарт рекомендует только несуперсингулярные кривые.

4. На кривой выбирается точка $(x_G, y_G), x_G, y_G \in GF(q)$ простого порядка $n > 2^{160}, n > 4\sqrt{q}$, и вычисляется кофактор $h = |E(GF(q))|/n$.

В качестве кривых можно и удобно выбирать в случае $p = 2$ кривые, у которых a, b равны $0, 1$, но стандарт рекомендует все же случайные кривые, т.е. кривые со случайно выбранными a, b.

При этом рекомендуется использовать следующие алгоритмы генерации случайных кривых.

Алгоритмы генерации случайных кривых. 1) Случай $q = p$. Положим $t = \lfloor \log_2 p \rfloor, s = \lfloor (t-1)/160 \rfloor, v = t - 160 \cdot s$.

1. Выбираем произвольную строчку битов («зерно», из которого вырастает кривая) $seedE$ длиной $g \geq 160$ бит, и полагаем z, равным числу, двоичная запись которого совпадает с $seedE$.

2. Применяя к $seedE$ стандартную хеш-функцию SHA_1, вычисляем g-битовую строку $H = SHA_1(seedE)$. Выбирая в H v самых правых битов, получаем строку c_0 длиной v битов.

3. Заменяя в c_0 самый левый бит на 0, получаем строку W_0.

4. Для i от 1 до s делаем следующее:

4.1 полагаем s_i равной $g-$битной строке, являющейся двоичной записью числа $z + i \bmod 2^g$

4.2 вычисляем g-битовую строку $W_i = SHA - 1(s_i)$.

5. Полагаем битовую строку W равной конкатенации (произведению) битовых строк $W_i, i = 0, \ldots, s$, т.е. $W = W_0 \ldots W_s$.

6. Полагаем r, равным целому числу с двоичной записью W. Выполнение пункта 3 гарантирует, что $r < p$.

7. Если $r = 0$ или $4r + 27 \equiv 0 (\mathrm{mod}\ p)$, то возвращаемся к шагу 1.

8. Выбираем ненулевые $a, b \in GF(p)$ так, чтобы $rb^2 \equiv a^3 (\mathrm{mod} p)$. Например, можно взять $a = b = r$.

9. Полученная кривая есть $E : y^2 = x^3 + ax + b$.

Заметим, что условие невырожденности кривой $4a^3 + 27b^2 \not\equiv 0 (\mathrm{mod} p)$ гарантировано выполняется так как при $b \neq 0$, $r = a^3/b^2 \bmod p$ удовлетворяет условиям $r \neq 0, 4r + 27 \not\equiv 0 (\mathrm{mod} p)$. Имеется только две попарно неизоморфные кривые с одним и тем же r; эти кривые являются скрученными, и сумма их порядков равна $2 + 2p$; кривые с разными r неизоморфны друг другу. На шаге 8 поэтому есть по существу только еще одна возможность выбора a, b, кроме явно указанной.

2)Случай $q = 2^m$. Положим, как и выше, $s = \lfloor (t-1)/160 \rfloor, v = t - 160 \cdot s$.

1. Выбираем произвольную строчку битов $seedE$ длиной $g \geq 160$ бит, и полагаем z, равным числу, двоичная запись которого совпадает с $seedE$.

2. Вычисляем g-битовую строку $H = SHA_1(seedE)$. Выбирая в H v самых правых битов, получаем строку b_0 длиной v битов.

3. Заменяя в b_0 самый левый бит на 0, получаем строку W_0.

4. Для i от 1 до s делаем следующее:

4.1 полагаем s_i равной $g-$битной строке, являющейся двоичной записью числа $z + i \bmod 2^g$

4.2 вычисляем g-битовую строку $b_i = SHA - 1(s_i)$.

5. Вычисляем битовую строку $b = b_0 \ldots b_s$ и полагаем b равным соответствующему элементу поля $GF(q)$.

6. Если $b = 0$, то возвращаемся к шагу 1.

7. Выбираем произвольный $a \in GF(q)$.

8. Полученная кривая есть $E : y^2 + xy = x^3 + ax^2 + b$.

Две такие кривые $E_i : y^2 + xy = x^3 + a_i x^2 + b_i$ изоморфны, если $b_1 = b_2, Tr(a_1) = Tr(a_2)$, где

$$Tr(a) = a + a^2 + a^4 + a^8 + \ldots + a^{2^{m-1}} \in GF(2).$$

Поэтому на шаге 7 есть только две существенно разные возможности, например достаточно выбрать $a = 0$ или $a = \gamma$, $Tr(\gamma) = 1$.

Генерация криптографически надежных параметров кривых . Стандартом рекомендуется определенный алгоритм генерации надежных параметров кривых.

1. Выбираем случайную кривую $E(GF(q))$ алгоритмом, указанным выше.

2. Вычисляем ее порядок $N = |E(GF(q))|$.

3. Проверяем, делится ли N на ранее выбранное простое n ($n > 2^{160}, n > 4\sqrt{q}$). Если нет, то переходим к шагу 1.

4. Проверяем, что n не делит ни одно из чисел $q^k - 1, k = 1, \ldots 20$. Если нет, то переходим к шагу 1.

5. Проверяем, что $n \neq q$. Если нет, то переходим к шагу 1.

6. Выбираем произвольную точку $G' \in E(GF(q))$ и полагаем $G = (N/n)G'$. Повторяем, пока не получим $G \neq O$.

Шаг 3 необходим для того, чтобы кривая содержала точку G порядка n. Так как согласно теореме Хассе

$$(\sqrt{q} - 1)^2 \leq |E(GF(q))| \leq (\sqrt{q} + 1)^2,$$

то условие $n > 4\sqrt{q}$ означает, что n^2 не делит $|E(GF(q))|$ и поэтому кривая не содержит подгрупп порядка n^2, значит подгруппа порядка n на ней только одна. Так как $(\sqrt{q} + 1)^2 - (\sqrt{q} - 1)^2 = 4\sqrt{q} < n$, то существует единственное h, такое, что

$$(\sqrt{q} - 1)^2 \leq nh \leq (\sqrt{q} + 1)^2.$$

Точка $G \neq O$, построенная в шаге 6, имеет очевидно порядок n.

Согласно [121] случайную кривую с подходящими криптографическими свойствами над полем $GF(2^m)$ при m примерно 200 можно с помощью варианта SEA алгоритма сгенерировать за несколько часов.

Известен также другой метод генерации криптографически подходящих кривых— метод CM — комплексного умножения. Над полем $GF(p)$ его усовершенствования известны как Atkin-Morain метод [137], а над полем $GF(2^m)$ — как Lay-Zimmer метод [123]. Подробное его описание имеется в IEEE 1363-2000 [110].

Пусть $E(GF(q))$ кривая порядка N. Пусть известно разложение $4q - (q + 1 - N)^2 = DV^2$, где D свободно от квадратов. Тогда кривая E обладает *комплексным умножением на* D .

СМ метод вначале находит D и порядок $N = nh$, где n — простое число, и далее проверяет, что $q \neq n$ и n не делит числа $q^k - 1, k = 1, \ldots, 20$. Потом вычисляются коэффициенты кривой. При малых D СМ метод существенно быстрее алгоритма Schoof'a, но при больших D его эффективность сильно снижается.

Стандарты для кривых Коблица. Кривыми Коблица называются несуперсингулярные кривые с коэффициентами $0, 1$. Естественно только их привести в качестве примеров кривых, рекомендованных стандартами. Мы сделаем это только для случая $q = 2^m$. В этом случае NIST рекомендует 5 кривых в полях размерностей $m = 163, 233, 283, 409, 571$. Коэффициенты кривой обозначаются a, b, порядок кривой равен nh, где n — простое число, образующий элемент подгруппы порядка n обозначается (x_G, y_G), где x_G, y_G — элементы поля $GF(2^m)$, которые задаются двоичным вектором координат в данном базисе. Младшие координаты записываются справа, причем для краткости 32-битные блоки представляются в виде восьми 16-ричных цифр $0, 1, \ldots, 9, a, b, c, d, e, f$. Стандарт рекомендует два базиса — гауссов нормальный базис, который однозначно определяется заданием параметра k, и стандартный полиномиальный базис, который однозначно определяется своим неприводимым многочленом. В качестве таких многочленов выбирается трехчлен, а если неприводимого трехчлена не существует, то пятичлен, причем из множества таких пятичленов $1 + x^{d_1} + x^{d_2} + x^{d_3} + x^m$ выбирается тот, который имеет минимальный показатель степени $d_3 > d_2 > d_1$. Далее приводятся для примера две стандартные кривые. Кривая K-163.

```
m  163
a  1
b  1
n  5846006549323611672814741753598448348329118574063
h  2
Gaussov normal base, k=4.
x_G  0 5679b353 caa46825 fea2d371 3ba450da 0c2a4541
y_G  2 35b7c671 00506899 06bac3d9 dec76a83 5591edb2
Полиномиальный базис с неприводимым многочленом
f(x) = x^163+x^7+x^6+x^3+1
x_G 2 fe13c053 7bbc11ac aa07d793 de4e6d5e 5c94eee8
y_G 2 89070fb0 5d38ff58 321f2e80 0536d538 ccdaa3d9
```

Кривая K-233.

```
m  233
a  0
b  1
h  4
n  3450873173395281893717377931138512760570940988862252126328087024741343
Gaussov normal base, k=2.
```

```
x_G  0fd e76d9dcd 26e643ac 26f1aa90 1aa12978 4b71fc07 22b2d056 14d650b3
y_G  064 3e317633 155c9e04 47ba8020 a3c43177 450ee036 d6335014 34cac978
Полиномиальный базис с неприводимым многочленом f(x) = x^233+x^74+1
x_G  172 32ba853a 7e731af1 29f22ff4 149563a4 19c26bf5 0a4c9d6e efad6126
y_G  1db 537dece8 19b7f70f 555a67c4 27a8cd9b f18aeb9b 56e0c110 56fae6a3
```

5.3 Умножение точек суперсингулярных кривых

Алгоритмы умножения умножения точки P эллиптической кривой на числовую константу k (кратко - алгоритмы вычисления $k \cdot P$), они же — алгоритмы скалярного умножения точки, являются основными в арифметике эллиптических кривых. Некоторые из них уже рассмотрены в разделах, посвященных аддитивным цепочкам и алгоритмам возведения в степень в конечных полях и группах. Напомним, что в случае эллиптических кривых особенно удобно использовать описанный выше «трюк» с использованием уравновешенных систем счисления (имеющих отрицательные цифры).

Алгоритмической особенностью суперсингулярных эллиптических кривых является то, что удвоение точки для таких кривых выполняется существенно быстрее умножения, а при использовании нормальных базисов в поле — почти бесплатно. Поэтому при оценке сложности алгоритмов, основанных на аддитивных цепочках, можно учитывать только операции сложения, не являющиеся удвоениями (как и в алгоритмах экспоненцирования в нормальных базисах конечных полей). Используя описываемые ниже алгоритмы, минимизирующие число «неудваивающих» шагов в аддитивных цепочках, можно существенно ускорить вычисления в случае, когда точка P не известна заранее. Если же P известна заранее и у нас достаточно памяти для хранения предвычисленной таблицы, то надо применять другие алгоритмы, также описанные выше, но в них использование суперсингулярных кривых не дает существенного выигрыша.

5.3.1 Вычисление $k \cdot P$ методом аддитивных цепочек

Напомним предлагавшийся ранее метод, используя на этот раз аддитивную символику.

Чтобы вычислить точку $k \cdot P$, разложим k в системе счисления по основанию 2^m, используя отрицательные цифры, получим

$$k = \sum_{i=0}^{\lfloor n/m \rfloor} a_i 2^{mi},$$

вычислим и запомним все кратные $a_i P$ (достаточно вычислить все нечетные кратные $P, 3P, \ldots, (2^{m-1}-1)P$ с помощью поочередных удвоений и прибавлений

P), а потом вычисляем kP по схеме Горнера

$$kP = (\dots(a_{s-1}2^m + a_{s-2})2^m + \dots + a_1)2^m) + a_0)P =$$

$$= (\dots(a'_{s-1}2^{m+l_{s-1}} + a'_{s-2})2^{m+l_{s-2}} + \dots + a'_1)2^{m+l_1}) + a'_0 2^{l_0})P,$$

используя $s = \lfloor n/m \rfloor$ сложений-вычитаний с уже вычисленными точками и столько же умножений на 2^{m+l} при подходящем l.

В общем случае оценка сложности имеет вид

$$2^{m-1}(M + K) + Mn/m + nK,$$

где M и K — сложности сложения-вычитания и удвоения точек соответственно. Выбирая

$$m = \lfloor \log_2 n - \log_2 \log_2 n - \log_2 \log_2 \log_2 n \rfloor$$

получаем оценку сложности

$$nK + Mn/m + o((M + K)n/m).$$

Для конкретных k ее можно улучшить, увеличивая m, но при этом более аккуратно вычисляя a_iP, которых при удачном выборе m может оказаться гораздо меньше, чем 2^m. Можно написать программу выбора m с минимизацией количества различных a_i, причем за счет сдвига можно всегда считать, что двоичные записи чисел a_i всегда начинаются с единицы.

Приведем еще один способ вычисления kP. Разбивая все числа a_i на группы одинаковых, можно представить kP в виде

$$\sum_{j=1}^{d} a_{i_j} \sum_{i \in M_j} 2^i = \sum_{j=1}^{d} 2^{m_j} a_{i_j} b_j,$$

откуда применяя схему Горнера, получаем, что сложность kP оценивается сверху как $dM+Kn$ плюс сложность системы всех различных точек a_iP, плюс сумма сложностей вычисления точек b_jP_j. Далее проводим минимизацию по параметру m. Сложность вычисления каждой из точек b_jP_j вычисляется рекурсивно, также рекурсивно вычисляется сложность вычисления системы точек a_iP, если разложить все эти числа по схеме Горнера с одним и тем же параметром m.

На практике наиболее важен случай n порядка 200, тогда m можно выбирать 6,7,8 в худшем случае. Если же точка P выбирается заранее, все ее кратные $P, 2P, \dots, (2^m - 1)P$ можно заранее занести в память, и указанная выше проблема минимизации исчезает, тогда m можно выбрать порядка 16 (чтобы запоминаемые точки уместились в быструю кэш-память).

Заметим еще, что на практике ускорения вычислений можно также достичь, выбирая k (если это будет в нашей власти) с числом единиц в диапазоне $40 - -60$.

Для этого годится такой простой алгоритм. Число k разложим в двоичной системе:

$$k = \sum_{i=0}^{m-1} a_i 2^i, \qquad a_i \in \{0,1\},$$

где $m =]\log_2(k+1)[$.

Пусть $i_1, i_2, \ldots, i_t$ – индексы единичных компонент в наборе

$$a_0, a_1, \ldots, a_{m-1}, \quad i_1 < i_2 < \ldots < i_t.$$

Тогда

$$kP = \left(\sum_{i=0}^{m-1} a_i 2^{a_i \cdot i}\right) P = \left(\sum_{j=1}^{t} 2^{i_j}\right) \cdot P = \sum_{j=1}^{t} (2^{i_j} \cdot P). \tag{5.33}$$

Выше были приведены формулы для вычисления точки $2P$. Используя n раз эту формулу, можно получить $2^n P$.

Далее найдем последовательность $A_j = 2^{i_j} P, \quad i = 1, 2, \ldots, t$ по следующей схеме

$$A_1 = 2^{i_1} P,$$

$$A_j = 2^{i_j - i_{j-1}} A_{j-1}, \quad j = 2, 3, \ldots, t.$$

Сложив все полученные $A_j, \quad j = 1, 2, \ldots, t$, получим искомое произведение kP:

$$kP = \sum_{j=1}^{t} A_j.$$

Этот алгоритм использует не более $\log_2 t$ умножений многочленов на двойку и не более $40 - 60$ операций сложения многочленов.

Пример 5.3.1 . Пусть в поле $GF(4)$ с модулярным многочленом $f(x) = x^4 + x + 1$ требуется вычислить $10P$, если $P = (x_0, y_0) = \left(x^3 + x + 1, x + 1\right)$. Имеем следующее разложение числа 10 в двоичной системе счисления

$$10 = 2^3 + 2^1 = 1010_2.$$

Поэтому $t = 2, \quad i_1 = 1, i_2 = 3$. Вычислим $A_1, \quad A_1 = 2P$. Обозначим через (x_1, y_1) координаты точки A_1. Тогда, используя рассмотренные выше формулы, получаем

$$x_1 = x_0^4 + 1 = \left(x^3 + x + 1\right)^4 + 1 = x^3 + x^2,$$

$$y_1 = x_0^4 + 1 + y_0^4 = x^3 + x^2 + (x+1)^4 = x^3 + x^2 + x.$$

Далее найдем координаты (x_2, y_2) точки $A_2 = 2^2 A_1$ Для этого вычисляем

$$x_1' = x_1^4 + 1 = x^3 + x + 1,$$

$$y_1' = x_1' + y_1^4 = x^3 + x + 1 + x^3 + 1 = x,$$

откуда

$$x_2 = (x_1')^4 + 1 = \left(x^3 + 1\right)^2 + 1 = x^3 + x^2,$$

$$y_2 = x_2 + (y_1')^4 = x^3 + x^2 + x + 1.$$

Таким образом,

$$10P = A_1 + A_2 = \left(x^3 + x^2, x^3 + x^2 + x\right) + \left(x^3 + x^2, x^3 + x^2 + x + 1\right)$$

Следуя правилам сложения точек, получаем $10P = 0$, т.к. точки A_1 и A_2 имеют равные абсциссы, но разные ординаты.

5.3.2 Применение для эллиптических кривых быстрого алгоритма возведения многочлена в степень в случае использования стандартного базиса

По Алгоритму 5.5. операция удвоения точки $P = (x, y)$ эллиптической кривой $\mathcal{E}_4$ или $\mathcal{E}_5$ выполняется по формуле

$$2P = (x^4 + 1, x^4 + y^4) \tag{5.34}$$

и поэтому сводится к возведению в квадрат в конечном поле.

Более того, вычисление $2^{2n}P,\ n \geq 1$, можно выполнить по формуле

$$2^{2n}P = \begin{cases} (x^{2^{4n}}, y^{2^{4n}} + 1), & \text{если } n \text{ нечетно,} \\ (x^{2^{4n}}, y^{2^{4n}}), & \text{если } n \text{ четно.} \end{cases} \tag{5.35}$$

Поэтому вычисление точки 2^nP можно выполнить быстро с использованием описанных выше процедур.

Действительно,

$$2^1P = 2P = (x^{4^1} + 1, x^{4^1} + y^{4^1}) = (x^{2^{2\cdot 1}} + 1, x^{2^{2\cdot 1}} + y^{2^{2\cdot 1}}).$$
$$2^2P = 2 \cdot 2^1P = (x^{4^2}, y^{4^2} + 1) = (x^{2^{2\cdot 2}}, y^{2^{2\cdot 2}} + 1).$$
$$2^3P = 2 \cdot 2^2P = (x^{4^3} + 1, x^{4^3} + y^{4^3} + 1) = (x^{2^{2\cdot 3}} + 1, x^{2^{2\cdot 3}} + y^{2^{2\cdot 3}} + 1).$$
$$2^4P = 2 \cdot 2^3P = (x^{4^4}, y^{4^4}) = (x^{2^{2\cdot 4}}, y^{2^{2\cdot 4}}).$$

Пусть

$$\begin{array}{lll} 2^nP = (x^{2^{2\cdot n}} + 1, x^{2^{2\cdot n}} + y^{2^{2\cdot n}}), & \text{если} & (n) \equiv 1 \pmod 4. \\ 2^nP = (x^{2^{2\cdot n}}, y^{2^{2\cdot n}} + 1), & \text{если} & (n) \equiv 2 \pmod 4. \\ 2^nP = (x^{2^{2\cdot n}} + 1, x^{2^{2\cdot n}} + y^{2^{2\cdot n}} + 1), & \text{если} & (n) \equiv 3 \pmod 4. \\ 2^nP = (x^{2^{2\cdot n}}, y^{2^{2\cdot n}}), & \text{если} & (n) \equiv 0 \pmod 4. \end{array}$$

Тогда применяя формулу удвоения, получим

$$\begin{array}{lll} 2^{(n+1)}P = (x^{2^{2\cdot(n+1)}} + 1, x^{2^{2\cdot(n+1)}} + y^{2^{2\cdot(n+1)}}), & \text{если} & (n+1) \equiv 1 \pmod 4. \\ 2^{(n+1)}P = (x^{2^{2\cdot(n+1)}}).y^{2^{2\cdot(n+1)}} + 1), & \text{если} & (n+1) \equiv 2 \pmod 4. \\ 2^{(n+1)}P = (x^{2^{2\cdot(n+1)}} + 1, x^{2^{2\cdot(n+1)}} + y^{2^{2\cdot(n+1)}} + 1), & \text{если} & (n+1) \equiv 3 \pmod 4. \\ 2^{(n+1)}P = (x^{4^{(n+1)}}, y^{4^{(n+1)}}) = (x^{2^{2\cdot(n+1)}}, y^{2^{2\cdot(n+1)}}), & \text{если} & (n+1) \equiv 0 \pmod 4. \end{array}$$

Таким образом, формула (3.5) доказана индукцией по n.

5.3.3 Использование проективных координат

Если время выполнения операции инвертирования в данном поле в пять раз и более раз больше времени умножения, то ускорения в вычислении $k \cdot P$ можно добиться почти полностью исключив операцию инвертирования за счет увеличения в $4,5$ раза общего числа умножений, переходя к *проективным координатам*. До этого момента мы избегали ими пользоваться, но час настал.

Заметим, что путем подстановки

$$x = \frac{x}{z}, \qquad y = \frac{y}{z}$$

в уравнения, задающие эллиптические кривые E_1, E_2 и E_3 мы приходим к следующим однородными уравнениями относительно x, y и z:

$$E_1 : y^2z + yz^2 = x^3, \quad E_2 : y^2z + yz^2 = x^3 + xz^2, \quad E_3 : y^2z + yz^2 = x^3 + xz^2 + z^3.$$

Наряду с обычными (аффинными) координатами мы можем теперь рассматривать проективные координаты — ненулевые тройки из K^3. Проективные координаты (x, y, z) и (x', y', z') считаются эквивалентными, если для некоторого ненулевого t из K, выполняется

$$x' = tx, \quad y' = ty, \quad z' = tz.$$

Класс эквивалентности, порожденный тройкой (x, y, z) обозначаем $(x : y : z)$. Множество всех классов эквивалентности для данной тройки и называем ее *проективными* координатами. Геометрически (двумерное) *проективное пространство* можно представлять себе как множество прямых в обычном трехмерном пространстве, проходящих через начало координат.

Теперь эллиптическую кривую можно рассматривать на проективной плоскости как множество проективных точек, удовлетворяющих соответствующему однородному уравнению. Заметим, что единственной проективной точкой с нулевой z координатой, лежащей на эллиптической кривой, будет точка $(0 : 1 : 0)$, которая соответствует бесконечно удаленной точке O. Для остальных точек кривой $(x : y : z) \sim (x/z : y/z : 1)$, так что проективная точка $(x : y : z)$ однозначно соответствует аффинной точке $(x/z, y/z)$.

Мы хотим теперь получить формулы для сложения проективных точек P и Q.

Пусть $P = (x_1 : y_1 : 1) \in E_i)$ (таким образом, одна точка у нас фактически задана обычными координатами) и $Q = (x_2 : y_2 : z_2) \in E_i$. Предположим, что $P, Q \neq O$ и $P \neq Q$ (нас интересует основной случай в сложении точек). Для точки $R = P + Q$, $R = (x_3' : y_3' : 1)$ мы можем использовать формулы сложения в аффинном случае, после применения которых, учитывая что $x_2' = x_2/z_2, y_2' = y_2/z_2$ получим

$$x_3' = \frac{a^2}{b^2} + x_1 + \frac{x_2}{z_2},$$

$$y_3' = 1 + y_1 + \frac{a}{b}(\frac{a^2}{b^2} + \frac{x_2}{z_2}),$$

где

$$a = y_1 z_2 + y_2, \quad b = x_1 z_2 + x_2.$$

Полагая

$$z_3 = b^3 z_2, \quad x_3 = x_3' z_3, \quad y_3 = y_3' z_3$$

находим $R = (x_3 : y_3 : z_3)$, где

$$x_3 = a^2 b z_2 + b^4,$$

$$y_3 = (1 + y_1) z_3 + a^3 z_2 + a b^2 x_2,$$

$$z_3 = b^3 z_2.$$

Отсюда ясно, что, не считая сложений и возведений в степень, нам для сложения точек в проективных координатах необходимо выполнить 9 умножений (в аффинных координатах только 2), но зато ни одного деления. Умножения можно производить в следующем порядке:

$$y_1 z_2, x_1 z_2, b z_2, z_3 = b^2(b z_2), a^2(b z_2), (1+y_1) z_3, a^2 z_2, b^2 x_2, a^3 z_2 + a b^2 x_2 = a(a^2 z_2 + b^2 x_2).$$

При вычислении kP мы последовательно удваиваем точки (что не требует деления), а затем складываем некоторые из них, накапливая результат в Q. Окончательный результат, полученный в проективных координатах, преобразуем в аффинные делением на z_3^{-1}.

5.3.4 Метод Монтгомери

Это метод вычисления $k \cdot P$ с использованием минимальной дополнительной памяти (минимального числа регистров), предложенный в [135]. В данном случае не следует искать сходства с методом Монтгомери ускорения модулярного экспоненцирования. Он является некоторой модификацией обычного бинарного метода, в котором k представляется в двоичном виде $k = (k_{l-1} \dots k_0)_2$ и последовательно вычисляются точки данной кривой $m_i P$, где $m_i = (k_{l-1} \dots k_{l-i})_2$. Но в отличие от бинарного метода вместо рекурсии

$$P_1 = 2P_1 \text{если } k_j = 0, \text{и} P_1 = P_1 + P \text{если } k_j = 1$$

применяется рекурсия

$$P_2 = P_2 + P_1, P_1 = 2P_1, \text{если } k_j = 0, \text{и} P_1 = P_2 + P_1, P_2 = 2P_2, \text{если } k_j = 1,$$

которая программно реализуется с помощью цикла при начальных установках $P_1 = P, P_2 = 2P$. По индукции легко проверяется, что инвариантом цикла является соотношение $P_2 = P_1 + P$, и на i шаге $P_1 = m_i P$. Этот алгоритм кажется более затратным, чем обычный бинарный, но Монтгомери заметил, что на каждом шаге цикла при вычислении новых значений x-координат

точек P_i можно использовать старые значения тоже только x координат, и x координату точки P, не меняющуюся во время цикла. Действительно, на каждом шаге цикла производятся вычисления $Q_2 = Q_2 + Q_1, Q_1 = 2Q_1$, где $Q_i = Q_j + P, i \neq j$, и согласно правилам сложения и удвоения для суперсингулярных кривых $y^2 + y = x^3 + x(+1)$ координаты точек Q_i, P связаны соотношениями

$$x_3 = \left(\frac{y_1 + y_2}{x_1 + x_2}\right)^2 + x_1 + x_2,$$

$$x_4 = x_1^4 + 1.$$

Но так как, если например для новых точек $P = Q_2 - Q_1$, то, учитывая что $-Q_1 = (x_1, y_1 + 1)$, имеем

$$x = \left(\frac{y_1 + y_2 + 1}{x_1 + x_2}\right)^2 + x_1 + x_2,$$

откуда

$$x_3 = x + \frac{1}{(x_1 + x_2)^2}.$$

Во втором случае ($P = Q_1 - Q_2$) получается точно такая же формула. Поэтому вычисление новых значений x координат можно выполнять по следующим формулам

$$x_4 = x_1^4 + 1,\ x_3 = x + \frac{1}{(x_1 + x_2)^2}$$

с помощью одного инвертирования, трех возведений в квадрат и двух сложений (в случае схемной реализации фактически одного сложения). Что касается y-координат, то они в цикле вообще не вычисляются. Только после того как $P_1 = kP, P_2 = (k+1)P$ вычисляется y координата y_1 точки $kP = P_1$ с помощью формул сложения $P_2 = P_1 + P$

$$x_2 = \left(\frac{y_1 + y}{x_1 + x}\right)^2 + x_1 + x,$$

из которых следует, что

$$x_2(x_1 + x)^2 = (y_1 + y)^2 + (x_1 + x)^3,$$

$$y_1^2 + y^2 = (x_1 + x)^2(x_1 + x_2 + x) = (x_1^2 + x^2)(x_1 + x_2 + x) =$$
$$x_1^3 + x_2^3 + x_1^2 x_2 + x_2^2 x_1 + x(x_1^2 + x_2^2),$$

а так как в силу принадлежности точек данной кривой

$$y_1^2 + y_1 + x_1^3 + x_1 = y^2 + y + x^3 + x,$$

то

$$y_1 + y = x_1^2 x_2 + x_2^2 x_1 + x(x_1^2 + x_2^2) + x + x_1 = x_1^2(x + x_2) + (x_2^2 + 1)(x + x_1).$$

Поэтому для для вычисления y_1 достаточно сделать два умножения, два возведения в квадрат и пять сложений (в случае схемной реализации фактически четыре). В случае программной реализации кроме регистров для хранения x, y, x_1, x_2 нужны еще два вспомогательных регистра. Они используются только при вычислении y_1 в конце работы программы. Если пренебречь сложностью выполнения сложения и квадрирования (возведения в квадрат), то программная сложность скалярного умножения точек суперсингулярной кривой методом Монтгомери оценивается как

$$L(k) = \lfloor \log_2 k \rfloor L(I(n)) + 2L(M(n))),$$

где $L(I(n)), L(M(n))$ — сложность инвертирования и умножения в поле $GF(2^n)$. Эта оценка лучше, чем оценка сложности бинарного метода в худшем случае, но она может быть хуже, в случае когда k имеет малое число $\nu_2(k)$ единиц в двоичной записи. В сравнении с оценкой худшего случая при использовании бинарного метода с применением уравновешенной двоичной системы метод Монтгомери вероятно проигрывает, если время инвертирования сильно превосходит время умножения в поле. Но зато он использует меньше пересылок во время работы, что отчасти компенсирует большее число операций в цикле. Заметим, что для имплементации разных вариантов бинарного метода нужно не менее семи регистров.

В случае схемной реализации сложность универсальной схемы экспоненциации и для разных вариантов бинарного метода не зависит от $\nu_2 k$. В этом случае метод Монтгомери всегда превосходит простой бинарный метод. Глубина схемы для скалярного умножения $k \cdot P$ с входами

$$(k, x, y), x = (x_1, \ldots, x_n), y = (y_1, \ldots, y_n), k = (k_{l-1}, \ldots, k_0),$$

оценивается сверху как

$$l(D(I(n)) + 4) + D(M(n)) + 4,$$

где $D(I(n)), D(M(n))$ — глубина инвертирования и умножения в поле $GF(2^n)$. Эта оценка также лучше оценки глубины схемы, построенной простым бинарным методом. Но оценка глубины для бинарного уравновешенного метода имеет вид

$$l(D(I(n)) + 2D(M(n)) + 6)/2 + 1$$

и при $n \geq 16$ будет меньше. При автоматной реализации схема Монтгомери будет иметь меньшую сложность, но работать медленнее.

Упражнение 5.3.1 Попробуйте применить метод Монтгомери при выборе проективных координат.

5.4 Умножения точек несуперсингулярных кривых

Описанные выше алгоритмы для суперсингулярных кривых можно применять и для несуперсингулярных кривых, заменив лишь правила удвоения и сложения точек. Но эти правила чуть сложнее, и работать соответствующие алгоритмы будут медленнее. А так как в несуперсингулярном случае удвоение делается лишь чуть медленнее сложения, то алгоритмы, минимизирующие число сложений, не будут давать существенного ускорения. Поэтому в этом случае возникает необходимость модернизации старых алгоритмов и создания новых.

5.4.1 Метод Монтгомери для несуперсингулярных кривых

Это метод был развит в работе латиноамериканских криптографов Лопеса и Дахаба [126]. Идея такая же, как и в суперсингулярном случае. Рассматриваем произвольную кривую $y^2 + xy = x^3 + ax^2 + b$. Также на каждом шаге цикла производим вычисления $Q_2 = Q_2 + Q_1, Q_1 = 2Q_1$, где $Q_i = Q_j + P, i \neq j$, и согласно правилам сложения и удвоения для несуперсингулярных кривых координаты точек Q_i, P связаны соотношениями

$$x_3 = \left(\frac{y_1 + y_2}{x_1 + x_2}\right)^2 + \frac{y_1 + y_2}{x_1 + x_2} + x_1 + x_2 + a,$$

$$x_4 = x_1^2 + b/x_1^2$$

(при $x_1 = 0$ в результате удвоения получается бесконечно удаленная точка).

Упражнение 5.4.1 Выведите последнее равенство из стандартного правила удвоения:

$$x_4 = \alpha^2 + \alpha + a,$$

где

$$\alpha = x_1 + \frac{y_1}{x_1}.$$

Указание: $y_1^2 + y_1 x_1 = x_1^3 + a x_1^2 + b$.
Заметим, что первое из них можно переписать в виде не содержащем коэффициент a

$$x_3 = \frac{x_2 y_1 + x_1 y_2 + x_1 x_2^2 + x_2 x_1^2}{(x_1 + x_2)^2}.$$

Упражнение 5.4.2 Проверьте это.

Указание:

$$y_1^2 + y_1 x_1 + y_2^2 + y_2 x_2 + x_1^3 + x_2^3 = a(x_1^2 + x_2^2) = a(x_1 + x_2)^2.$$

Но так как, если например для новых точек $P = Q_2 - Q_1$, то, учитывая что $-Q_1 = (x_1, y_1 + x_1)$, имеем

$$x = \frac{x_2(x_1 + y_1) + x_1 y_2 + x_1 x_2^2 + x_2 x_1^2}{(x_1 + x_2)^2}.$$

откуда

$$x_3 = x + \frac{x_2 x_1}{(x_1 + x_2)^2} = x + t^2 + t, t = \frac{x_1}{x_1 + x_2}.$$

Упражнение 5.4.3 Проверьте это.

Указание: $x_2 x_1 = x_1^2 + x_1(x_2 + x_1)$.

Поэтому вычисление новых значений x координат можно выполнять по следующим формулам

$$x_4 = x_1^2 + b/x_1^2, \quad x_3 = x + t^2 + t, t = \frac{x_1}{x_1 + x_2}$$

с помощью одного инвертирования, одного деления, одного умножения на константу b (для кривых Коблица она равна 1 и это умножение исчезает), двух квадрирований и четырех сложений; y-координаты в цикле не вычисляются, а находятся после его окончания, когда $P_1 = kP, P_2 = (k+1)P$. Вычисляется y координата y_1 точки $kP = P_1$ с помощью формул сложения $P_2 = P_1 + P$

$$x_2 = \left(\frac{y_1 + y}{x_1 + x}\right)^2 + \frac{y_1 + y}{x_1 + x} + x_1 + x + a,$$

которые согласно упражнению переписываются в виде

$$x_2(x_1 + x)^2 = xy_1 + x_1 y + x_1 x^2 + x x_1^2,$$

а потом в виде

$$xy_1 = x_2(x_1 + x)^2 + x_1 y + x_1 x^2 + x x_1^2 = x_2 x_1^2 + x_2 x^2 + x_1 y + x_1 x^2 + x x_1^2 =$$

$$= x_1(x_1 x_2 + x_1 x + x^2 + y^2) + x^2 x_2 =$$

$$= x_1(x_1 x_2 + x_1 x + x^2 + x x_2 + x^2 + y) + x(x_1 x_2 + x_1 x + x x_2 + y) + xy =$$

$$= (x + x_1)((x + x_1)(x + x_2) + x^2 + y) + xy,$$

откуда имеем

$$y_1 = (x + x_1)((x + x_1)(x + x_2) + x^2 + y)/x + y.$$

Для вычисления y_1 нужен еще один вспомогательный регистр s, с помощью которого y_1 вычисляется следующим образом:

$$s = x + x_1, y_1 = x + x_2, y_1 = y_1 s, y_1 = y_1 + y, t = x^2, y_1 = y_1 + t, y_1 = y_1 s, y_1 = y_1/x, y_1 = y_1 + y.$$

Для этого используется 5 сложений, одно квадрирование, 2 умножения и одно деление. Еще одно квадрирование и сложение используется при инициализации цикла с помощью равенств

$$x_1 = x, x_2 = x^2 + b/x^2.$$

Сложность скалярного умножения точек несуперсингулярной кривой методом Монтгомери оценивается как

$$L(k) = 2\lfloor \log_2 k \rfloor (L(I(n)) + 2L(A(n)) + L(S(n)) + L(M(n))) +$$
$$+2L(M(n))) + L(I(n)) + 6L(A(n)) + 2L(S(n)),$$

где $L(A(n)), L(S(n))$ — сложность квадрирования и сложения в поле $GF(2^n)$. Глубина схемы для скалярного умножения $k \cdot P$ с входами

$$(k, x, y), x = (x_1, \ldots, x_n), y = (y_1, \ldots, y_n), k = (k_{l-1}, \ldots, k_0),$$

оценивается сверху как

$$l(D(I(n)) + D(M(n)) + 4) + 2(D(M(n)) + D(I(n))) + 2.$$

5.4.2 Метод Монтгомери в проективных координатах

В этом случае x- координаты точек P_i представляются как X_i/Z_i. Тогда вычисление координат точки $2P_i$ производится по формулам

$$x = X_i^4 + bZ_i^4, z = Z_i^2 X_i^2,$$

действительно

$$\frac{x}{z} = \frac{X_i^4 + bZ_i^4}{Z_i^2 X_i^2} = \left(\frac{X_i}{Z_i}\right)^2 + b\left(\frac{Z_i}{X_i}\right)^2 = x_i^2 + \frac{b}{x_i^2}.$$

Вычисление x-координаты точки $P_1 + P_2$ производится по формулам

$$Z_3 = (X_1 Z_2 + X_2 Z_1)^2, X_3 = xZ_3 + (X_1 Z_2)(X_2 Z_1),$$

действительно

$$\frac{X_3}{Z_3} = x + \frac{X_1 Z_2 X_2 Z_1}{(X_1 Z_2 + X_2 Z_1)^2} =$$
$$= x + \frac{\frac{X_1}{Z_1}\frac{X_2}{Z_2}}{\left(\frac{X_1}{Z_1} + \frac{X_2}{Z_2}\right)^2} = x + \frac{x_1 x_2}{(x_1 + x_2)^2}.$$

Вычисление удвоения данной точки можно организовать с помощью одного вспомогательного регистра T_1 следующим образом:

$$T_1 = \sqrt{b} = b^{2^{n-1}}; X = X^2; Z = Z^2; T_1 = ZT_1, Z = ZX,$$
$$T_1 = T_1^2, X = X^2, X = X + T_1.$$

Упражнение 5.4.4 Проверьте это.

В вычислении используется 4 квадрирования, одно сложение и 2 умножения (одно из них на константу, равную единице для кривых Коблица).

Вычисление суммы двух точек можно организовать с помощью одного вспомогательного регистра T_1 следующим образом:

$$X_1 = X_1Z_2, Z_1 = Z_1X_2, T_1 = X_1Z_1, Z_1 = Z_1 + X_1, Z_1 = Z_1^2,$$

$$X_1 = Z_1x, X_1 = X_1 + T_1.$$

При этом используется одно квадрирование, 2 сложения и 4 умножения.

Вычисление аффинных координат (x, y) результата kP проводится после окончания цикла по его проективным координатам $(X_1 : Y_1 : Z_1)$следующим образом:

$$x_1 = X_1/Z_1; y_1 = (x + x_1)((X_1 + xZ_1)(X_2 + xZ_2) + (x^2 + y))(xZ_1Z_2)^{-1} + y.$$

Эти вычисления можно организовать с помощью двух вспомогательных регистров T_1, T_2 следующим образом:

$$T_1 = Z_1Z_2; Z_1 = Z_1x; Z_1 = Z_1 + X_1; Z_2 = Z_2x; X_1 = Z_2X_1; Z_2 = Z_2 + X_2;$$

$$Z_2 = Z_2Z_1; T_2 = x^2; T_2 = T_2 + y; T_2 = T_2T_1; T_2 = T_2 + y; T_1 = T_1x; T_1 = \frac{1}{T_1};$$

$$T_2 = T_2T_1; X_2 = X_1T_1; Z_2 = X_2 + x; Z_2 = Z_2T_2; Z_2 = Z_2 + y.$$

Упражнение 5.4.5 Проверьте это.

В вычислении используется одно инвертирование, одно квадрирование, 6 сложений и 10 умножений.

Сложность скалярного умножения точек несуперсингулярной кривой методом Монтгомери в проективных координатах оценивается как

$$L(k) = \lfloor \log_2 k \rfloor (3L(A(n)) + 5L(S(n)) + 6L(M(n))) +$$

$$+12L(M(n))) + L(I(n)) + 7L(A(n)) + 6L(S(n)),$$

где $L(A(n)), L(S(n))$ — сложность квадрирования и сложения в поле $GF(2^n)$. Глубина схемы для скалярного умножения $k \cdot P$ с входами

$$(k, x, y), x = (x_1, \ldots, x_n), y = (y_1, \ldots, y_n), k = (k_{l-1}, \ldots, k_0),$$

оценивается сверху как

$$l(2D(M(n)) + 3) + 5D(M(n)) + D(I(n)) + 2.$$

5.4.3 Метод Лопеса-Дахаба использования проективных координат

С целью уменьшения числа операций при применении проективных координат в [125] было предложено сопоставлять координатам $(X : Y : Z)$ аффинную точку $(X/Z, Y/Z^2)$. Тогда в этих проективных координатах кривая $y^2 + xy = x^3 + ax^2 + b$ представляется в виде

$$Y^2 + XYZ = X^3Z + aX^2Z^2 + bZ^4.$$

Упражнение 5.4.6 Проверьте это.

Указание: сделайте подстановку $x = X/Z, y = Y/Z^2$ и домножьте на Z^4.

Бесконечно удаленной точке соответствует точка $(1 : 0 : 0)$. Обратной для точки $(X : Y : Z)$ является точка $(X : XZ + Y : Z)$, так как ее аффинные координаты $(X/Z, (XZ + Y)/Z^2) = (X/Z, X/Z + Y/Z^2) = (x, x + y)$, где $(x, y) = (X/Z, Y/Z^2)$ — аффинные координаты точки $(X : Y : Z)$. Удвоение точки $2(x, y) = (x_1, y_1)$ в аффинных координатах выполняется по следующим правилам

$$x_1 = \alpha^2 + \alpha + a,$$
$$y_1 = y + \alpha(x + x_1) = x^2 + (\alpha + 1)x_1,$$

где

$$\alpha = x + \frac{y}{x}.$$

Выше уже проверялось, что первое равенство можно переписать в виде

$$x_1 = x^2 + b/x^2.$$

Упражнение 5.4.7 Проверьте, что удвоение точки $2(X : Y : Z) = (X_1 : Y_1 : Z_1)$ в рассматриваемых проективных координатах можно выполнить по следующему правилу:

$$Z_1 = X^2Z^2, X_1 = X^4 + bZ^4, Y_1 = bZ^4Z_1 + X_1(aZ_1 + Y^2 + bZ^4).$$

Указание: положим $x = X/Z, y = Y/Z^2$, тогда

$$x_1 = \frac{X_1}{Z_1} = x^2 + x^{-2}b,$$

$$y_1 = \frac{Y_1}{Z_1^2} = \frac{bZ^4}{Z_1} + x_1(a + \frac{Y^2}{Z_1} + \frac{bZ^4}{Z1}),$$

$$\frac{bZ^4}{Z_1} = \frac{Y^2}{X^2Z^2} + \frac{Y}{XZ} + \frac{X}{Z} + a = \frac{bZ^2}{X^2} = x^{-2}b = x^2 + x_1,$$

$$a + \frac{Y^2}{Z_1} + \frac{bZ^4}{Z1} = \frac{Y}{XZ} + \frac{X}{Z} = x + y/x = \alpha.$$

Сложение точек в проективных координатах задается слишком сложными формулами.

Упражнение 5.4.8 Получите эти формулы с использованием 14 операций умножения.

Однако, если одна из точек задана аффинными координатами, то как и в суперсингулярном случае формулы упрощаются.

Упражнение 5.4.9 Проверьте, что сложение точек $(X_1 : Y_1 : Z_1) + (X_2 : Y_2 : 1) = (X_3 : Y_3 : Z_3)$ в смешанных координатах может быть выполнено по следующему правилу:

$$A = Y_2 Z_1^2 + Y_1, B = X_2 Z_1 + X_1, C = Z_1 B, D = B^2(C + aZ_1^2), Z_3 = C^2,$$

$$E = AC, X_3 = A^2 + D + E, F = X_3 + X_2 Z_3, G = (X_2 + Y_2)Z_3^2, Y_3 = (E + Z_3)F + G.$$

Указание. Положим $x_1 = X_1/Z_1, y_1 = Y_1/Z_1^2, x_2 = X_2, y_2 = Y_2$ тогда

$$x_3 = \frac{X_3}{Z_3} = \frac{A^2 + D + E}{C^2} = \frac{A^2 + AC + B^2(C + aZ_1^2)}{C^2} =$$

$$= \frac{A^2}{C^2} + \frac{A}{C} + \frac{B^2}{C} + \left(\frac{BZ_1}{C}\right)^2 a,$$

$$\frac{A}{C} = \frac{Y_2 Z_1^2 + Y_1}{X_2 Z_1^2 + X_1 Z_1} = \frac{y_2 + y_1}{x_2 + x_1}, \frac{BZ_1}{C} = \frac{X_2 Z_1^2 + X_1 Z_1}{X_2 Z_1^2 + X_1 Z_1} = 1,$$

$$\frac{B^2}{C} == \frac{X_2^2 Z_1^2 + X_1^2}{X_2 Z_1^2 + X_1 Z_1} = \frac{X_2^2 + x_1^2}{X_2 + x_1} = \frac{(X_2 + x_1)^2}{X_2 + x_1} = X_2 + x_1 = x_2 + x_1,$$

откуда имеем

$$x_3 = \lambda^2 + \lambda + x_2 + x_1 + a, \lambda = \frac{y_2 + y_1}{x_2 + x_1},$$

$$y_3 = \frac{Y_3}{Z_3^2} = \frac{(X_3 + X_2 Z_3)(AC + Z_3) + (X_2 + Y_2)Z_3^2}{Z_3^2} = \left(\frac{X_3}{Z_3} + X_2\right)\left(\frac{AC}{Z_3} + 1\right) + X_2 + Y_2 =$$

$$(x_3 + x_2)\left(\frac{A}{C} + 1\right) + x_2 + y_2 = \lambda(x_3 + x_2) + x_3 + y_2.$$

Таким образом смешанное проективно-аффинное сложение точек выполняется с помощью 9 умножений (одно из них умножение на константу a, обычно равную 0 или 1), 9 сложений и 5 квадрирований. Удвоение требует 5 умножений (одно из них умножение на константу a, и одно из них умножение на константу b), 4 сложений и 5 квадрирований.

Как и в случае суперсингулярных кривых, этот метод можно сочетать с любой версией метода аддитивных цепочек, в которой строятся линейные цепочки. Вычисляемые в начале работы алгоритма и хранимые далее «малые» кратные точки P можно сразу вычислять в аффинных координатах, так как при вычислении в проективных координатах все равно потом результат придется переводить в аффинные координаты, затрачивая по два деления на запоминаемую точку. При вычислении же в аффинных координатах на каждую новую точку как правило требуется только одно инвертирование.

Если аддитивная цепочка для числа k содержит $\nu(k)$ удвоений и $\ell(k)$ сложений, то сложность соответствующего ей вычисления, если пренебречь сложностью сложений и квадрирований, а также сложностью перевода в аффинные координаты, равна

$$L(k) = 4\nu(k) + 8\ell(k),$$

а в случае кривых Коблица (когда $b = 1$)

$$L(k) = 3\nu(k) + 8\ell(k).$$

Поэтому в сравнении с суперсингулярным случаем сложность больше и минимизация числа общих сложений точек дает меньший эффект. При использовании обычного бинарного метода или бинарного метода с уравновешенной двоичной системой указанный метод проигрывает описанному выше методу Монтгомери, но при использовании 2^k-арных методов при $k \geq 4$ начинает его превосходить (за счет использования памяти, конечно).

5.4.4 Алгоритм скалярного умножения точек, использующий операцию «ополовинивания»

Этот алгоритм был предложен независимо Кнудсеном [114] и Шрепелем [152]. Он основан на том, что операция вычисления «половины» точки может быть выполнена быстрее ее удвоения, и в бинарном алгоритме (называемом иногда алгоритм «удвоения-сложения») можно заменить удвоение *ополовиниванием*. Так мы назовем операцию нахождения по точке P такой точки Q, что $2Q = P$. Можно эту точку обозначить $\frac{1}{2}P$. Если n — порядок точки P, то очевидно $\frac{1}{2}P = \frac{n+1}{2}P$ при нечетном n.

Но такой способ ополовинивания пригоден не всегда и неэффективен.

Эффективный способ основан на обращении операции удвоения. Действительно, если $2Q(u, v) = P(x, y)$, то

$$x = \lambda^2 + \lambda + a, y = u^2 + x(\lambda + 1), \lambda = u + v/u.$$

Для решения этой системы относительно u, v сначала находим λ из уравнения $\lambda^2 + \lambda = x + a$, потом $u^2 = y + x(\lambda + 1)$, извлекая корень находим u, а потом $v = \lambda u + u^2$. При этом используется 4 сложения (одно из них просто прибавление 1), 2 умножения, одно квадрирование, а вместо деления — извлечение квадратного корня и решение квадратного уравнения.

Далее мы покажем, что эти операции можно делать быстрее инвертирования. Но вначале объясним, как вместо операции удвоения применять операцию ополовинивания. Кратко напомним, как работает 2^w-арный алгоритм вычисления $k \cdot P$.

Представляем k в виде

$$\sum_{i=0}^{t} k_i 2^{l_i}, t \leq \frac{\log_2 k}{w} + 1,$$

где $|k_i| \leq 2^{w-1}$. Это представление можно получить из обычного 2^w-представления с цифрами $0, 1, \ldots, 2^w - 1$, заменяя по очереди каждую цифру на равную ей по модулю 2^w из интервала $-2^{w-1}, \ldots, 2^{w-1} - 1$ и прибавляя, если надо, единицу к следующей цифре. Далее можно, изменяя l_i, сделать так, чтобы все цифры стали нечетными. Для удобства используем запись вида

$$k = \sum_{i=0}^{l} k_i 2^i,$$

полагая в добавленных слагаемых $k_i = 0$.

Потом вычисляем и запоминаем точки $P_i = iP$, где $i = 1, 3, \ldots, 2^{w-1} - 1$. Полагаем $Q = O$ — нулевому элементу группы (бесконечно удаленной точке). Далее в цикле с обратным порядком изменения параметра i от $l-1$ до 0 выполняются следующие операции: если $k_i = 0$, то $Q = 2Q$, если $k_i > 0$, то $Q = Q + P_{k_i}$, если $k_i < 0$, то $Q = Q - P_{k_i}$.

Использования ополовинивания обосновывается следующими леммами.

Лемма 5.4.1 *Пусть для нечетного n дано описанное выше представление*

$$2^{t-1}k \bmod n = \sum_{i=0}^{t} k'_i 2^i, t = \lfloor \log_2 n \rfloor + 1.$$

Тогда

$$2k'_t + \sum_{i=0}^{t-1} k'_{t-i-1} 2^{-i} = k \bmod n,$$

дробь $a/2^l$ понимается как единственное число b в интервале от 0 до $n-1$, такое, что $2^l b = a \bmod n$. Кроме того, очевидно $k'_t = 0, 1$.

Доказательство. Заметим, что

$$a/2^l + b/2^l = (a+b)/2^l \bmod n, 2^s(a/2^l) = a/2^{l-s} \bmod n, (a/2^l)/2^s = a/2^{l+s} \bmod n.$$

Тогда деля обе части равенства

$$2^{t-1}k \bmod n = \sum_{i=0}^{t} k'_i 2^i, t = \lfloor \log_2 n \rfloor + 1.$$

на 2^{t-1}, получаем требуемое равенство.

Лемма 5.4.2 *Пусть порядок P нечетное число n. Тогда $\frac{1}{2}(a/2^l))P = (a/2^{l+1})P$, $\frac{1}{2}((a+b)P) = \frac{1}{2}(aP) + \frac{1}{2}(bP)$.*

Доказательство:

$$2(a/2^{l+1})P = (2(a/2^{l+1}) \bmod n)P = (a/2^l)P,$$

$$2(\frac{1}{2}(aP)+\frac{1}{2}(bP))=2(\frac{1}{2}(aP))+2(\frac{1}{2}(bP))=aP+bP=(a+b)P,$$

$$\frac{1}{2}(aP)+\frac{1}{2}(bP)=((a+b)/2)P.$$

Пусть $m_i = k'_0/2^i + \ldots + k'_i, i = 0, 1, \ldots$. Очевидно $m_{t-1} + 2k'_t = k \bmod n$. Тогда согласно лемме $m_{i+1}P = \frac{1}{2}(m_iP) + k'_{i+1}P$. Поэтому kP можно вычислить следующим алгоритмом, аналогичным вышеописанному. Вычисляем и запоминаем $P_i = iP$, $i = 1, 3, \ldots, 2^{w-1} - 1$. Полагаем $Q = O$. Далее в цикле с прямым порядком изменения параметра i от 0 до $t-1$ выполняются следующие операции: $Q = \frac{1}{2}Q$, если $k_i > 0$, то $Q = Q + P_{k_i}$, если $k_i < 0$, то $Q = Q - P_{k_i}$. После окончания цикла вычисляем Q = Q+2P, если $k'_t = 1$ и возвращаем Q в качестве результата.

Однако этот алгоритм непригоден для применения проективных координат. Аккумулирующая результат точка Q в случае их применения должна быть записана в проективных координатах, однако тогда не применим указанный выше способ ополовинивания. Для применения проективных координат удобен другой алгоритм, являющийся аналогом изложенного в разделе об аддитивных цепочках алгоритма Bricell at al [88].

В этом алгоритме вначале инициализируются точки $Q_i = O$, $i = 1, 3, \ldots, 2^{w-1} - 1$, потом полагаем $Q_1 = 2P$, если $k'_t = 1$, и в цикле с обратным изменением параметра от $t-1$ до 0 делаются следующие процедуры:

если $k'_i > 0$, то $Q_{k'_i} = Q_{k'_i} + P$,

если $k'_i < 0$, то $Q_{-k'_i} = Q_{-k'_i} - P$,

и потом всегда $P = \frac{1}{2}P$. После завершения цикла каждая точка Q_j будет равна

$$\left(\sum_{i,k'_i=\pm j} \pm 2^{t-i-1}\right) P.$$

После окончания работы цикла вычисляем

$$Q = \sum_{i=0}^{2^{w-2}-1} (2i+1)Q_{2i+1}.$$

Упражнение 5.4.10 Проверьте, что $Q = kP$.

Заметим, что количество операций ополовинивания равно $t = \log_2 n + O(1)$, причем часто эти операции выполняются блоками идущих подряд операций, а количество операций прибавления или вычитания P будет не больше $t/w + O(1)$.

Сумму Q можно вычислять следующим образом:

$$A_0 = Q_{2l+1}, A_1 = A_0 + Q_{2l-1}, \ldots, A_l = A_{l-1} + Q_1, Q = 2(A_0 + \ldots + A_{l-1}) + A_l, l = 2^{w-2} - 1.$$

Для этого требуется $2l = 2^{w-1} - 2$ сложений и одно удвоение.

Упражнение 5.4.11 Проверьте это.

В случае применения проективных координат в этом алгоритме точки Q_j задаются проективными, а точка P — аффинными координатами. В конце работы все точки складываются не в смешанных, а целиком в проективных координатах. Это медленнее, чем в смешанных координатах, но этих последних операций не более $2^{w-1}-2$ т.е. не более 14, так как в реальных вычислениях обычно $w \leq 5$. Можно однако для этих вычислений перейти к аффинным координатам.

Для быстрого выполнения ополовинивания нам надо уметь быстро быстро вычислять след, извлекать квадратные корни и решать уравнения вида $\lambda^2 + \lambda = a + x$. Ранее мы уже видели, что и то и другое очень быстро делается в нормальном базисе. В случае нечетного n для решения квадратного уравнения была даже дана явная формула. Но быстрое умножение в нормальных базисах удается выполнять не для всех размерностей n. Поэтому далее будет рассматриваться также случай выбора в поле стандартного базиса. Но вначале рассмотрим вопрос о выборе корня квадратного уравнения.

Для разрешимости квадратного уравнения необходимо и достаточно, чтобы след $Tr(x+a) = 0$. Так как $Tr(x+a) = Tr(x) + Tr(a) = 0$, то из разрешимости уравнения следует что $Tr(x) = Tr(a)$. Это верно для любой точки (x, y) нечетного порядка, потому что для любой такой точки ополовинивание существует. Согласно стандарту NIST выбираются кривые, у которых $Tr(a) = 1$. Далее мы предполагаем, что выбрана такая кривая.

Уравнение $\lambda^2+\lambda = a+x$ имеет два корня. Как выбрать нужный нам корень? Так как он удовлетворяет еще условию $u^2 = y + x(\lambda + 1)$, и точка (u, v) — половина точки (x, y), тоже имеет нечетный порядок, то $Tr(y + x(\lambda + 1)) = Tr(u^2) = Tr(u) = Tr(a) = 1$. В то же время согласно линейности следа $Tr(y + x\lambda)) = Tr(y + x(\lambda + 1)) + Tr(x) = 1 + Tr(x) = 0$.

Поэтому для проверки, что λ именно нужный нам корень, достаточно вычислит $t = y+x\lambda$ и проверить, что $Tr(y+x\lambda)) = 0$. Если это верно, то вычисляем $u = \sqrt{t + x}$, в противном случае вместо λ надо брать $\lambda + 1$, значит в качестве u нужно брать $u = \sqrt{t}$. После этого находим $v = \lambda u + u^2$. Но если нам аффинные координаты половинной точки сразу не нужны, так как после ее вычисления мы опять будем ополовинивать полученную точку, то вместо v можно использовать $\lambda(u) = u + v/u$, не вычисляя v. Далее нам понадобится $t = v + u\lambda'$, где λ' корень уравнения с правой частью $u + a$, но t можно вычислить по формуле

$$t = v + u\lambda' = u(u + u + v/u) + u\lambda' = u(u + \lambda' + \lambda(u)),$$

не зная v. Таким образом можно экономить вычисления при многократном повторном ополовинивании, что делает перспективным использование алгоритмов, основанных на аддитивных цепочках с малым числом сложений. На одно ополовинивание в среднем будет расходоваться одно умножение и два сложения, не считая извлечений корней и решения уравнений.

След в нормальном базисе очень легко вычисляется по формуле

$$Tr(c_1\alpha_1 + \ldots + c_n\alpha_n) = c_1 + \ldots + c_n,$$

так как после каждого возведения в квадрат координаты вектора $(c_1, \ldots, c_n)$ циклически сдвигаются, и след оказывается равен

$$(c_1 + \ldots + c_n)(\alpha_1 + \ldots + \alpha_n) = c_1 + \ldots + c_n.$$

Далее будем рассматривать вычисления в случае выбора в поле стандартного базиса.

Вычисление следа. Вычисление следа согласно его определению требует n квадрирований и $n-1$ сложение. В стандартном базисе это довольно дорого. Лучше воспользоваться свойством линейности

$$Tr(c_0 + c_1 z + \ldots + c_n z^n) = c_0 Tr(1) + \ldots + c_n Tr(z^n),$$

предварительно вычислив и запомнив $Tr(z^i) \in GF(2)$. Фактически можно вычислять след как линейный функционал. Часто многие из его коэффициентов равны нулю.

Пример 5.4.1 Выберем в поле $GF(2^{163})$ базис с неприводимым пятичленом $z^{163} + z^7 + z^6 + z^3 + 1$. Тогда $Tr(z^i) = 1$ только для $i = 0, 157$. Поэтому

$$Tr(c_0 + c_1 z + \ldots + c_n z^n) = c_0 + c_{157}.$$

Это еще проще, чем в нормальном базисе.

Решение квадратного уравнения. Пусть n нечетно. Назовем *полуследом*

$$H(x) = x + x^4 + x^{16} + \ldots + x^{2^{n-1}}.$$

Очевидно, это линейный оператор над полем $GF(2)$, т.е. $H(x+y) = H(x)+H(y)$.

Лемма 5.4.3 *$H(c)$ — это решение уравнения $x^2 + x = c + Tr(c)$, и $H(c) = H(c^2) + c + Tr(c)$ для любого $c \in GF(2^n)$.*

Доказательство. Очевидно согласно тождествам Фробениуса и Ферма

$$H(c^2) + H(c) = H(c)^2 + H(c) = (c^2 + c^8 + c^{32} + \ldots + c^{2^{n-2}} + c) +$$

$$+ (c + c^4 + c^{16} + \ldots + c^{2^{n-1}}) = c + Tr(c).$$

Так как нас интересуют уравнения $x^2 + x = c, Tr(c) = 0$, то одним из его решений является $H(c)$.

Способ вычисления полуследа на основе тождества линейности

$$H(c_0 + c_1 z + \ldots + c_n z^n) = c_0 H(1) + \ldots + c_n H(z^n),$$

очевидно в n раз менее эффективен, чем аналогичный способ вычисления следа. Так как полуслед — это линейный оператор, то несколько ускорить его вычисление можно с помощью алгоритма «четырех русских».

Но прежде, чем это делать, можно уменьшить вдвое объем используемой памяти и вдвое длину вектора, на который будет умножаться матрица, а значит и вдвое сложность вычисления. Правда, при этом будет вычисляться не всегда точно полуслед, иногда к нему будет прибавляться единица. Но так как нас интересует полуслед как формула для решения квадратного уравнения, то этот факт не будет иметь значения.

Идея состоит в том, чтобы использовать равенство

$$H(z^{2i}) = H(z^i) + z^i + Tr(z^i),$$

и заранее табулировать только $H(z^{2i+1}), i = 0, 1, \ldots$. Применяя индукцию, можно получить формулу

$$H\left(z^{2^j i}\right) = H(z^i) + z^{2^{j-1} i} + \ldots z^{2i} + z^i + jTr(z^i).$$

Упражнение 5.4.12 Проверьте это.

Указание: $Tr(z^2) = Tr(z)$.

Для вычисления полуследа тогда можно использовать формулу

$$H(c_0 + c_1 z + \ldots + c_n z^n) = c_0 H(1) + \ldots + c_n H(z^n) =$$

$$= c_0 H(1) + (c_1 + c_2 + c_4 + \ldots)H(z) + (c_3 + c_6 + c_{12} + \ldots)H(z^3) + \ldots + c_{n-2} H(z^{n-2}) +$$

$$(c_2 + c_4 + c_8 + \ldots)z + (c_4 + c_8 + \ldots)z^2 + (c_6 + c_{12} + \ldots)z^3 + \ldots c_{n-r} z^{(n-r)/2} + t,$$

где $t \in GF(2)$, $r = 1$ если $n \equiv 3 \pmod 4$, $r = 3$, если $n \equiv 1 \pmod 4$.

Упражнение 5.4.13 Проверьте это.

Указание. Примените предыдущее упражнение. в слагаемом t аккумулируются следы $Tr(z^i)$.

Таким образом, вычисление полуследа можно выполнить по формуле

$$H(c_0 + c_1 z + \ldots + c_n z^n) = C_0 H(z) + C_3 H(z^3) + \ldots + C_{n-2} H(z^{n-2}) +$$

$$+ s_1 z + \ldots + s_{(n-1)/2} z^{(n-1)/2} + t, t \in GF(2).$$

Обозначим s и C вектора длины n, дополненные нулями для тех индексов, которые не входят в формулу.

Эти вектора можно вычислить более просто, чем непосредственно по приведенной выше формуле, а именно можно в сумме

$$c_0 H(1) + \ldots + c_n H(z^n)$$

избавляться от четных слагаемых, применяя формулу

$$c_{2i}H(z^{2i}) = c_{2i}H(z^i) + c_{2i}z^i + c_{2i}Tr(z^i),$$

по очереди, начиная со старших индексов. Если в начале положить вектора s, C равными нулю, то указанная процедура их вычисления заключается в выполнении в цикле с обратным порядком изменения параметра i от $(n-1)/2$ до 1 присваиваний

$$C_i = C_i \oplus C_{2i}, s_i = s_i \oplus C_{2i}.$$

После окончания работы цикла получаются в точности те вектора, которые нам нужны.

Упражнение 5.4.14 Проверьте, хотя нам это не понадобится, что

$$t = s_0Tr(1) + s_1Tr(z) + \ldots + s_{n-1}Tr(z^{n-1}).$$

Если указанным способом построить схему в базисе $\oplus$, реализующую линейное преобразование вектора c в вектора C, s то ее сложность будет равна $n-1$, а глубина $\lceil \log_2(n-1) \rceil - 1$. Выходы этой схемы надо подать на входы схемы в том же базисе, реализующей линейное преобразование $(n-1)/2+1$-мерного вектора в n-мерный вектор.

Вместо формулы

$$H\left(z^{2^j i}\right) = H(z^i) + z^{2^{j-1}i} + \ldots z^{2i} + z^i + jTr(z^i)$$

можно использовать формулу

$$H\left(z^{2^j i}\right) = H(z^s r(z)) = H(z^{s+b_l}) + \ldots + H(z^{s+b_1}) + H(z^s),$$

где порождающий рассматриваемый базис неприводимый многочлен $f(z) = z^n + r(z)$, $r(z) = z^{b_l} + \ldots + z^{b_1} + 1$, и $n + s = 2^j i, 0 \leq s < n$. Поэтому вместо запоминания $H\left(z^{2^j i}\right)$ можно запоминать $H(z^{s+b_i}), i = 1, \ldots, l$ и $H(z^s)$.

Упражнение 5.4.15 Докажите эту формулу.

Указание:

$$z^{2^j i} = z^{n+s} = z^s(f(z) + r(z)) = z^s r(z) = z^{s+b_l} + \ldots + z^{s+b_1} + z^s.$$

Если $b_l = \deg r < n/2$ (а так обычно и бывает), то для любого i, $n/2 < i < n - \deg r$

$$H(z^i) = H(z^{2i}) + z^i + Tr(z^i) = H(r(z)z^{2i-m}) + z^i + Tr(z^i) =$$

$$H(z^{s+b_l}) + \ldots + H(z^{s+b_1}) + H(z^s) + z^i + Tr(z^i), s = 2i - m.$$

Так как $s+\deg r < i$, эту редукцию можно применять для исключения нечетных слагаемых $H(z^i)$ при $n/2 < i < n - \deg r$. После этого исключаются все четные слагаемые. Этим экономится еще память, и соответственно уменьшается число слагаемых до $n/4 + \deg r + O(1)$ в формуле для вычисления

$$H(c_0 + c_1 z + \ldots + c_n z^n) = s(z) + \sum_i C_i H(z^i).$$

Заметим, что сложность схемы исключения нечетных слагаемых оценивается сверху как $(l+2)(n/2-\deg r)/2$, а глубина — как $\log_2((n/2-\deg r)(l+2)) + O(1)$

Пример 5.4.2 В качестве примера опять рассмотрим неприводимый пятичлен $z^{163}+z^7+z^6+z^3+1$. Тогда исключение нечетных слагаемых с номерами $155-2i, i = 0, \ldots 36$ выполняется с помощью формулы

$$H(z^{155-2i}) = H(z^{310-4i}) + z^{155-2i} + Tr(z^{155-2i}) =$$

$$= H(z^{147-4i} z^{163}) + z^{155-2i} + Tr(z^{155-2i}) =$$

$$= z^{155-2i} + H(z^{147-4i}(z^7 + z^6 + z^3 + 1)).$$

Сложность соответствующей схемы равна $5 \cdot 37 = 185$. Потом, применяя схему сложности 162 исключаем все четные слагаемые. После этого остается построить схему для $(44, 163)$ линейного преобразования.

Однако число входов у подобной схемы можно и еще уменьшить. Так, еще до применения второй схемы, можно для $i = 0, \ldots, 5$ применять формулу

$$H(z^{51-2i}) = H(z^{102-4i}) + z^{51-2i} + Tr(z^{51-2i}) =$$

$$= H(z^{204-8i}) + z^{102-4i} + z^{51-2i} + Tr(z^{102-4i}) + Tr(z^{51-2i}) =$$

$$H(z^{41-8i} z^{163}) + z^{102-4i} + z^{51-2i} =$$

$$= H(z^{41-8i}(z^7 + z^6 + z^3 + 1)) + z^{102-4i} + z^{51-2i} =$$

$$H(z^{48-8i}) + H(z^{47-8i}) + H(z^{44-8i}) + H(z^{41-8i})) + z^{102-4i} + z^{51-2i}.$$

Сложность этой схемы $6 \cdot 5$. Если после ее применения применить схему сложности 162 для исключения всех четных слагаемых, число оставшихся слагаемых уменьшится до 38. Действуя так дальше, можно оставить только 21 слагаемое. К оставшейся схеме можно применить алгоритм «четырех русских». Какой из многих возникающих вариантов предпочтительнее, зависит от того, схему мы строим или пишем программу, от особенностей используемых компьютеров и т.д. Такие вопросы приходится решать экспериментально. В [95] утверждается, что время работы алгоритма примерно $2/3$ времени умножения.

Вычисление квадратных корней. Очевидный метод извлечения квадратных корней в поле $GF(2^n)$ основан на тождестве Ферма $\left(z^{2^{n-1}}\right)^2 = z$ и сводит задачу к $n-1$ квадрированию.

Более эффективный метод основан на свойстве линейности квадратного корня

$$\sqrt{c} = \sqrt{\sum_{i=0}^{n-1} c_i z^i} = \left(\sum_{i=0}^{n-1} c_i z^i\right)^{2^{n-1}}$$

$$= \sum_{i=0}^{n-1} c_i \left(z^{2^{n-1}}\right)^i.$$

Разделяя четные и нечетные слагаемые, имеем

$$\sqrt{c} = \sum_{i=0}^{(n-1)/2} c_{2i} \left(z^{2^{n-1}}\right)^{2i} + \sum_{i=0}^{(n-3)/2} c_{2i+1} \left(z^{2^{n-1}}\right)^{2i+1} =$$

$$\sum_{i=0}^{(n-1)/2} c_{2i} z^i + \sum_{i=0}^{(n-3)/2} c_{2i+1} z^{2^{n-1}} z^i = \sum_{i=0}^{(n-1)/2} c_{2i} z^i + \sqrt{z} \sum_{i=0}^{(n-3)/2} c_{2i+1} z^i.$$

Поэтому извлечение корня сводится к сложению и умножению на заранее вычисленный элемент $\sqrt{z}$, причем второй сомножитель имеет степень $(n-3)/2$. Схемная сложность получается примерно вдвое меньше сложности обычного умножения в поле. Тоже самое верно и для программной сложности. Заметим еще, что вычисление векторов коэффициентов обоих многочленов можно ускорить, применяя заранее вычисленные таблицы для аналогичных преобразований машинных слов.

Если используемый неприводимый многочлен есть трехчлен $f(z) = z^n + z^k + 1$, и n нечетно, то $\sqrt{z}$ можно выразить явно. А именно, если k нечетно, то $1 \equiv z^n + z^k (\mathrm{mod} f)$, откуда

$$\sqrt{z} = z^{(n+1)/2} + z^{(k+1)/2}.$$

Если же k четно, то $z^n \equiv z^k + 1 \ (\mathrm{mod} f)$, откуда

$$\sqrt{z} = z^{-(n-1)/2}(z^{k/2} + 1) \bmod f.$$

Для вычисления $z^{-s} \bmod f$ можно пользоваться равенствами $z^{-t} = z^{k-t} + z^{m-t} \bmod f, t \leq k$.

Пример 5.4.3 Пусть$f = z^{233} + z^{74} + 1$. Тогда

$$\sqrt{z} = z^{-116}(z^{37} + 1) \bmod f.$$

Так как $z^{-74} = 1 + z^{159}, z^{-42} = z^{32} + z^{191}$, то

$$z^{-116} = (1 + z^{159})(z^{32} + z^{191}) = z^{32} + z^{117} + z^{191} \bmod f,$$

$$\sqrt{z} = (z^{32} + z^{117} + z^{191})(z^{37} + 1) \bmod f.$$

После раскрытия скобок получается шестичлен.

Таким образом в приведенном примере сложность извлечения квадратного корня будет приблизительно в шесть раз больше сложности сложения и очевидно будет мала по сравнению со сложностью умножения. Поэтому средняя сложность ополовинивания приблизительно вдвое меньше сложности умножения (при использовании)нормальных базисов приблизительно равна сложности умножения). Если для вычисления kP применять описанный выше метод с использованием проективных координат, то его сложность приблизительно равна $2\log_2 k + (8\log_2 k)/w + 14(2^{w-1} - 2)$ умножений. На практике обычно $k < 2^{300}, w \leq 5$.

5.5 Умножение точек аномальных кривых

Аномальными бинарными кривыми называются кривые $E_0 : y^2+xy = x^3+1$, $E_1 : y^2 + xy = x^3 + x^2 + 1$. Они были предложены в [116] и иногда называются кривыми Коблица. Дальнейшее изложение основано, однако, на [157].

5.5.1 Свойства кривых Коблица

Легко проверить, что число точек в кривой $E_a(GF(2)) = 3 - \mu$, $\mu = (-1)^{1-a}$. При четном n эти кривые изоморфны, а при нечетном n они являются скручиванием друг друга.

Согласно теореме Хассе число точек на кривой $E_a(GF(2^n))$ равно

$$N(n) = 2^n + 1 - \alpha^n - \beta^n,$$

где α и β – корни квадратного уравнения $x^2-tx+2 = 0$, в котором коэффициент $t = q + 1 - N(1) = 3 - E_a(GF(2)) = \mu$. Корни уравнения $x^2 \pm x + 2 = 0$ в случае $a = 1$ $\alpha, \beta = (1 \pm i\sqrt{7})/2$, а в случае $a = 0$

$$\alpha, \beta = (-1 \pm i\sqrt{7})/2,$$

т.е. отличаются только знаком.

Известно, что последовательность $V_n = \alpha^n + \beta^n$ удовлетворяет рекуррентному соотношению $V_{n+1} - \mu V_n + 2V_{n-1} = 0$. Иногда такие последовательности называются последовательностями Люка.

Упражнение 5.5.1 Проверьте это.

Так как $V_0 = 2, V_1 = \alpha+\beta = \mu$, то эту последовательность можно вычислять с помощью указанного рекуррентного соотношения. Однако иногда проще это делать, пользуясь явной формулой. Например, при $a = 1$ применяя формулу Муавра имеем

$$V_n = \alpha^n + \beta^n = 2\sqrt{2}^n \cos(n \arccos(1/\sqrt{8})).$$

Упражнение 5.5.2 Проверьте, что при нечетных n последовательности V_n для $a = 0, 1$ противоположны друг другу, а при четных n совпадают.

Так как группа $E_a(GF(2))$ является подгруппой $E_a(GF(2^n))$, то порядок последней $N(n) = 2^n + 1 - V_n$ кратен $N(1) = 3 - \mu$. Для практического использования надо выбирать кривые у которых $r = N(n)/N(1)$ — простое число. Далее это мы будем предполагать.

Группы обоих кривых Коблица — циклические. Действительно, у кривой E_1 порядок равен удвоенному простому, и она циклическая как произведение циклических групп взаимно простых порядков.

Упражнение 5.5.3 Докажите это.

У кривой E_0 подгруппа $E_0(GF(2))$ четвертого порядка циклическая, так как $2(1,0) = (1,1)$, а в нециклической группе четвертого порядка все ненулевые элементы имеют порядок 2.

Поэтому согласно предыдущему упражнению группа $E_0(GF(2^n))$ тоже циклическая.

Поэтому в группе $E_a(GF(2^n))$ имеется циклическая подгруппа порядка r.

Упражнение 5.5.4 Докажите это.

Ее назовем *главной подгруппой.*

Упражнение 5.5.5 Точка P принадлежит главной подгруппе если и только если $P = N(1)Q$

Указание: воспользуйтесь свойствами циклических групп.

Есть более явный критерий принадлежности точки (x, y) главной подгруппе.

Упражнение 5.5.6 Точка (x, y) принадлежит главной подгруппе кривой E_1 если и только если $Tr(x) = 1$.

Упражнение 5.5.7 Точка (x, y) принадлежит главной подгруппе кривой E_0 если и только если $Tr(x) = 0$ и $Tr(y) = Tr(\lambda x)$, где $\lambda^2 + \lambda = x$.

Первое из них вытекает из более общего факта.

Упражнение 5.5.8 Точка (x, y) кривой $y^2 + yx = x^3 + ax^2 + b$ является удвоением другой точки если и только если $Tr(x) = Tr(a)$.

Указание. Согласно правилу удвоения $x = \lambda^2 + \lambda + a$, откуда

$$Tr(x) = Tr(\lambda^2) + Tr(\lambda) + Tr(a) = Tr(a).$$

Обратно, пусть $Tr(x) = Tr(a)$, тогда $Tr(a + x) = 0$, значит $\lambda^2 + \lambda = a + x$ при некотором λ. Возьмем x_1 такое, что $x(\lambda+1)+x_1^2 = y$. Тогда $x^2(\lambda^2+1)+x_1^4 = y^2$, $x^2(\lambda + 1) + x_1^2 x = yx$, значит

$$y^2 + xy = (\lambda^2 + \lambda)x^2 + x_1^4 + x_1^2 x = (a + x)x^2 + x_1^4 + x_1^2 x,$$

откуда $x_1^4 + x_1^2 x = b$. Положим $y_1 = \lambda x_1 + x_1^2$. Тогда

$$y_1^2 = \lambda^2 x_1^2 + x_1^4 = (\lambda + x + a)x_1^2 + x_1^4,$$

и $x_1 y_1 = \lambda x_1^2 + x_1^3$, откуда

$$y_1^2 + x_1 y_1 = (\lambda + x + a)x_1^2 + x_1^4 + \lambda x_1^2 + x_1^3 =$$

$$= (x_1^3 + ax_1^2) + xx_1^2 + x_1^4 = x_1^3 + ax_1^2 + b.$$

Второе тоже вытекает из более общего факта.

Упражнение 5.5.9 Точка (x,y) кривой $y^2+yx=x^3+ax^2+b$, $Tr(a)=0$, является учетверением другой точки если и только если $Tr(x)=0$ $Tr(y)=Tr(\lambda x)$, где $\lambda^2+\lambda=x+a$.

Указание. Пусть $(x,y)=4(x_1,y_1),(x_2,y_2)=2(x_1,y_1)$. Тогда $Tr(x)=Tr(x_2)=0$. Пусть $\lambda=x_2+y_2/x_2$, тогда $y=x_2^2+(\lambda+1)x,\lambda^2+\lambda=x+a$. Так как $Tr(x_2^2)=Tr(x_2)=0$, то $Tr(y)=Tr(\lambda x)$. Обратно, пусть $Tr(x)=0,Tr(y)=Tr(\lambda x),\lambda^2+\lambda=x+a$. Из $Tr(x)=0$ следует, что $(x,y)=2(x_2,y_2)$ для некоторой точки (x_2,y_2). Из формулы удвоения следует, что $\lambda=x_2+y_2/x_2$, $y=x_2^2+(\lambda+1)x$, $\lambda^2+\lambda=x+a$. Тогда $y+\lambda x=x_2^2+x$, значит

$$0=Tr(y)+Tr(\lambda x)=Tr(y+\lambda)=Tr(x_2^2+x)=Tr(x)+Tr(x_2^2)=Tr(x_2^2)=Tr(x_2).$$

Поэтому $(x_2,y_2)=2(x_1,y_1)$.

Очевидно, что если $(x,y)\in E_a(GF(2^n))$, то $(x^2,y^2)\in E_a(GF(2^n))$. (имеет место Эндоморфизм Фробениуса).

Упражнение 5.5.10 Проверьте это.

Указание: обе части возведите в квадрат, пользуясь тождеством Фробениуса.

Рассмотрим на группе $E_a(GF(2^n))$, отображение Фробениуса $\tau:(x,y)\to(x^2,y^2)$.

Упражнение 5.5.11 Проверьте, что $\tau(P_1+P_2)=\tau(P_1)+\tau(P_2)$.

Указание: рассмотрите случаи $P_1=P_2$ и $P_1\neq P_2$ и обе части возведите в квадрат.

Отображения с таким свойством называют *эндоморфизмами* кривой. Эндоморфизмы кривой можно складывать по формуле $(f_1+f_2)(P)=f_1(P)+f_2(P)$ и умножать по формуле $(f_1*f_2)(P)=f_1(f_2(P))$. Кроме эндоморфизма Фробениуса есть еще два тривиальных примера эндоморфизмов — нулевой и тождественный эндоморфизмы.

Упражнение 5.5.12 Проверьте, что сумма и произведение эндоморфизмов являются эндоморфизмами. Множество всех эндоморфизмов данной кривой образует кольцо.

Упражнение 5.5.13 Проверьте, что τ^n — тождественный эндоморфизм кривой $E(GF(2^n))$.

Известно, что $\tau^2+2=\mu\tau$, то есть для любой P $\tau(\tau(P))+2P=\mu\tau(P)$. Короткое доказательство этого тождества требует глубокого развития теории эллиптических кривых. Поэтому мы выполним прямую проверку. Сделаем ее в случае $a=0$, т.е. кривой $y^2+xy=x^3+1$. Тогда $\mu=-1$. Пусть P имеет координаты (x,y). Тогда очевидно

$$\tau(P)=(x^2,y^2),\mu\tau(P)=(x^2,x^2+y^2),\tau^2(P)=(x^4,y^4),$$

и согласно правилу сложения

$$2(x,y)=(x^2+x^{-2},(\frac{y}{x}+x)(x+x^2+x^{-2})+x^2+x^{-2}+y),$$

при $x = 0$ очевидно $y = 1$ и $2(x, y) = O$ — бесконечно удаленная точка. Нам надо проверить тождество

$$(x^4, y^4) + 2(x, y) = (x^2, x^2 + y^2).$$

При $x = 0$ оно очевидно, так как и слева и справа получается точка $(0, 1)$. Далее предполагаем, что $x \neq 0$. Сначала рассмотрим случай $x^4 = x^2 + x^{-2}$. Так как $x^6 + x^4 + 1 = (x^3 + x^2 + 1)^2$, то тогда $x^3 + x^2 + 1 = 0$,

$$(\frac{y}{x} + x)(x + x^2 + x^{-2}) + x^2 + x^{-2} + y = (\frac{y}{x} + x)(x + x^4) + x^4 + y =$$

$$= yx^3 + x^2 + x^5 + x^4 = (yx + x^3 + 1)x^2 + x^6 + 1 = y^2x^2 + x^6 + 1 = (yx + x^3 + 1)^2 = y^4,$$

значит $(x^4, y^4) = 2(x, y)$, и для сложения надо применять формулы удвоения. Этот случай возможен только при n кратном трем, так как корни уравнения $x^3 + x^2 + 1 = 0$ лежат в поле $GF(2^3)$ и порождают его, а оно содержится в поле $GF(2^n)$ только при указанном условии. В этом случае достаточно рассмотреть удвоение трех точек кривой $E_1(GF(2^3))$, что можно сделать непосредственным вычислением, которое мы оставляем читателю.

В случае же $x^4 \neq x^2 + x^{-2}$ надо применять формулы сложения. Сделаем вначале проверку для x-координат. Надо проверить тождество

$$\Lambda^2 + \Lambda + x^4 + x^2 + x^{-2} = x^2,$$

где

$$\Lambda = \frac{y^4 + (\frac{y}{x} + x)(x + x^2 + x^{-2}) + x^2 + x^{-2} + y)}{x^4 + x^2 + x^{-2}}.$$

Заменяя y^4 на $(yx + x^3 + 1)^2 = y^2x^2 + x^6 + 1 = yx^3 + x^5 + x^2 + x^6 + 1$, имеем

$$\Lambda = \frac{yx^3 + x^5 + x^2 + x^6 + 1 + (y + x^2)(1 + x + x^{-3}) + x^2 + x^{-2} + y)}{x^4 + x^2 + x^{-2}}.$$

$$\frac{y(x + x^3 + x^{-3}) + x^5 + x^6 + 1 + x^2(1 + x + x^{-3}) + x^{-2}}{x^4 + x^2 + x^{-2}} =$$

$$= \frac{y}{x} + \frac{x^5 + x^6 + 1 + x^2 + x^3 + x^{-1} + x^{-2}}{x^4 + x^2 + x^{-2}} =$$

$$= \frac{y}{x} + \frac{x^7 + x^8 + x^2 + x^4 + x^5 + x + 1}{x^6 + x^4 + 1} = \frac{y}{x} + x^2 + x + 1.$$

Подставляя $\Lambda = \frac{y}{x} + x^2 + x + 1$, имеем далее

$$\Lambda^2 + \Lambda + x^4 + x^{-2} = \frac{y^2}{x^2} + \frac{y}{x} + x^4 + x^2 + 1 + x^2 + x + 1 + x^4 + x^{-2} =$$

$$\frac{y^2}{x^2} + \frac{y}{x} + x + x^{-2} = \frac{y^2 + yx + x^3 + x}{x^2} = 0,$$

и равенство x-координат доказано.

Для проверки равенства y-координат, используя доказанное тождество, достаточно проверить тождество

$$\Lambda(x^4+x^2)+x^2+y^4=x^2+y^2.$$

Подставляя $\Lambda=\frac{y}{x}+x^2+x+1$ и заменяя y^4 на $yx^3+x^5+x^2+x^6+1$, имеем

$$\Lambda(x^4+x^2)+y^4=(\frac{y}{x}+x^2+x+1)(x^4+x^2)+yx^3+x^5+x^2+x^6+1=$$

$$=(x^2+x+1)(x^4+x^2)+yx+x^5+x^2+x^6+1=yx+x^3+1=y^2,$$

что и требовалось доказать.

Кольцо эндоморфизмов и комплексное умножение. Рассмотрим подкольцо кольца эндоморфизмов кривой $E_a(GF(2^n))$, порожденное эндоморфизмом Фробениуса.

Упражнение 5.5.14 Проверьте, что оно состоит из эндоморфизмов вида

$$a_{m-1}\tau^{m-1}+\ldots+a_0,$$

где a_i — целые числа, $2f=f+f, 3f=f+f+f$, и так далее, $(-f)(P)=-f(P)$.

Далее оно обозначается через $Z[\tau]$. Представление его элементов в виде многочленов от τ с целыми коэффициентами неоднозначно. Например, согласно доказанному тождеству эндоморфизм

$$\tau^2-\mu\tau+2=0.$$

Рассмотрим написанное выше уравнение в поле комплексных чисел и обозначим также через τ его корень

$$\frac{\mu+\sqrt{-7}}{2}.$$

Сопоставим каждому эндоморфизму из кольца $Z[\tau]$ комплексное число

$$a_{m-1}\tau^{m-1}+\ldots+a_0.$$

Множество всех таких чисел также обозначим $Z[\tau]$. Говорят, что кривые E_a обладают *комплексным умножением* на τ.

Упражнение 5.5.15 Проверьте что это множество является подкольцом поля комплексных чисел, а указанное выше отображение — гомоморфизмом колец (т.е. сохраняет кольцевые операции). Пользуясь тождеством

$$\tau^2-\mu\tau+2=0,$$

проверьте, что кольцо $Z[\tau]$ состоит только из элементов вида $a+b\tau$, где a,b — целые числа, и запись в таком виде однозначна.

Назовем нормой $N(\alpha)$ элемента этого кольца число $\alpha\overline{\alpha}$, где $\overline{\alpha}$ — число, сопряженное к α.

Упражнение 5.5.16 Проверьте, что

$$N(a+b\tau) = (a+b\tau)(a+b\overline{\tau}) = a^2 + \mu ab + 2b^2.$$

В частности, $N(\tau) = 2, N(\tau - 1) = 3 - \mu = N(1)$.

Указание: $\tau\overline{\tau} = 2, \tau + \overline{\tau} = \mu$.

Упражнение 5.5.17 Проверьте, что для $\alpha, \beta \in Z[\tau]$

$$N(\alpha\beta) = N(\alpha)N(\beta), N(\alpha/\beta) = N(\alpha)/N(\beta)$$

(мультипликативность нормы).

Указание: $\overline{\alpha}\overline{\beta} = \overline{\alpha\beta}, 1/\overline{\alpha} = \overline{(1/\alpha)}$.

Упражнение 5.5.18 Проверьте, что для $\alpha \in Z[\tau]$

$$\sqrt{N(\alpha)} = \sqrt{\alpha\overline{\alpha}}$$

совпадает с модулем комплексного числа α. Поэтому справедливо неравенство треугольника

$$\sqrt{N(\alpha+\beta)} \leq \sqrt{N(\alpha)} + \sqrt{N(\beta)}.$$

Упражнение 5.5.19 Проверьте, что $N(\tau^n - 1) = |E_a(GF(2^n))| = N(n)$.

Указание:

$$N(\tau^n - 1) = (\tau^n - 1)(\overline{\tau}^n - 1) = (\tau\overline{\tau})^n + 1 - tau^n - \overline{\tau}^n =$$

$$= 2^n + 1 - V_n = |E_a(GF(2^n))|.$$

Упражнение 5.5.20 Проверьте, что $N((\tau^n - 1)/(\tau - 1)) = N(n)/N(1)$.

Евклидовость и факториальность кольца $Z[\tau]$ В кольце $Z[\tau]$ можно выполнять деление с остатком в том смысле, что для любых его элементов $\alpha, \beta \neq 0$, найдутся элементы $\delta, \rho \in Z[\tau]$, такие, что $\alpha = \beta\delta + \rho, N(\rho) < N(\beta)$. Далее это будет доказано. Кольца с таким свойством называют *евклидовыми.* Элемент ρ называется *остатком,* а элемент δ *частным.* Вообще говоря, они могут быть определены неоднозначно. Если остаток равен нулю, то частное называется *делителем. Наибольшим общим делителем* двух элементов кольца называется такой их общий делитель, который делится без остатка на любой другой общий делитель. Наибольший общий делитель определен неоднозначно, а с точностью умножения на обратимый элемент кольца.

В любом евклидовом кольце с помощью алгоритма Евклида можно найти наибольший обший делитель любых его элементов α, β и представить его в виде $x\alpha + y\beta$, где x, y тоже элементы этого кольца. Доказывается это точно также, как для кольца целых чисел.

Упражнение 5.5.21 Докажите это для кольца $Z[\tau]$ без использования алгоритма Евклида.

Указание. Пусть $\delta = x\alpha+y\beta \neq 0$ —элемент с минимальной нормой среди всех элементов такого вида. Тогда $alpha$ и β делятся на δ без остатка, в противном случае остаток $\rho = \alpha - \delta\gamma = \alpha(1 - x\gamma) - \beta y\gamma$ имел бы меньшую норму. Если δ' другой общий делитель α, β, то он делит $x\alpha + y\beta = \delta$.

Упражнение 5.5.22 Докажите это в кольце $Z[\tau]$ наибольший общий делитель является наибольшим по норме среди всех общих делителей и определен с точностью до смены знака.

Указание: примените свойство мультипликативности нормы и предыдущие упражнения.

Элемент кольца (коммутативного, с единицей, без делителей нуля) называется *простым*, если его нельзя представить в виде произведения необратимых элементов кольца. Элемент называется *необратимым*, если он не имеет обратного элемента в этом кольце. Очевидно, что кольцо $Z[\tau]$ коммутативно, имеет единицу и не имеет делителей нуля, так как лежит в поле.

Упражнение 5.5.23 Проверьте, что в кольце $Z[\tau]$ обратимы только ± 1, и других элементов с единичной нормой в нем нет.

Указание. Из мультипликативности нормы следует, что если элемент обратим, то его норма равна 1. Очевидно $N(a + b\tau) = a^2 + \mu ab + 2b^2 = (a + \mu b/2)^2 + 7b^2/4 \geq 7b^2/4$. Поэтому $N(a + b\tau) = 1$ если и только если $b = 0, a = \pm 1$.

Упражнение 5.5.24 Проверьте, что если у элемента норма — простое число, то элемент — простой. В частности, τ — простой элемент и простые целые числа являются простыми элементами кольца $Z[\tau]$.

Указание: примените мультипликативность нормы и предыдущее упражнение.

Упражнение 5.5.25 Докажите, что любой необратимый ненулевой элемент можно представить в виде произведения простых элементов.

Единственность такого представления не для всех таких колец имеет место, но ее можно доказать для евклидовых колец.

Упражнение 5.5.26 Докажите для евклидовых колец, что если ab делится на простой элемент p, то a или b делится на p.

Указание. Пусть a не делится на p. Тогда наибольший общий делитель a и p равен ± 1, так как p простой элемент. Поэтому для некоторых x, y имеем $ax + py = 1$. Отсюда $b = abx + pyb$ кратно p, так как ab и pby делятся на p.

Упражнение 5.5.27 Докажите для евклидовых колец, что если $a_1 \dots a_n$ делится на простой элемент p, то хотя бы один из a_i делится на p.

Указание: примените индукцию и предыдущее упражнение.

Упражнение 5.5.28 Докажите для евклидовых колец, что разложение на простые множители определено однозначно (с точностью до перестановки сомножителей и домножения на обратимые элементы).

Указание. Также, как и в случае кольца целых чисел, примените индукцию по числу простых сомножителей и предыдущее упражнение.

Кольца с единственностью разложения на простые элементы называются *факториальными*.

Минимальные уравновешенные τ-адические коды. Представление произвольного элемента кольца $Z[\tau]$ в виде

$$a_m\tau^m + \ldots + a_0,$$

где $a_i = 0, \pm 1$, назовем *уравновешенным τ-адическим кодом*. Уравновешенный τ-адический код назовем *минимальным*, если $a_i a_{i+1} = 0$ при любом i.

Пример 5.5.1 Число 9 имеет минимальный τ-адический код $(1, 0, -1, 0, 0, 1)$, так как

$$9 = \tau^5 - \tau^3 + 1.$$

Упражнение 5.5.29 Проверьте это прямым вычислением.

τ-адические коды для натуральных чисел позволяют умножать скалярно на эти числа точки кривой E_a, не используя операцию удвоения точек, а используя только сложение.

Пример 5.5.2 Если $P = (x, y)$, то согласно формулам Фробениуса

$$9P = (x^{32}, y^{32}) - (x^8, y^8) + (x, y).$$

Пренебрегая сложностью квадрирования, получаем оценку сложности

$$L(9) \leq 2(L(I) + 2L(M)),$$

где $L(I), L(M)$ сложность инвертирования и умножения в данном поле. Применяя формулы

$$9P = \tau^3(\tau^2 P - P) + P,$$

можно воспользоваться проективными координатами и получить оценку

$$L(9) \leq= 16L(M) + L(I) + 2L(M).$$

Если же воспользоваться двоичным кодом $9 = 2^3 + 1$, то

$$9P = 2(2(2P)) + P,$$

откуда имеем

$$L(9) \leq= 4(L(I) + 2L(M)).$$

Используя проективные координаты, получаем, с учетом возвращения к аффинным координатам

$$L(9) \leq= 3(4L(M)) + 8L(M) + L(I) + 2L(M).$$

Упражнение 5.5.30 Попробуйте улучшить эту оценку методом Монтгомери и методом использования ополовинивания вместо удвоения.

Справедлива

Теорема 5.5.1 *Любой элемент кольца $Z[\tau]$ имеет единственный минимальный код.*

Доказательство основано на следующей лемме.

Лемма 5.5.1 *Элемент $c_0 + c_1\tau \in Z[\tau]$ делится на τ если и только если c_0 четно и делится на τ^2 если и только если $c_0 \equiv 2c_1 (\mathrm{mod} 4)$. Всегда справедливо только одно из утверждений: α делится на τ, $\alpha \equiv \pm 1 (\mathrm{mod}\tau^2)$.*

Упражнение 5.5.31 Докажите лемму.

Указание. Если $c_0 + c_1\tau = \tau(d_0 + d_1\tau)$, то согласно тождеству Фробениуса $c_0 + c_1\tau = -2d_1 + (d_0 + \mu d_1)\tau$, откуда $c_0 = -2d_1$. Обратно, если c_0 четно, то

$$\frac{c_0 + c_1\tau}{\tau} = \frac{\mu c_0 + 2c_1}{2} - \frac{c_0}{2}\tau.$$

Вместо деления на τ^2 делим на $\mu\tau - 2$. Тогда

$$c_0 + c_1\tau = (d_0 + d_1\tau)(\mu\tau - 2) = -2(d_0 + \mu d_1) + (\mu d_0 - d_1)\tau.$$

Учитывая $\mu = \pm 1$, имеем $c_0 = 2c_1 \bmod 4$. Обратно,

$$\frac{c_0 + c_1\tau}{\tau^2} = -\frac{(1 + 2\mu)c_0 + 2\mu c_1}{4} + \mu\frac{c_0 - 2c_1}{4}\tau.$$

Если c_0 четно, то $\alpha = c_0 + c_1\tau$ делится на τ, если c_0 нечетно, то $c_0 \pm 1 = 2c_1 \bmod 4$, значит $\alpha \pm 1$ делится на τ^2.

Из последнего утверждения леммы следует, что два минимальных кода для одного элемента из $Z[\tau]$ имеют одинаковые самые правые цифры.

Упражнение 5.5.32 Проверьте это.

Отсюда, применяя индукцию, выводим единственность минимальных кода.

Упражнение 5.5.33 Проведите это рассуждение более подробно.

Существование минимального кода для $c_0 + c_1\tau$ вытекает из следующего алгоритма его построения. Положим $r_0 = c_0, r_1 = c_1, i = 0$. Пока $r_0 \neq 0$ или $r_1 \neq 0$ в цикле делаем следующие операции: если r_0 нечетно, то $u_i = 2 - (r_0 - 2r_1 \bmod 4), r_0 = r_0 - u_i$ иначе $u_i = 0$ (вычисление очередной цифры $u_i = 0, \pm 1$); потом $i = i + 1, r_0 = r_1 + \mu r_0/2, r_1 = -r_0/2$ (деление $r_0 + r_1\tau$ на τ).

После окончания работы цикла в массиве $(u_{i-1}, \ldots, u_0)$ будет записан минимальный код.

Упражнение 5.5.34 Проверьте это.

Определение минимального τ-адического кода совершенно аналогично определению минимального 2-адического кода и между ними можно установить взаимно-однозначное соответствие. Известно, что плотность $pm1$ в минимальном 2-адическом коде в среднем асимптотически равна 1/3. Поэтому то же верно и для τ-адического кода. Однако длина τ-адического кода для числа n асимптотически вдвое больше, чем у 2-адического кода, так как эта длина l для произвольного $\alpha \in Z[\tau]$ оценивается как

$$\log_2 N(\alpha) - 0.55 < l < \log_2 N(\alpha) + 3.52,$$

а $\log_2 N(n) = 2\log_2 n$.

Доказательство этих неравенств основано на следующих утверждениях, которые мы формулируем в виде упражнений.

Обозначим $N_{max}(l)$ наибольшую, а через $N_{min}(l)$ наименьшую норму элементов $Z[\tau]$ с минимальным кодом длины l.

Упражнение 5.5.35 Проверьте, что для любого l $2N_{max}(l) \le N_{max}m(l+1)$.

Указание: если α элемент с минимальным кодом длины l и нормой $N_{max}(l)$, то $N(\alpha\tau) = 2N(\alpha)$ и его минимальный код имеет длину $l+1$.

Упражнение 5.5.36 Проверьте, что наибольшая норма элементов $Z[\tau]$ с минимальным кодом длины *не более* l равна $N_{max}(l)$.

Указание: примените предыдущее упражнение.

Упражнение 5.5.37 Докажите, что при $c > k$

$$\sqrt{N_{max}(c)} \le 2^{k/2}\sqrt{N_{max}(c-k)} + \sqrt{N_{max}(k)}.$$
$$\sqrt{N_{min}(c)} \ge 2^{k/2}\sqrt{N_{min}(c-k)} - \sqrt{N_{max}(k)}.$$

Указание. Пусть γ имеет код $(u_{c-1}, \ldots, u_0)$, $N(\gamma) = N_m(c)$, где $N_m = N_{max}$ или N_{min}, и ρ имеет код $(u_{k-1}, \ldots, u_0)$, тогда $N(\rho) \le N_{max}(k)$, $\gamma = \tau^k\delta + \rho$, $N_{min}(c-k) \le N(\delta) \le N_{max}(c-k)$, и из неравенства треугольника имеем

$$\sqrt{N(\gamma)} \le 2^{k/2}\sqrt{N(\delta)} + \sqrt{N(\rho)},$$

$$\sqrt{N(\gamma)} \ge 2^{k/2}\sqrt{N(\delta)} - \sqrt{N(\rho)}.$$

Положим для краткости $M_l = \sqrt{2^{-l}N_{max}(l)}$.

Упражнение 5.5.38 Выведите из предыдущего упражнения, что при $c > k$

$$M_c \le M_{c-k} + 2^{(c-k)/2}M_k.$$

Упражнение 5.5.39 Докажите для положительных d, q

$$\frac{M_{dq}}{1-2^{-dq/2}} \le \frac{M_d}{1-2^{-d/2}}.$$

Указание: из предыдущего упражнения следует, что

$$M_{(k+1)d} - M_{kd} \le 2^{-kd/2} M_d;$$

просуммируйте эти неравенства и примените формулу суммирования геометрической прогрессии.

Упражнение 5.5.40 Докажите, что для $l > d$

$$M_l < \frac{M_d}{1-2^{-d/2}},$$

$$N_{max}(l) < \frac{N_{max}(d)}{(2^{d/2}-1)^2} \cdot 2^l.$$

Указание: пусть $(k-1)d < l \le kd$; примените предыдущее упражнение.

Упражнение 5.5.41 Докажите, что для $l > 2d$

$$N_{min}(l) > \left(\sqrt{N_{min}(d)} - \frac{N_{max}(d)}{2^{d/2}-1}\right)^2 \cdot 2^{l-d}.$$

Указание: примените предыдущие упражнения.

Из предыдущих упражнений следует, что если α имеет минимальный уравновешенный τ-адический код длины $l > 2d$, то

$$\left(\sqrt{N_{min}(d)} - \frac{N_{max}(d)}{2^{d/2}-1}\right)^2 \cdot 2^{l-d} < N(\alpha) < \frac{N_{max}(d)}{(2^{d/2}-1)^2} \cdot 2^l.$$

Применяя это неравенство при $d = 15$ и вычисляя, что $N_{max}(15) = 47324$, $N_{min}(15) = 2996$, имеем при $l > 30$

$$1.399 \cdot 2 \cdot 2^{l-4} < N(\alpha) < 0.731 \cdot 2^{l+1},$$

откуда

$$\log_2 N(\alpha) - 0.55 < l < \log_2 N(\alpha) + 3.52.$$

В [157] показано, что эти оценки близки к точным.

Используя их, получаем, что kP можно вычислить, сделав в среднем $2/3 \log_2 k$ сложений со сложностью $(2/3 \log_2 k)(L(I)+2L(M))$ без использования проективных координат и со сложностью $(2/3 \log_2 k)(8L(M)) + L(I) + 2L(M)$ с использованием проективных координат. В стандартном алгоритме с применением двоичного кода среднее число используемых сложений точек кривой вдвое меньше, но зато используется $\log_2 k$ удвоений, и общая сложность без использования проективных координат равна $(4/3 \log_2 k)(L(I)+2L(M))$, а с использованием проективных координат она равна $((4+8/3)\log_2 k)L(M) + L(I) + 2L(M)$.

Преимущество указанного метода еще более возрастает, если перед построением минимального уравновешенного кода выполнить модулярную редукцию, которая уменьшает в два раза длину кода.

5.5.2 Использование модулярной редукции

Эквивалентность τ-адических разложений. Будем говорить, что $\gamma, \rho \in Z[\tau]$ эквивалентны, если эндоморфизмы $\gamma(P), \rho(P)$ совпадают на данной кривой $E_a(GF(2^n))$..

Лемма 5.5.2 *Если $\gamma = \rho \bmod \tau^n - 1$, то γ и ρ эквивалентны*

Доказательство. Так как согласно одному из упражнений τ^n — тождественный эндоморфизм, то $(\tau^n - 1)P = O$ для любой точки P данной кривой. Если в кольце $Z[\tau]$ $\gamma = \rho \bmod \tau^n - 1$, то $\gamma = \rho + \delta(\tau^n - 1)$, значит

$$\gamma(P) = \rho(P) + \delta(\tau^n - 1)(P) = \rho(P) + \delta(O) = \rho(P) + O = \rho(P).$$

Положим $\delta = (\tau^n - 1)/(\tau - 1)$. Было ранее проверено, что $N(\delta) = r = |E_a(GF(2^n))|/h$, $h = 3 - \mu$, и r — порядок главной подгруппы.

Лемма 5.5.3 *Если P точка главной подгруппы, то $\delta(P) = O$.*

Доказательство. Из упражнений следует, что $P = hQ$, $(\tau^n - 1)Q = O$. Поэтому $\delta((\tau - 1)Q) = (\tau^n - 1)Q = O$. Положим эндоморфизм $\overline{\tau - 1} = -1 + \mu - \tau$, тогда $\overline{\tau - 1}(\tau - 1) = 1 - \mu + \mu\tau - \tau^2 = 3 - \mu = h$,

$$O = (\overline{\tau - 1})(\delta((\tau - 1)Q)) = \delta(\overline{\tau - 1}(\tau - 1)Q) = \delta(hQ) = \delta(P).$$

Из леммы следует

Теорема 5.5.2 *Если P принадлежит главной подгруппе группы $E_a(GF(2^n))$, и $\rho = \gamma \bmod \delta$, то эндоморфизмы ρ и γ совпадают на главной подгруппе.*

Доказательство. Если в кольце $Z[\tau]$ $\gamma = \rho \bmod \delta$, то $\gamma = \rho + \delta\beta$, значит

$$\gamma(P) = \rho(P) + \beta(\delta(P)) = \rho(P) + \beta(O) = \rho(P) + O = \rho(P).$$

Пользуясь предыдущей теоремой, для вычисления kP в случае P принадлежащего главной подгруппе сначала можно найти $\rho \in Z[\tau]$, такой, что $\rho = k \bmod q\delta$, а потом для него вычислить минимальный τ-адический код и вычислить $\rho(P)$, пользуясь этим кодом. Для нахождения ρ можно применить алгоритм деления, так как в качестве ρ можно взять остаток от деления k на δ. Тогда $N(\rho) < N(\delta) = r$. Далее мы покажем, что на самом деле даже $N(\rho) \leq 4N(\delta)/7 = 4r/7$. Так как $r = |E_a(GF(2^n))|/h$, $h = 3 - \mu = 3 - (-1)^{1-a} = 2^{2-a}$, $E_a(GF(2^n)) = 2^n + 1 - V_n = 2^n + O(2^{n/2}$, то

$$N(\rho) \leq \frac{4N(r)}{7} \leq \frac{2^{n+a}}{7} + O(2^{n/2}.$$

Поэтому для длины l минимального τ-кода для ρ имеем неравенство

$$l < \log_2 N(\rho) + 3.52 < n + a + 0.8,$$

а так как l — целое, то $l \le n + a, a = 0, 1$.

Можно доказать, что среднее число ненулевых цифр в минимальном τ-коде для $\rho = k \bmod \delta$ асимптотически равно $l/3$. Поэтому число сложений точек при вычислении kP при $k = O(r)$ будет вдвое меньше, чем было раньше.

Далее будет доказана евклидовость этого кольца и предложен алгоритм деления, для которого оценка нормы остатка через норму делителя имеет вид $N(\rho) \le 4N(\delta)/7$.

Пчелиные соты и округление до целых точек. Определим на плоскости (λ_0, λ_1) выпуклый центрально-симметричный шестиугольник U системой неравенств

$$-1 \le 2\lambda_0 + \mu\lambda_1 < 1$$
$$-2 \le \lambda_0 + 4\mu\lambda_1 < 2$$
$$-2 \le \lambda_0 - 3\mu\lambda_1 < 2.$$

Очевидно U является пересечением трех полос, каждая из которых ограничена параллельными прямыми. Для каждой целой точки ξ этой плоскости обозначим $U(\xi)$ шестиугольник, полученный из U параллельным переносом на вектор ξ. Центром $U(\xi)$ будет, естественно, точка ξ.

Упражнение 5.5.42 Проверьте, что система $\{U(\xi), \xi \in Z^2\}$ состоит из попарно непересекающихся шестиугольников, объединение которых покрывает всю плоскость.

Указание. Каждый из шестиугольников решетки имеет 6 соседей, каждые два соседа имеют одну общую сторону.

Упражнение 5.5.43 Докажите, что для любой точки $\lambda \in U$ $N(\lambda) \le 4/7$.

Указание. Рассмотрите эллипс $\lambda_0^2 + \mu\lambda_0\lambda_1 + 2\lambda_1^2 = 4/7$, состоящий из точек с нормой не большей $4/7$ и проверьте, что он проходит через вершины U, и поэтому его содержит в силу выпуклости. Например, одна из вершин является точкой пересечения прямых $2\lambda_0 + \mu\lambda_1 = 1, \lambda_0 + 4\mu\lambda_1 = 2$, т.е. точкой $(2/7, 3/7mu) = (2/7, \pm 3/7)$, лежащей на эллипсе, так как $4/49 + 6/49 + 18/49 = 4/7$. В силу симметрии остается проверить еще две вершины.

Упражнение 5.5.44 Докажите, что для любой точки $\lambda \in U$ и любого $\xi \ne 0, \xi \in Z[\tau]$ $N(\lambda) < N(\lambda + \xi)$.

Указание. Проверьте, что $N(\lambda) < N(\lambda \pm 1)$ если и только если $|2\lambda_0 + \mu\lambda_1| < 1$, $N(\lambda) < N(\lambda \pm \tau)$ если и только если $|4\lambda_1 + \mu\lambda_0| < 2$, $N(\lambda) < N(\lambda \pm \overline{\tau})$ если и только если $|mu\lambda_0 - 3\lambda_1| < 2$, где $\overline{\tau} = \frac{\mu - i\sqrt{7}}{2} = -\tau + \mu$. Так как $\lambda \in U$, то выполнены все три условия, поэтому неравенство доказано для $\xi = \pm 1, \pm\tau, \pm\overline{\tau}$. Для остальных ненулевых $\xi \in Z[\tau]$ $N(\xi) \ge 4$, , а так как $N(\lambda) \le 4/7$, то согласно неравенству треугольника

$$\sqrt{N(\lambda)} \le 2/\sqrt{7} < 2 - 2/\sqrt{7} \le \sqrt{N(\xi)} - \sqrt{N(\lambda)} < \sqrt{N(\lambda + \xi)}.$$

Для произвольной точки λ обозначим $\kappa = Round(\lambda)$ такую целую точку, что $\lambda \in U(\kappa)$ и определим $\zeta := ((\lambda)) = \lambda - \kappa$. Очевидно $\zeta \in U$, поэтому $N(\zeta) \leq 4/7$, и для любого $\alpha \in Z[\tau]$ $N(\zeta) < N(\zeta + \alpha)$.

Определим $Round(\lambda)$ для действительных чисел равенством $Round(\lambda) = \lfloor \lambda + 1/2 \rfloor$, тогда очевидно $((\lambda)) \leq 1/2$.

Для комплексных чисел $\lambda = (\lambda_1, \lambda_2)$ вычислить $Round(\lambda)$ можно следующим алгоритмом.

Положим $f_i = Round(\lambda_i), \eta_i = \lambda_i - f_i, h_i = 0, i = 1, 2$.

Положим $\eta = 2\eta_0 + \mu\eta_1$.

Если $\eta \geq 1$,

то

если $\eta_0 - 3\mu\eta_1 < -1$, то $h_1 = \mu$, иначе $h_0 = 1$

иначе

если $\eta_0 + 4\mu\eta_1 \geq 2$, то $h_1 = \mu$

Если $\eta < -1$,

то

если $\eta_0 - 3\mu\eta_1 \geq -1$, то $h_1 == \mu$, иначе $h_0 = -1$

иначе

если $\eta_0 + 4\mu\eta_1 < -2$, то $h_1 = -\mu$

На выходе алгоритм выдает $q_i = f_i + h_i, i = 1, 2$.

Упражнение 5.5.45 Убедитесь в правильности алгоритма.

Деление с остатком в кольце $Z[\tau]$. Пусть делимое $\gamma = c_0 + c_1\tau \in Z[\tau]$, делитель $\delta = d_0 + d_1\tau \in Z[\tau]$, частное $\kappa = q_0 + q_1\tau \in Z[\tau]$, остаток $\rho = r_0 + r_1\tau \in Z[\tau]$ таковы, что $\gamma = \kappa\delta + \rho$, и мы хотим минимизировать $N(\rho)$. Положим $\lambda = \gamma/\delta$, тогда

$$\lambda = \frac{\gamma\overline{\delta}}{\delta\overline{\delta}} = \frac{\gamma\overline{\delta}}{N(\delta)} = \frac{g_0 + g_1\tau}{N}.$$

Положим $\kappa = Round(\frac{g_0}{N} + \frac{g_1}{N}\tau)$. Тогда $\rho = \gamma - \kappa\delta$ и алгоритм деления выглядит следующим образом.

Положим

$$g_0 = c_0d_0 + \mu c_0d_1 + 2c_1d_1,$$

$$g_1 = c_1d_0 - c_0d_1,$$

$$N = d_0^2 + \mu d_0d_1 + 2d_1^2,$$

$$\lambda_i = g_i/N, i = 1, 2,$$

$$(q_0, q_1) = Round(\lambda_0, \lambda_1),$$

$$r_0 = c_0 - d_0q_0 + 2d_1q_1,$$

$$r_1 = c_1 - d_1q_0 - d_0q_1 - mud_1q_1.$$

Упражнение 5.5.46 Проверьте правильность алгоритма.

Так как

$$\rho = \gamma - \kappa\delta = (\lambda - \kappa)\delta = (\lambda - Round(\lambda))\delta = ((\lambda))\delta = ((\gamma/\delta))\delta,$$

то $N(\rho) = N(((\lambda))\delta) = N(((\lambda)))N(\delta) \leq \frac{4}{7}N(\delta)$.

Далее этот остаток ρ обозначаем $\gamma \bmod \delta$.

Вычисление редуцированного τ-адического кода. Для этого надо вычислить $k \bmod \delta$, где $\delta = d_0 + d_1\tau = (\tau^n - 1)/(\tau - 1)$, $N(\delta) = r$. а потом вычислить минимальный τ-адический код. Для выполнения деления с остатком применяем предыдущий алгоритм к $\gamma = k$. Тогда $g_i = s_i k, s_0 = d_0 + \mu d_1, s_1 = -d_1$.

Для вычисления целых чисел s_i воспользуемся второй последовательностью Люка, определяемой рекуррентными соотношениями

$$U_0 = 0, U_1 = 1, U_{i+1} = \mu U_i - 2U_{i-1}, i > 0.$$

Упражнение 5.5.47 Докажите по индукции, что U_i целые нечетные числа.

Упражнение 5.5.48 Докажите по индукции, что $U_i = (\tau^i - \overline{\tau})/\sqrt{-7}$.

Упражнение 5.5.49 Докажите по индукции, что $\tau^i = U_i\tau - 2U_{i-1}, i > 0$.

Упражнение 5.5.50 Докажите, что $U_i^2 - \mu U_i U_{i-1} + 2U_{i-1}^2 = 2^{i-1}, i > 0$.

Указание: предыдущее тождество умножьте на сопряженное.

Так как при переходе к сопряженным числам имеем

$$d_0 + d_1\overline{\tau} = (\overline{\tau}^n - 1)/(\overline{\tau} - 1),$$

то, складывая с равенством

$$d_0 + d_1\tau = (\tau^n - 1)/(\tau - 1),$$

находим, складывая дроби, что

$$2d_0 + \mu d_1 = (U_n\tau - 2U_{n-1} - 1)/(\tau - 1) + (U_n\overline{\tau} - 2U_{n-1} - 1)/(\overline{\tau} - 1) =$$
$$= \frac{(4-\mu)U_n + (4-2\mu)U_{n-1} + 2 - \mu}{3 - \mu}.$$

Вычитая равенства, имеем

$$d_1\sqrt{-7} = (U_n\tau - 2U_{n-1} - 1)/(\tau - 1) - (U_n\overline{\tau} - 2U_{n-1} - 1)/(\overline{\tau} - 1) =$$
$$= \frac{\sqrt{-7}(-U_n + 2U_{n-1} + 1)}{3 - \mu},$$

откуда

$$s_1 = -d_1 = \frac{U_n - 2U_{n-1} - 1}{3 - \mu},$$

значит

$$2s_0 = 2d_0 + 2\mu d_1 == \frac{(4-\mu)U_n + (4-2\mu)U_{n-1} + 2 - \mu}{3 - \mu} + \frac{\mu(-U_n + 2U_{n-1} + 1)}{3 - \mu} =$$
$$= \frac{(4-2\mu)U_n + 4U_{n-1} + 2}{3 - \mu} = 2\frac{(2-\mu)U_n + 2U_{n-1} + 1}{3 - \mu}.$$

Упражнение 5.5.51 Проверьте, что

$$s_i = (-1)^i(1 - \mu U_{n+3-a-i})/(3-\mu), i = 0, 1.$$

Указание: при $a = 0$

$$U_{i+1} = \mu U_i - 2U_{i-1} = -U_i - 2U_{i-1} = (U_{i-1} + 2U_{i-2}) - 2U_{i-1} =$$

$$= -U_{i-1} + 2U_{i-2}.$$

Вычисление $r_0 + r_1\tau = k \bmod (\tau^n - 1)/(\tau - 1)$ выполняется следующим алгоритмом.

Положим $d_0 = s_0 + \mu s_1, \lambda_i = s_i k/r$, $(q_0, q_1) = Round(\lambda_0, \lambda_1)$, тогда

$$r_0 = k - d_0 q_0 - 2s_1 q_1, r_1 = s_1 q_0 - s_0 q_1.$$

Оконный τ-адический метод. Этот метод в определенном смысле аналогичен оконному 2-адическому методу, и также является обобщением обычного τ-адического метода на случай окна произвольной длины k. В случае отсутствия ограничений на память, этот метод ускоряет скалярное умножение на несуперсингулярных кривых при подходящем выборе ширины окна. Основан он на следующей теореме.

Теорема 5.5.3 *Для любого натурального w и любого элемента кольца $Z[\tau]$ существует единственное его представление в виде*

$$a_m \tau^{@} + \ldots + a_0,$$

где $a_i = 0, \pm 1, \pm 3, \ldots, \pm 2^{w-1} - 1$, и среди любых w подряд идущих коэффициентов максимум один ненулевой.

Для доказательства понадобятся следующие определение и лемма.

Определим число t_m так, что $t_m = 2U_{m-1}U_m^{-1} \bmod 2^m$. Так как U_m нечетно, то $U_m^{-1} \bmod 2^m$ существует и однозначно определено, значит t_m тоже определено однозначно, четно, но не кратно 4. Согласно одному из предыдущих упражнений

$$t_m^2 - \mu t_m + 2 = 0 \bmod 2^m.$$

Упражнение 5.5.52 Проверьте это.

Таким образом t_m удовлетворяет по модулю 2^i тому же уравнению, что и число τ в поле комплексных чисел. Поэтому соответствие $\tau \to t_m$ можно естественным образом продолжить до отображения

$$\phi_m : u_0 + u_1\tau \to u_0 + u_1 t_m$$

кольца $Z[\tau]$ в кольцо Z_{2^m} вычетов по модулю 2^m.

Будем говорить, что элемент $u_0 + u_1\tau$ нечетен, если u_0 нечетно; элемент $u_0 + u_1 t_m \bmod 2^m$ нечетен, если u_0 нечетно. Аналогично определяем понятие четности.

Упражнение 5.5.53 Проверьте, что это отображение является гомоморфизмом колец, т.е. сумму переводит в сумму и произведение — в произведение (а также нуль — в нуль и единицу — в единицу). Проверьте, что нечетные элементы оно переводит в нечетные, а четные — в четные

Следующая лемма обобщает одну из предыдущих.

Лемма 5.5.4 *$\phi_m(\alpha) = 0$ если и только если $\alpha \in Z[\tau]$ делится нацело в этом кольце на τ^m.*

Доказательство. Так как $\phi_m(\tau) = t_m = 2 \bmod 4$, то по индукции можно проверить, что $\phi_m(\tau^j) = t_m^j = 2^j \bmod 2^{j+1}$. Значит $\phi_m(\tau^m) = 0 \bmod 2^m$, поэтому

$$\phi_m(\alpha) = \phi_m(\beta\tau^m) = \phi_m(\beta)\phi_m(\tau^m) = 0 \bmod 2^m.$$

Для доказательства в другую сторону предположим, что $\alpha = \beta\tau^j, j < m, \beta = u_0 + u_1\tau, u_0 \neq 0$, тогда можно считать, что $u_0 \in Z[\tau]$ есть остаток от деления β на τ, поэтому $0 < N(u_0) < N(\tau) = 2$, значит $N(u_0) = 1$, поэтому $u_0 = \pm 1$, откуда следует нечетность $\phi_m(\beta) = u_0 + \phi_m(u_1)\phi_m(\tau)$, и

$$\phi_m(\alpha) = \phi_m(\beta\tau^j) = \phi_m(\beta)\phi_m(\tau^j) = 2^j \bmod 2^{j+1} \neq 0 \bmod 2^m.$$

Из леммы следует, что при отображении ϕ_m нечетные классы вычетов в кольце $Z[\tau]$ по модулю τ^m переходят в нечетные классы вычетов в кольце Z_{2^m}, а четные — в четные, причем равные по модулю τ^m элементы имеют одинаковые образы при этом отображении.

Упражнение 5.5.54 Докажите это.

Указание. Если два разных по модулю τ^m элемента из $Z[\tau]$ переходят в один элемент Z_{2^m}, то их разность согласно лемме должна делиться на τ^m, что невозможно.

Доказательство единственности разложения в теореме можно провести индукцией по его длине. База индукции очевидна. Для обоснования шага индукции заметим, что любой нечетный элемент кольца $Z[\tau]$ по модулю τ^w равен только одному из чисел $\pm 1, \pm 3, \ldots, \pm 2^{w-1} - 1$. Поэтому в разложении нечетного элемента

$$\alpha = a_m\tau^m + \ldots + a_0$$

$a_0 \neq 0$, поэтому $a_i = 0, i = 1, \ldots, w-1$, значит $\alpha = a_0 \bmod \tau^w$, равный одному из чисел $\pm 1, \pm 3, \ldots, \pm 2^{w-1} - 1$, определен однозначно. По предположению индукции $(\alpha - a_0)/\tau^w$ имеет единственное разложение

$$a_m\tau^{m-w} + \ldots + a_w.$$

Значит разложение нечетного элемента α определено однозначно. В разложении четного элемента

$$\alpha = a_m\tau^m + \ldots + a_0$$

a_0 всегда должно быть четным, а значит по условию теоремы нулем, и определяется также однозначно. Остальные коэффициенты также однозначно определяются из разложения

$$\alpha/\tau = a_m\tau^{m-1} + \ldots + a_1$$

Доказательство существования разложения проводится индукцией, а само разложение строится следующими алгоритмом. Пусть $\alpha = u_0 + u_1\tau, u_i \in Z$. Вначале список S коэффициентов разложения, в котором в конце работы алгоритма появятся коэффициенты $a_m, \ldots, a_0$, пуст. Если u_0 четно, то, то записываем $a_0 = 0$ в этот список и заменяем элемент α на элемент $\alpha/\tau = \mu u_0/2 + u_1 - \tau u_0/2$, т.е. делаем присваивания $u_0 = u_1 + \mu u_0/2, u_1 = -u_0/2$. Если же u_0 нечетно, то вычисляем $\phi_w(\alpha) = u_0 + u_1 t_w \in Z_{2^w}$, находим равный ему по модулю 2^w элемент u в списке $\pm 1, \pm 3, \ldots, \pm 2^{w-1} - 1$, вычисляем $\alpha = \alpha - u$, делая присваивание $u_0 = u_0 - u$, заносим u в список коэффициентов на очередное место, замечаем, что теперь α согласно лемме будет кратно τ^w, и повторяем (не менее чем w раз подряд) указанную выше процедуру деления четного элемента на τ, пока не получится нечетный элемент или u_1 не окажется равным нулю, на чем работа алгоритма заканчивается.

Известно [157], что среднее доля ненулевых коэффициентов в разложении с шириной окна w асимптотически равна $1/(w+1)$. Поэтому общая оценка числа сложений точек кривой $E_a(GF(2^m))$ для выполнения скалярного умножения равна $2^{w-2} - 1 + m/(w+1)$ и требуется запоминание $2^{w-2} + 2$ точек. Алгоритм заключается в следующем. Берется точка P из главной подгруппы. При вычислении точки kP можно предполагать, что $k \le r/2$, где r— порядок главной подгруппы, иначе вместо P можно взять $-P$, тогда $kP = (r-k)(-P)$. Далее вычисляем $k_0 + k_1\tau = k \bmod \delta$ где $\delta = (\tau^m - 1)/(\tau - 1)$. Потом строим для $k_0 + k_1\tau$ разложение

$$a_n\tau^n + \ldots + a_0, n \le m + a + 2, a_i = 0, \pm 1, \pm 3, \ldots \pm 2^{w-1} - 1.$$

Потом с помощью этого разложения вычисляем

$$kP = (k_0 + k_1\tau)P = (a_n\tau^n + \ldots + a_0)P$$

аналогично тому как это делалось в случае использования 2-адического разложения, но при этом удвоение (или ополовинивание) точек заменяется на гораздо более быструю операцию возведения в квадрат обоих координат точки. При этом точки $P, 3P, \ldots 2^{w-1} - 1$ вычисляются, как обычно, заранее и запоминаются. Если точка P заранее известна, то $2^{w-2} - 2$ предварительных сложений можно не учитывать в окончательной оценке сложности.

Для ускорения вычисления $k \bmod \delta$ в описанном выше алгоритме деления с остатком можно заменить целочисленное деление на норму $N(\delta) = r = (2^m + 1 - V_m)/(3 + \mu)$, асимптотически близкую к степени двойки, приближенным выполнением этой операции, использующим только целочисленные умножения.

А именно, вместо точного вычисления $\lambda_i = s_i n/r$ оно находится с точностью C бит с помощью следующей процедуры:

$$K := C + \frac{m+5}{2}; n' = \left\lfloor \frac{n}{2^{m-K-2+a}} \right\rfloor;$$

$$g'_i = s_i n'; h'_i = \left\lfloor \frac{g'}{2^m} \right\rfloor; j'_i = V_m h'_i,$$

$$l'_i = Round\left(\frac{g'_i + j'_i}{2^{K-C}}\right); l_i := \frac{l'_i}{2^C}.$$

Вероятность того, что полученный с использованием этой процедуры деления остаток ρ' не совпадает с остатком ρ стандартной процедуры деления не превосходит 2^{-C+5}. Но при это длина τ-адического разложения, построенного с помощью этой приближенной процедуры, не превосходит $m+a+3$, где $a = 0, 1$ — номер используемой кривой, т.е. не более чем на 3 больше стандартной оценки. Все эти утверждения доказаны в [157].

Глава 6

Протоколы эллиптической криптографии

6.1 Выбор точки и размещение данных

6.1.1 Введение

Использование группы точек эллиптической кривой в криптографии связано с выбором определённых её точек. При этом в зависимости от криптографической задачи выбирают точку случайно или точку, координаты которой отражают данные, помещаемые на кривую. Так значение координаты x может содержать как часть бинарного вектора подпоследовательность x', значение координаты y при этом определяется по уравнению кривой. Последнее связано с решением квадратного уравнения определённого вида. Мы рассмотрим методы решения квадратных уравнений, возникающих при вычислении координаты y точки эллиптической кривой. Затем мы рассмотрим метод помещения данных в эллиптическую кривую.

6.1.2 Решение квадратных уравнений

Суперсингулярный случай. Рассмотрим квадратное уравнение

$$Y^2 + Y = \sigma, \tag{6.1}$$

где σ – элемент поля $GF(2^n)$, такой, что $Tr(\sigma) = 0$.

(Если $Tr(\sigma) = 1$, то уравнение не имеет решения, так как $Tr(y) = Tr(y^2)$ и, следовательно, $Tr(y^2 + y) = 0$. Значения функции следа левой и правой части уравнения после подстановки вместо переменной Y решения y должны быть одинаковыми. [1]

Не трудно видеть, что если y – корень этого уравнения, то $y + 1$ – второй его корень уравнения, так как $y^2 + y = y(y + 1)$.

[1] Здесь мы используем свойство линейности функции следа: $Tr(ax + by) = aTr(x) + bTr(y)$

Решение y как элемент поля $GF(2^m)$ можно представить в полиномиальном базисе в виде вектора

$$(y_0, y_1, \ldots, y_{n-2}, y_{n-1}),$$

а также в виде многочлена – произведения этого вектора коэффициентов на единичную матрицу T_1 и на вектор степеней корня λ неприводимого многочлена.

$$y = (y_0, y_1, \ldots, y_{n-2}, y_{n-1}) \begin{pmatrix} 1 & 0 & \ldots & 0 \\ 0 & 1 & \ldots & 0 \\ \ldots & \ldots & \ldots & \ldots \\ 0 & 0 & \ldots & 1 \end{pmatrix} \begin{pmatrix} 1 \\ \lambda \\ \ldots \\ \lambda^{n-2} \\ \lambda^{n-1} \end{pmatrix} \tag{6.2}$$

Возведением в квадрат каждого терма многочлена (соответствующего i-ой строке матрицы) и последующим приведением результата по модулю неприводимого многочлена получим аналогичное (в виде произведения вектора на матрицу T_2 и затем на вектор степеней корня λ) представление для y^2 :

$$y^2 = (y_0, y_1, \ldots, y_{n-2}, y_{n-1}) \times$$

$$\begin{pmatrix} 1 & 0 & 0 & \ldots & 0 \\ 0 & 0 & 1 & \ldots & 0 \\ \ldots & \ldots & \ldots & \ldots & \ldots \\ s_{n-2,0} & s_{n-2,1} & \ldots & s_{n-2,n-2} & s_{n-2,n-1} \\ s_{n-1,0} & s_{n-1,1} & \ldots & s_{n-1,n-2} & s_{n-1,n-1} \end{pmatrix} \begin{pmatrix} 1 \\ \lambda \\ \ldots \\ \lambda^{n-2} \\ \lambda^{n-1} \end{pmatrix} \tag{6.3}$$

Как видим, i-ая строка матрицы T_1 представляет собой вектор полиномиального представления элемента λ^i поля $GF(2^m)$, тогда как i-ая строка матрицы T_2 представляет в полиномиальном базисе элемент λ^{2i} этого поля.

Сумму $T_1 + T_2$ матриц T_1 и T_2 в представлениях (1.2) и (1.3) обозначим T. Тогда уравнение (1.1) можно записать в матричной форме

$$y \cdot T = \sigma = (\sigma_0, \sigma_1, \ldots, \sigma_{n-1}). \tag{6.4}$$

Первая строка матрицы T оказывается нулевой, но последующие $n-1$ строк соответствуют ровно $n-1$ переменным $y_1, \ldots y_{n-1}$. Этим строкам соответствует матричное уравнение с $n-1$ неизвестными:

$$(y_1, \ldots, y_{n-2}, y_{n-1}) \times$$

$$\begin{pmatrix} 1 & 0 & 1 & \ldots & 0 \\ \ldots & \ldots & \ldots & \ldots & \ldots \\ s_{n-2,0} & s_{n-2,1} & \ldots & 1 + s_{n-2,n-2} & s_{n-2,n-1} \\ s_{n-1,0} & s_{n-1,1} & \ldots & s_{n-1,n-2} & 1 + s_{n-1,n-1} \end{pmatrix} \begin{pmatrix} 1 \\ \lambda \\ \ldots \\ \lambda^{n-2} \\ \lambda^{n-1} \end{pmatrix} = \sigma, \tag{6.5}$$

Переменная y_0 не участвует в этом уравнении и может иметь любое из значений 0,1, что отражает неопределенность решения квадратного уравнения (если

при выборе $y_0 = 0$ получается решение y, то при выборе $y_0 = 1$ получилось бы решение $y + 1$).

Заметим, что при работе с суперсингулярной эллиптической кривой

$$Y^2 + Y = X^3 + X + 1$$

или

$$Y^2 + Y = X^3 + X$$

уравнение (1.1) возникает при подстановке в правую часть уравнения элемента x поля, над которым она определена. При подстановке разных элементов x матрица T вычисляются только один раз для заданного неприводимого многочлена.

Пример 6.1.1 Рассмотрим суперсингулярную эллиптическую кривую

$$Y^2 + Y = X^3 + X + 1$$

над полем $GF(2^4) = GF(2)(\lambda)$, где λ есть корень неприводимого многочлена $x^4 + x + 1$,

Выберем элемент $x = (x_0, x_1, x_2, x_3) = (1, 0, 0, 0)$. Как видим,

$$Tr(x) = x^2 + x^4 + x^8 + x = 0.$$

Вычислим $\sigma(x) = 1 + x + x^3 = (1, 0, 0, 0)$. Как видим, $Tr(\sigma) = Tr(1, 0, 0, 0) = (1, 0, 0, 0) + (1, 0, 0, 0) + (1, 0, 0, 0) + (1, 0, 0, 0) = 0$.

Квадратное уравнение (1.1) принимает вид

$$Y^2 + Y = (1, 0, 0, 0).$$

Примечание. В данном случае матрицы T_1 и T_2 следующие

$$T_1 = \begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix}; \; T_2 = \begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 1 & 1 & 0 & 0 \\ 0 & 0 & 1 & 1 \end{pmatrix}.$$

Матрица $T = T_1 + T_2$ имеет вид

$$T_1 + T_2 = \begin{pmatrix} 0 & 0 & 0 & 0 \\ 0 & 1 & 1 & 0 \\ 1 & 1 & 1 & 0 \\ 0 & 0 & 1 & 0 \end{pmatrix}.$$

Уравнение (1.5) в матричной форме имеет вид

$$(y_0, y_1, y_2, y_3) \begin{pmatrix} 0 & 0 & 0 & 0 \\ 0 & 1 & 1 & 0 \\ 1 & 1 & 1 & 0 \\ 0 & 0 & 1 & 0 \end{pmatrix} \cdot \begin{pmatrix} \lambda^0 \\ \lambda^1 \\ \lambda^2 \\ \lambda^3 \end{pmatrix} = (1, 0, 0, 0).$$

После упрощения имеем матричное уравнение:

$$(y_1, y_2, y_3) \begin{pmatrix} 0 & 1 & 1 & 0 \\ 1 & 1 & 1 & 0 \\ 0 & 0 & 1 & 0 \end{pmatrix} \cdot \begin{pmatrix} \lambda^0 \\ \lambda^1 \\ \lambda^2 \\ \lambda^3 \end{pmatrix} = (1, 0, 0, 0).$$

Умножая матрицу на вектор, получим

$$(y_1, y_2, y_3)\begin{pmatrix} 0 & \lambda & \lambda^2 & 0 \\ 1 & \lambda & \lambda^2 & 0 \\ 0 & 0 & \lambda^2 & 0 \end{pmatrix} = (1,0,0,0).$$

Умножая вектор на матрицу, получаем $(y_2, (y_1+y_2)\lambda, (y_1+y_2+y_3)\lambda^2) = (1,0,0,0)$.
Приравнивая коэффициенты, получаем систему уравнений

$$\begin{gathered} y_2 = 1, \\ y_1 + y_2 = 0, \\ y_1 + y_2 + y_3 = 0. \end{gathered}$$

Решая ее, получим $y_2 = 1,\ y_1 = 1,\ y_3 = 0$.
Таким образом, имеем два решения уравнения $(0,1,1,0)$ и $(1,1,1,0)$.

Заметим, что для решения квадратного уравнения (1) над полем $GF(2^n)$ при нечетном n, можно использовать следующий алгоритм.

Алгоритм 6.1.1

ВХОД: Уравнение 1.1 над $GF(2^n),\ n = 2k+1$.
Неприводимый многочлен $p(x)$, порождающий $GF(2^n)$
ВЫХОД: Решения $y, y+1$ уравнения 1.1
1. Вычислить $T = Tr(\sigma) = \sigma + \sigma^2 + \cdots + \sigma^{2^{n-1}}$ – след элемента σ.
2. Если $T \neq 0$, вернуть «решений нет»
3. Вычислить $y = \sigma + \sum_{i=1}^{i=k} \sigma^{2^{2i-1}}$

Действительно, при любом y след левой части уравнения равен нулю, так как $Tr(y) = Tr(y^2)$. Поэтому, если решение существует, то след правой части также нулевой. Далее не трудно заметить, что если вычислить

$$y = \sigma + \sum_{i=1}^{k} \sigma^{2^{2i-1}},$$

а затем взять $s = y + y^2$, то получится $s = Tr(\sigma) + \sigma = \sigma$, так как след элемента σ, как проверено в п.2 алгоритма, равен нулю. Значит,

$$y^2 + y = \sigma.$$

Пример. При $n = 5$

$$y = \sigma + \sigma^4 + \sigma^{16}.$$

$$y^2 = \sigma^2 + \sigma^8 + \sigma^{32}.$$

$$y + y^2 = \sigma + \sigma^4 + \sigma^{16} + \sigma^2 + \sigma^8 + \sigma^{32} = \sigma + \sigma^4 + \sigma^{16} + \sigma^2 + \sigma^8 + \sigma =$$

$$\sigma + \sigma^2 + \sigma^4 + \sigma^8 + \sigma^{16} + \sigma = Tr(\sigma) + \sigma = \sigma.$$

Например, если $p(x) = x^2 + x^5$ и $\sigma = x^4$, то

$$Tr(\sigma) = x^4 + x^8 + x^{16} + x^{32} + x^{64} = 0,$$

$$y = x^4 + x^{16} + x^{64} = 1 + x + x^2 + x^3;$$

$$y^2 = x^8 + x^{64} = 1 + x + x^2 + x^3 + x^4;$$

$$y + y^2 = x^4.$$

Несуперсингулярный случай. Рассмотрим квадратное уравнение

$$Y^2 + XY + f(X) = 0, \tag{6.6}$$

получающееся при переносе в левую часть правой части f(X) уравнения несуперсингулярной эллиптической кривой. Положим $X = x$, $Y = xZ$ и получим уравнение

$$Z^2x^2 + x^2Z + f(x) = 0,$$

при $x \neq 0$ оно эквивалентно уравнению

$$Z^2 + Z + \sigma = 0, \tag{6.7}$$

где $\sigma = f(x) \cdot x^{-2}$.

Это уравнение (относительно z) можно решить методами, описанными в предыдущем параграфе для уравнения (1.1). Далее это решение можно преобразовать в решение $y = zx$ уравнения (1.5). Заметим, что в этом случае $-z = z + 1$. Отсюда $-y = -zx = (z + 1)x = zx + x = y + x$, как и должно быть в несуперсингулярном случае.

6.1.3 Выбор точки эллиптической кривой

Приведём два вероятностных алгоритма выбора точки эллиптической кривой (суперсингулярный случай.)

Алгоритм 6.1.2

ВХОД: Уравнение суперсингулярной эллиптической кривой над $GF(2^n)$.
Неприводимый многочлен $p(x)$, порождающий $GF(2^n)$
ВЫХОД: Точка (x, y), удовлетворяющая уравнению кривой
0. (На этапе инициализации).
В соответствии с (1.2),(1.3) вычислить матрицу T
1. Выбрать случайно элемент x, $x \neq 0$ поля $GF(2^n)$.
2. Вычислить $\sigma = f(x)$.
3. Вычислить след $Tr(\sigma)$ элемента σ :
$Tr = Tr(\sigma) = \sum_{i=0}^{m-1} \sigma^{2^i}$.
4.Если $Tr = 1$, то перейти к п. 1.
5.Решить уравнение 1.1
6. Вернуть (x,y)

При нечетной степени расширения поля применяется более простой алгоритм, использующий Алгоритм 1.1.

Алгоритм 6.1.3

ВХОД: Уравнение суперсингулярной эллиптической кривой над $GF(2^n)$.
Неприводимый многочлен $p(x)$, порождающий $GF(2^n)$, n=2k+1
ВЫХОД: Точка (x, y), удовлетворяющая (1.1)
1. Выбрать случайно элемент x, $x \neq 0$ поля $GF(2^n)$.
2. Вычислить $\sigma = f(x)$ – значение правой части уравнения эллиптической кривой при X=x.
3. Применить Алгоритм 1.1
Если решение y имеется, то вернуть (x, y).
4. Перейти к п. 1.

Вероятностные алгоритмы выбора точки несуперсингулярной эллиптической кривой незначительно отличаются алгоритмов суперсингулярного случая.

Приведем два алгоритма, использующие алгоритмы 3.3 и 3.4

Алгоритм 6.1.4

ВХОД: Уравнение $Y^2 + XY = f(x)$, где $f(x)$ –
значение правой части уравнения
эллиптической кривой при X=x.
Неприводимый многочлен $p(x)$, порождающий $GF(2^n)$
ВЫХОД: Точка (x, y), удовлетворяющая 1.6
@ 0. (На этапе инициализации).
В соответствии с (1.2),(1.3) вычислить матрицу $T = T_1 + T_2$
1. Выбрать случайно элемент x, $x \neq 0$ поля $GF(2^n)$.
2. Вычислить $\sigma = f(x) \cdot x^{-2}$,где $f(x)$ –
значение правой части уравнения
эллиптической кривой при X=x.
3. Вычислить след $Tr(\sigma)$ элемента σ :
$Tr = Tr(\sigma) = \sum_{i=0}^{m-1} \sigma^{2^i}$.
4.Если $Tr = 1$, то перейти к п. 1.
5.Решить уравнение $y \cdot T = \sigma$
6.Вычислить $y = yx$.
7. Вернуть (x, y).

При нечетной степени расширения поля применяется более простой алгоритм, использующий Алгоритм 1.4.

Алгоритм 6.1.5

ВХОД: Уравнение $Y^2 + XY = f(x)$, где $f(x)$ –
значение правой части уравнения
эллиптической кривой при X=x.
Неприводимый многочлен $p(x)$, порождающий $GF(2^n)$.
ВЫХОД: Точка (x, y), удовлетворяющая 1.6@
1. Выбрать случайно элемент x, $x \neq 0$ поля $GF(2^n)$.
2. Вычислить $\sigma = f(x) \cdot x^{-2}$, где $f(x)$ –
значение правой части уравнения
эллиптической кривой при X=x.
3. Применить Алгоритм 1.1.
Если решение y имеется, то вернуть (x, xy).
4. Перейти к п. 1.

6.1.4 Размещение данных на эллиптической кривой

Заметим, что данные, которые можно разместить в некоторой точке эллиптической кривой должны состоять из несколько меньшего числа бит, чем

элемент конечного поля, над которым строится кривая. (Несколько бит должны оставаться неопределёнными, чтобы обеспечить возможность попадания на кривую). Алгоритмы размещения данных на эллиптической кривой аналогичны алгоритмам выбора точки эллиптической кривой. Их отличие состоит в том, что выбирается не произвольная точка, а точка (x, y) такая, то часть вектора x фиксирована и соответствует размещаемым данным. В предыдущем параграфе были приведены некоторые кривые, пригодные для использования в криптосистеме. Методы выбора кривых даны в стандарте $IEEE\ P1363$.

Пример 6.1.2 Пусть требуется разместить данные $d = (d_1, d_2) = (\mathbf{0}, \mathbf{1})$ на эллиптической кривой из примера 1.1. Прежде всего выберем элемент $x = (x_3, x_2, x_1, x_0)$ такой, что $(x_1, x_0) = (d_1, d_2)$ и $Tr(\sigma) = 0$. Элементы x_1 и x_0 заданы, а x_3 и x_4 подбираются:

$$Tr(0, 0, \mathbf{0}, \mathbf{1}) = 0,$$

и остальные варианты для x_3 и x_2 можно не рассматривать. Данные d можно разместить в точке $(x, y) = ((0, 0, \mathbf{0}, \mathbf{1}), (0, 1, 1, 0))$ или в точке $((0, 0, \mathbf{0}, \mathbf{1}), (0, 1, 1, 1))$, используя решения $y, y + 1$ квадратного уравнения из примера 1.5. @

6.1.5 Определение порядка точки эллиптической кривой и нахождение образующего элемента группы точек эллиптической кривой

Применение алгоритмов для конечных групп. Описанные в параграфе 1.3 алгоритмы можно использовать при определении порядка точки эллиптической кривой и нахождении образующего элемента группы точек эллиптической кривой. Так найти образующий элемент группы точек эллиптической кривой можно, применяя описанный в главе 1 общий алгоритм нахождения образующего элемента конечной группы. Это возможно, если известна факторизация порядка n группы. Пусть $p_1, p_2, \ldots p_k$ – все простые множители порядка p группы точек эллиптической кривой. Обозначим $t_i = \frac{p}{p_i}$. Точка (x, y) является образующим элементом группы точек эллиптической кривой тогда и только тогда, когда

$$\forall i\ t_i \cdot (x, y) \neq \mathcal{O}. \tag{6.8}$$

Алгоритм выбора образующего элемента группы точек эллиптической кривой выбирает случайные точки по соответствующему кривой алгоритму из параграфа 1.3 до тех пор, пока не будет найдена точка, удовлетворяющая условию (1.8) @

Дадим описания алгоритмов из первой главы применительно к группе точек эллиптической кривой. (Предполагается, что операция умножения на константу соответствует типу кривой)

Определение порядка точки эллиптической кривой при известной факторизации порядка n группы

Используется свойство, что порядок элемента делит порядок группы.

Алгоритм 6.1.6

ВХОД: Элемент (x, y) группы точек эллиптической кривой, факторизация порядка группы $n = p_1^{e_1} \cdot p_2^{e_2} \cdots p_k^{e_k}$, где $p_i,\ i = 1, \ldots, k$ – простые числа.
ВЫХОД: порядок t элемента (x, y).
1. Присвоить $t \leftarrow n$.
2. Для i от 1 до k выполнять:
   2.1 Присвоить $t \leftarrow t/p_i^{e_i}$.
   2.2 Вычислить $(\tilde{x}, \tilde{y}) \leftarrow t \cdot (x, y)$.
   2.3 Пока $(\tilde{x}, \tilde{y}) \neq \mathcal{O}$ выполнять $(\tilde{x}, \tilde{y}) \leftarrow p_i \cdot (\tilde{x}, \tilde{y})$ и присваивать $t \leftarrow t \cdot p_i$.
3. Вернуть (t).

Поиск образующего элемента циклической группы Приведём вероятностный алгоритм поиска образующего элемента циклической группы. Эффективность алгоритма определяется тем, что группа содержит $\varphi(n)$ образующих элементов, и вероятность того, что случайно выбираемый элемент является образующим равна $\varphi(n)/n > \frac{1}{6 \ln \ln n}$.

Алгоритм 6.1.7

ВХОД: Уравнение эллиптической кривой, факторизация порядка группы $n = p_1^{e_1} \cdot p_2^{e_2} \cdots p_k^{e_k}$, где $p_i,\ i = 1, \ldots, k$ – простые числа.
ВЫХОД: образующий элемент $\alpha = (x, y)$ группы точек эллиптической кривой.
1. Выбрать случайную точку (x, y) эллиптической кривой.
2. Для i от 1 до k выполнять:
   2.1 Вычислить $\beta = n/p_1 \cdot \alpha$.
   2.2 Если $\beta = \mathcal{O}$, то перейти к 1.
3. Вернуть (α).

6.2 Распределение ключей

6.2.1 Введение

Рассмотрим ряд протоколов, криптографическая стойкость которых основана на трудности решения проблемы дискретного логарифма и проблемы

Диффи-Хеллмана. Проблема дискретного логарифма имеет место в каждом случае, когда задана некоторая циклическая группа, известна степень $y = g^x$ некоторого её элемента (если группа мультипликативная) или кратное $y = x * g$ (в аддитивной группе) и требуется найти значение показателя степени или коэффициент кратности x – дискретный логарифм элемента y по основанию g.

В одних случаях, например для аддитивной группы, заданной на множестве Z_n вычетов по модулю простого числа p, эта проблема легко решаема с использованием алгоритма, подобного алгоритму Евклида, в других случаях, например, для мультипликативной группы Z_p^* известны субполиномиальные алгоритмы для этой проблемы.

Для группы точек эллиптической кривой проблема дискретного логарифма заключается в определении числа k по известным точке P данной эллиптической кривой и точке $Q = kP$. Сложность этой проблемы не меньше сложности проблемы дискретного логарифма в общей постановке для произвольной группы. Исключение составляют суперсингулярные эллиптические кривые, для некоторых из которых проблема дискретного логарифма решается эффективно. Для несуперсингулярных кривых субэкспоненциальные алгоритмы решения проблемы дискретного логарифма неизвестны.

Классическая проблема Диффи-Хеллмана формулируется применительно к мультипликативной группе Z_p^*, где p – простое число. Она заключается в вычислении элемента α^{xy} по элементам α, α^x и α^y. Не известно, возможно ли ее решение без предварительного вычисления индексов x и y, то есть минуя проблему дискретного логарифма. Применительно к циклической подгруппе группе точек эллиптической кривой эта проблема заключается в том, чтобы при известных точке $P \in \mathcal{EF}$ и двух ее кратных k_AP и k_BP найти точку k_Ak_BP. Также неизвестно, можно ли это сделать без предварительного вычисления констант k_A и k_B, то есть не решая проблему дискретного логарифма для эллиптических кривых. Не доказана и гипотеза об эквивалентности проблем дискретного логарифма и Диффи-Хеллмана для эллиптических кривых.

6.2.2 Распределение ключей для классической криптосистемы (протокол Диффи-Хеллмана)

Заметим, что в качестве ключа классической криптосистемы можно использовать неизвестную посторонним (секретную) случайную точку $(x, y) \neq O$ группы точек эллиптической кривой $\mathcal{EF}$, если условиться, как конвертировать ее в натуральное число, например, одну из координат, скажем, x считать двоичной записью натурального числа[2].

Для получения такой секретной точки на двух терминалах открытого канала связи можно использовать модификацию протокола Диффи-Хеллмана. [3]

[2] В общем случае следует иметь в виду некоторое инъективное отображение из $\mathcal{EF} \setminus \{O\}$ в множество натуральных чисел N.

[3] Классическая версия этого протокола основана на проблеме Диффи-Хеллмана для груп-

Допустим, что $\mathcal{E}$ – эллиптическая кривая и P – предварительно согласованная и опубликованная точка этой кривой. Абонент A выбирает, сохраняя в секрете случайное число k_A(секретный ключ A), вычисляет координаты точки k_AP (свою «половинку» ключа) и пересылает их абоненту B. Аналогично B выбирает секретный ключ k_B, вычисляет и пересылает абоненту A «половинку» k_BP ключа. Общим ключом является точка $P = k_Ak_BP$. A вычисляет ее умножая на свой секретный ключ k_A «половинку» ключа, вычисленную B, а B вычисляет эту же точку, умножая сообщение, поступившее от A на свой секретный ключ k_B. Ввиду того, что группа точек эллиптической кривой абелева, результат не зависит от порядка вычисления и, следовательно, A и B имеют координаты секретной точки:

$$k_A(k_BP) = k_B(k_A)P = k_Ak_BP = (x, y)^4$$

и могут использовать x в качестве ключа симметричной криптосистемы (при условии достаточности длины этой двоичной записи, что зависит от степени расширения поля, над которым построена эллиптическая кривая, и при условии, что секретные ключи k_A и k_B были выбраны как случайные или как криптографически стойкие псевдослучайные числа). Теперь A и B имеют одинаковые копии искомой секретной точки эллиптической кривой.

Проблема, стоящая перед посторонним наблюдателем, имеющим намерение узнать секретный ключ, заключается в вычислении k_Ak_BP по известным P, k_AP, k_BP, но при неизвестных k_A, k_B, она и есть проблема Диффи-Хеллмана для эллиптических кривых.

Пример 6.2.1 Как отмечено в 3.3.2 эллиптическая кривая

$$Y^2 + XY = X^3 + X^2 + 1$$

над полем GF(2)(λ), где λ есть корень неприводимого многочлена степени 163, имеет порядок $2 \times P49$. Выберем неприводимый многочлен

$$1 + X + X^2 + X^8 + X^{163}$$

пы Z_p^*. Абоненты A и B, предварительно по открытому каналу уславливаются об использовании большого простого числа p и образующего элемента α мультипликативной группы Z_p^*. Для совместной выработки секретной точки они выбирают независимо друг от друга секретные числа $x \in Z_p^*$ и $y \in Z_p^*$, вычисляют «половинки» α^x и α^y и обмениваются ими по открытому каналу. После этого каждый из них вычисляет секретный ключ, возводя полученную «половинку» ключа в свою секретную степень:

$$(\alpha^x)^y = \alpha^{xy},$$

$$(\alpha^y)^x = \alpha^{yx} = \alpha^{xy}.$$

[4]Предполагается, что полученная точка не есть O. Вероятность получить O при больших степенях расширения поля F чрезвычайно мала, но тем не менее для логической завершенности следует осуществлять проверку и предусматривать возврат на этап выбора секретных ключей k_A и k_B.

и возьмем точку этой эллиптической кривой

$P =$
(101100100100100000101001011100001001001001001001101111110010
101001101100011100111000000110000110001000010001001110010100 1
011011100011001111110011100010110011101,
001100111011100000011011011000110011110000000010010000110010 0
011100011100101010010001100101000000110110000101010001100010 0
001001110110100101001110001011001011100 11);

Проверим, что ее порядок не равен 2: $2P \neq O$. Значит, ее порядок равен порядку $2 \times P49$ группы или числу $P49$, и ее можно использовать для построения ключа.

Пусть $k_A = 12$, $k_B = 123$. (реально должны быть большие числа).

Тогда $k_AP =$
(110110111001111011100110111010111010111000111100101000111011000
000001000010110101110100011100101000101010011010010011111010 0
100111001110111011110010100000111010010 11,
001101101110010101000001011110111110110110000110011100111111101
010011000111100111001101010001011111000000101100101111010001 1
011111011000100111101001010011101001000 11);

$k_BP =$
(010110100110000100100111010000100110000000100010111100111000
010001100111100100011100010010111000111010101000110011000001 1
110011100101110001001001001000010100010111,
101001100001110100101000110011101111000010011111110101111111101
001010100010101011011010000111101110010001000111011111100101 0
001011000101110110000101010111010000 11011);

$k_Bk_AP = k_Ak_BP =$
(110111011110000110100110001101110011011110001100001111101000 1
001100010011010110000100001111000011101001111010110111111000 10
100010001111101011011010001100001110010 1,
111101100110100011111111101111110111100011010101000100111101 10
001101010001011011100011101101011010101010000000110110101001 01
010101101011001011110000111011101 11);

В качестве ключа симметричной системы используется код

$x =$
011000100110101100001000011110000111010011110101101111110001 0
100010001111101011011010001100001110010 1.

Заметим, что для выполнения этого протокола точное знание порядка эллиптической кривой не потребовалось.

6.2.3 Эллиптический вариант криптосистемы Эль Гамаля

Элементы рассмотренного протокола Диффи-Хеллмана с использованием эллиптических кривых можно усмотреть в варианте криптосистемы Эль Гамаля [5] применительно к группе точек эллиптической кривой.

[5] Классический вариант криптоситемы Эль Гамаля формулируется применительно к группе Z_p^*. Внешними параметрами криптоситемы являются простое число p, и образующий элемент α мультипликативной группы Z_p^*. Секретным ключом абонента A является выбираемый им вычет $a \in Z_p^*$, его открытым ключом объявляется тройка $(p, \alpha, \alpha^a \bmod p)$.

Абонент B для передачи абоненту A секретного сообщения $m \in Z_p^*$

Допустим, что множество сообщений представляется точками эллиптической кривой $\mathcal{E}$ ("вложено"в эту кривую условленным способом, например, как это описано в 3.3.3, для упрощения дальнейшего изложения будем считать, что передаваемым сообщением является некоторая точка M эллиптической кривой, которая для сообщения m выбирается известным способом и из которой нужная форма сообщения m также просто получается).

Пусть абонент B намерен переслать абоненту A секретное сообщение m. Для этого можно построить криптосистему Эль Гамаля в эллиптическом исполнении. В качестве внешних параметров выбираются Эллиптическая кривая $\mathcal{E}$, точка P высокого порядка из группы точек $\mathcal{EF}$ и порядок N этой точки.

Абонент A выбирает секретный ключ $k_A \in Z_N^*$, вычисляет и объявляет свой открытый ключ $(\mathcal{E}, P, A)$, где $A = k_A P$.

Абонент B для передачи абоненту A секретного сообщения m

1. получает авторизованную копию открытого ключа $(\mathcal{E}, N, P, A)$;

2. «вкладывает» сообщение m в точку $M \in \mathcal{EF}$;

3. выбирает случайное число $r \in Z_N^*$ (рандомизатор);

4. вычисляет сеансовый ключ $\Delta = r \times A$;

5. вычисляет криптограмму $C = (C_1, C_2) = (rP, M + \Delta)$;

6. отправляет криптограмму C абоненту A.

Для расшифрования криптограммы абонент A, используя свой секретный ключ k_A

1. вычисляет $k_A C_1$ (получает в результате $(k_A \cdot r)P$);

2. обращает результат п.1 $((-k_A \cdot r)P) = ((N-1) \cdot k_A \cdot r)P$ и складывает полученную точку с точкой C_2 , получая при этом точку M :

$$((-k_A \cdot r)P) + M + \Delta = M + (r \cdot k_A)P - (k_A \cdot r)P = M + (r \cdot k_A)P - (r \cdot k_A)P = M.$$

Криптоаналитику известны открытый ключ $(\mathcal{E}, P, A)$ и криптограмма (C_1, C_2), таким образом для получения точки M ему необходимо вычислить точку $(k_A \cdot r)P$. Для этого ему придется решить задачу Диффи-Хеллмана для эллиптической кривой: найти точку $(k_A \cdot r)P$ по известным точкам rP и $A = k_A P$, либо ему придется решать задачи дискретного логарифмирования, вычисляя

---

1. получает авторизованную копию открытого ключа (p, α, β);

2. выбирает случайное число $r \in Z_p^*$ (рандомизатор);

3. вычисляет сеансовый ключ $\delta = \beta^r \bmod p$;

4. вычисляет криптограмму $c = (c_1, c_2) = (\alpha^r, m \cdot \delta) \bmod p$;

5. отправляет криптограмму c абоненту A.

Для расшифрования криптограммы абонент A, используя свой секретный ключ a

1. вычисляет $c_1^a \bmod p$(получает в результате $\alpha^{r \cdot a}$);

2. инвертирует результат п.1 и умножает полученный вычет на c_2 по модулю p, получая при этом сообщение m :

$$\alpha^{-r \cdot a} \cdot c_2 \bmod p = \alpha^{-r \cdot a} \cdot m \cdot \alpha^{a \cdot r} \bmod p = \alpha^{-r \cdot a} \cdot \alpha^{r \cdot a} \cdot m \bmod p = 1 \cdot m = m.$$

секретный ключ k_A и рандомизатор r по точкам k_AP, rP и P известной ему эллиптической кривой $\mathcal{E}$ и известному ему порядку точки P.

Отметим также, что как и в классическом варианте криптосистемы Эль Гамаля повторное использование рандомизатора r недопустимо. Действительно, если криптоаналитику удалось расшифровать одну криптограмму (C_1, C_2) или узнать точку M иным способом, то он легко получит и другие сообщения, зашифрованные с тем же рандомизатором: Пусть (C_1, C_2') криптограмма, полученная с тем же рандомизатором r, что и криптограмма (C_1, C_2). первые точки в этих парах одинаковые, а вторые связаны соотношением:

$$C_2' = C_2 - M + M'.$$

Таким образом,

$$M' = C_2' - C_2 + M.$$

Отметим, что знание порядка N точки P в данном случае необходимо (только для расшифрования), без этого невозможно получить точку $-(k_ArP)$. При зашифровании необходимо знать только максимально допустимый размер PNсообщения, допускающий вложение в кривую. Однако, N не является секретным параметром хотя бы потому, что может быть вычислен по уравнению кривой и степени расширения поля.

Пример 6.2.2

Используем эллиптическую кривую и точку P из примера в предыдущем параграфе. В 3.3.2. указан порядок группы точек для этой кривой над расширением поля $GF(2)$ степени 163.

Определим порядок точки P, используя разложение порядка кривой. Получим $N =$ 5846006549323611672814741753598448348329118574063.

Заведомо допустим размер сообщения PN=48

Пусть сообщение $m =$ 123456789123456789123456789.

Вложим его в эллиптическую кривую, получим точку

$M =$
(11111111111101011000011001000110001100110100111111111100111111010100010
11111001111111011011101011101100011010011010000101100010101101011110010
11101100010011010110101 1,
11001110011001011010001001001110101101100001001000011001010011001000 10
01110101010000011111011011010011101011100110010111111011001010010000000
10001100100101011110001)

Пусть, как и в предыдущем примере $k_a = 12$, тогда $A = k_aP$ возьмем из предыдущего примера. Пусть рандомизатор $r = 123$, тогда первую точку $C_1 = 123P$ криптограммы C также возьмем из предыдущего примера (точка k_BP).

Вычислим сеансовый ключ Δ

$\Delta = rA = 123A =$
(11011101111000011010011000110111001101111000110000111111010001001100010
01101011000010000111100001110100111101011011111100010100010001111101011
01101000110000111001 01,
11110110011010001111111111011111101111000110101010001001111011000110101
00010110111000111011010110101010100000001101101010010101010110101101100 10
11110000111011110010 1)

Вычислим вторую точку криптограммы $C_2 = M + \Delta =$
(1110110010100110100001100001100100011011011001011011011100110010011011
0
1011010011110011111110010101110100000111000110101010110101111000001101
11
1110101001000000001111,
0111000100000001100101101110101100100111111100100111010000000111101100
1
0100100110010101000101011111101000010001111110000010101101101101000111
0
0010001011111111110111)

Для расшифрования умножим первую точку криптограммы на секретный ключ $a = k_A$, получим

$(a \cdot r)P =$
(1101110111100001101001100011011100110111100011000011111010001001100010
01
1010110000100001111000011101001111010110111111000101000100011110101101
10
100011000011100101,
1111011001101000111111111011111101111000110101010001001111011000110101
00
0101101110001110110101101010101000000011011010100101010101101011001011
11
0000111010111100101.)

Обращая полученную точку умножение на $N - 1$, получим

$-(a \cdot r)P = ((N-1)a \cdot r)P =$
(1101110111100001101001100011011100110111100011000011111010001001100010
01
1010110000100001111000011101001111010110111111000101000100011110101101
10
100011000011100101,
0010101110001001010110011000100001001111010110010010110101010001010111
01
1110111101011110011011101111001110101011001011000000100011101011001100
10
0000101100101111.)

Складывая результат обращения со второй точкой криптограммы, получим точку M :

$-(a \cdot r)P + M + \Delta =$
(1111111111110101100001100100011000110011010011111111100111110101000101
11
1100111101101110101110110001101001101000010110001010110101111001011101
10
00
100110101101011,
1100111001100101101000100100111010110110000100100001100101001100100010
01
1101010000011111011011010011101011100110010111111011001010010000001000
11
00
100101011110001.)

Как видим, результат расшифрования оказался правильным.

Между прочим, получение правильного результата свидетельствует о правильности реализации операций в поле $GF(2^{163})$, как и в группе точек несуперсингулярной эллиптической кривой над этим полем.

Подобное заключение по результатам имплементации протокола Диффи-Хеллмана, описанного в предыдущем параграфе, было бы не вполне корректным, поскольку некоторые ошибки реализации, например, операции сложения точек не привели бы к получению разных точек при вычислениях абонентами A и B, что и наблюдалось в процессе разработки программного комплекса.

6.2.4 Распределение ключей для классической криптосистемы (протокол Massey-Omura)

Рассматриваемый ниже эллиптический вариант протокола Massey-Omura [6] позволяет передать сообщение от абонента A абоненту B по открытому каналу связи без предварительной передачи какой бы то ни было ключевой информации. В данном случае системными параметрами являются уравнение эллипти-

[6]Первоначально данный протокол был описан применительно к мультипликативной группе Z_p^*, где p – простое число как аналог передачи секрета с помощью ящиков, запираемых на один или два замка: абонент A запирает ящик с письмом своим ключом и пересылает ящик абоненту B, который запирает ящик своим ключом и отправляет его к A. Последний снимает свой замок и возвращает ящик к B, который снимает свой замок. Вместо механических замков абоненты (два или более) могут использовать электронные, то есть хранимые в компьютерной памяти ключи. Для их организации они выбирают как системный параметр большое простое число p. Затем абоненты A, и B выбирают случайные числа e_A и e_B, взаимно простые с $p-1$, и вычисляют числа d_A и d_B, обратные по модулю числа $\varphi(p)=p-1$ к выбранным ранее числам (φ – функция Эйлера):

$$e_A \cdot d_A \equiv 1 \pmod{\varphi(p)},$$
$$e_B \cdot d_B \equiv 1 \pmod{\varphi(p)}.$$

Пары чисел (e_A, d_A), (e_B, d_B), представляют собой секретные ключи абонентов.
Отметим, что

$$m^{e_A \cdot d_A} = m \bmod p,$$

так как

$$m^{e_A \cdot d_A} = m^{j\cdot\varphi(p)+1} = m^{j\cdot\varphi(p)} \cdot m = m.$$

(Первый сомножитель равен 1 по теореме Эйлера.)
Аналогично,

$$m^{e_B \cdot d_B} = m \bmod p,$$

Пусть абоненту A необходимо послать сообщение m, $0 < m < p-1$, абоненту B (более длинные сообщения разбиваются на блоки). Абонент A шифрует сообщение своим первым ключом, то есть находит

$$m_1 = m^{e_A} \mod p,\ 0 < m_1 < p,$$

и пересылает m_1 к абоненту B. Этот абонент "навешивает"свой первый замок на это сообщение и формирует

$$m_2 = m_1^{e_B} \bmod p = m^{e_A \cdot e_B} \bmod p,\ 0 < m_2 < p,$$

и пересылает m_2 обратно к A. Последний снимает свой первый замок с помощью второго секретного ключа, формируя:

$$m_3 = m_2^{d_A} \bmod p = m^{e_A \cdot e_B \cdot d_A} \bmod p,\ 0 < m_3 < p,$$

Это сообщение пересылается абоненту B, который снимает свой первый замок с помощью своего второго секретного ключа d_B :

$$m_4 = m_3^{d_B} \bmod p = m^{d_B \cdot e_A \cdot e_B \cdot d_A} \bmod p = m.$$

Таким образом, абоненту B доставлено секретное сообщение m от абонента A.

ческой кривой $\mathcal{E}$ и поле $\mathcal{F}$, над которым она построена (поле задается неприводимым многочленом). Этими параметрами определена группа $\mathcal{EF}$ точек эллиптической кривой и ее порядок, который также публикуется как системный параметр (хотя он может быть и вычислен по уравнению кривой и расширению поля $\mathcal{F}$).

Пусть $\mathcal{E}$ – эллиптическая кривая порядка N, e – целое, $1 < e < N$, взаимно простое с N. Используя алгоритм инвертирования, найдём

$$d \equiv e^{-1} (\mathrm{mod}\ N). \tag{6.9}$$

По определению сравнимости по модулю имеем

$$e \cdot d = jN + 1.$$

Поэтому для любой точки P эллиптической кривой $\mathcal{E}$ порядка N

$$(e \cdot d)Q = (j \cdot N + 1)P = (j \cdot N)P + P = jO + P = O + P = P,$$

то есть выполняется тождество

$$(e \cdot d)P = P. \tag{6.10}$$

Используя e и d из (2.1), и любую точку P эллиптической кривой, можно вычислить

$$Q = eP$$
$$R = dQ.$$

Очевидно, что $R = P$.

Заметим, что вычисление точки P по наблюдаемой точке eP эквивалентно решению задачи дискретного логарифмирования для эллиптической кривой.

Это свойство эллиптической кривой приводить константы, на которые она умножается по модулю ее порядка, и используется с учетом трудности проблемы дискретного логарифмирования в эллиптическом варианте протокола Massey-Omura.

Согласно этому протоколу после согласования системных параметров абонент A выбирает как ключ шифрования число e_A, взаимно простое с порядком N эллиптической кривой, и вычисляет по (2.1) мультипликативно обратное число d_A – ключ расшифрования. Аналогично абонент B выбирает число e_B и вычисляет d_B то есть свои ключи.

Абонент A помещает свое сообщение m в некоторую точку $M(m)$ эллиптической кривой и, умножая её на свое секретное значение e_A, получает точку

$$P_1 = e_A M(m).$$

Эту точку A посылает абоненту B.

Он вычисляет

$$P_2 = e_B P_1$$

и посылает результат абоненту A, который снимает свой «замок», вычисляя

$$P_3 = d_A P_2.$$

Он возвращает полученную точку абоненту B.

Последнему остается только расшифровать сообщение, зашифрованное на его ключе шифрования, то есть умножить полученную от A точку на свой секретный ключ расшифрования:

$$M(m) = d_B P_3.$$

Действительно, с учетом коммутативности и ассоциативности операции мультипликативной группы Z_N^*

$$\begin{aligned} &d_B P_3 = (d_B \cdot d_A) P_2 = (d_B \cdot d_A \cdot e_B) P_1 = \\ &= (d_B \cdot d_A \cdot e_B \cdot e_A) M(m) = \\ &(e_B \cdot d_B) \cdot (e_A \cdot d_A) M(m) = M(m). \end{aligned}$$

Сообщение m может быть использовано в качестве ключа симметричной криптосистемы.

Заметим, что в данном случае не требуется опубликования никакой информации о параметрах протокола, кроме самой эллиптической кривой. Платой за это является необходимость трехкратной передачи по открытым каналам.

Пример 6.2.3 Используем ту же несуперсингулярную эллиптическую кривую $\mathcal{E}$, что и в примерах двух предыдущих параграфов. Ее порядок есть
N =1169201309864722334562948350719689669658237148126;
Пусть абоненты A и B выбрали следующие секретные числа в качестве ключей шифрования и расшифрования

$e_A = 12345$,

$d_A = e_A^{-1} \bmod N$ = 436520764452531813689803884039453194612375982306 3;

$e_B = 54321$,

$d_B = e_B^{-1} \bmod N$ = 199935770095053584539224742361797414287767833561 5.

Для передачи секретного сообщения m=987654321987654321 абоненту B абонент A размещает его в точке эллиптической кривой
$M(m) =$
(111111111111010110110111100100011011110011001101101100001111111001101
001010111000101000010101000100011101010100001001110101101001010111011000110
00111101011011111,
010011011110000010100001101110011011010111110010010001010101110010100 1
100001000000001001010010000001011001100001010100011001011001110000100000111
110101011100110101).

Затем он шифрует эту точку и пересылает результат
$P_1 = e_A M(m) =$
(111100001010100111101111110011111010000000111100011101100001101101100100000
001001111001001110000010111001011000001110001010101000100110000010010001111
100100001011,
010011011110000010100001101110011011010111110010010001010101110010100110000
100000000100101001000000101100110000101010001100101100111000010000011111010
1011100110101)
абоненту B. Последний шифрует его своим ключом и возвращает результат
$P_2 = e_B P_1 =$
(101010110111110000101110000101010111000000011000011110010010110001010001111
100101011000111000100110111111010000100101011100001100110011011101100000001011
101001111,
110001001110001101010000001110011111010111100001111101110110000000101100011101
100101111001011001101010111100111010110000001001011110011100111110101010010 1
101111011)
абоненту A, который осуществляет первичное расшифрование (снимает свой «замок») и осуществляет повторную пересылку результата
$P_3 = d_A P_2 =$
(011110111000011011101100000010001010100101101010010001010100100001110011101
011100110101010101110010100001001011010010011101100010111101110100110101010 1
11000000101,
011101010001111111111100101101010001101111100100101000011011000110010100000 0
000011010111011111001010000110010110000000110010110110110010011000110010011 0
110001111);
абоненту B. Последний завершает расшифрование своим ключом, получая точку $M(m)$.

$P_4 = d_B P_3 =$
(111111111111010110110111100100011011110011001101101100001111111001101001010
111000101000010101000100011101010100001001110101101001010111011000110001111 0
1011011111,
010011011110000010100001101110011011010111110010010001010101110010100110000 1
000000001001010010000001011001100001010100011001011001110000100000111110101 0
11100110101)=M(m).
Заметим, что порядок использованной для передачи сообщения точки $M(m) = N/2$.

6.2.5 Протокол распределения ключей Менезеса-Кью-Венстона (MQV-протокол)

Рассмотренные в параграфах 4.1.1 и 4.1.3 протоколы обладают тем недостатком, что некоторое третье лицо C может взять на себя функции посредника в передаче сообщений между двумя абонентами и завладеть при этом их секретом.

Действительно, если A и B взаимодействуют, например, по протоколу Диффи-Хеллмана, то посторонний наблюдатель C, перехватив передачу открытого ключа $k_A P$ абонента A, передаст абоненту B свой открытый ключ $k_C P$, абонент B передаст C свой открытый ключ $k_B P$, после чего B и C будут иметь общий закрытый ключ

$$(k_C \cdot k_B)P. \tag{6.11}$$

Далее если C передаст свой открытый ключ также абоненту A, то C и A будут иметь общий секретный ключ $(k_A \cdot k_C)P$ (A вычислит этот ключ, используя $k_C \cdot P$ вместо $k_B P$)

Однако при хорошо замаскированных действиях C легальные абоненты A и B не будут знать, что имеется посредник, который, получая сообщение одного абонента, способен его расшифровать и вновь зашифровать с использованием другого закрытого ключа.

Для предотвращения таких действий активного криптоаналитика необходима аутентификация (авторизация) этих кратковременных ключей $k_A P$ и $k_B P$ (ключей одноразового использования), для чего используются публикуемые долговременные ключи $d_A P$ и $d_B P$ (ключи многоразового использования). При этом протокол организуется таким образом, что кратковременный открытый ключ связывается с долговременным и поэтому третье лицо, не имеющее долговременного ключа (не зарегистрированное на сервере, где такие ключи хранятся) не сможет стать посредником коммуникаций между двумя абонентами.

Использование кратковременного ключа обеспечивает невозможность использования раскрытого при одной из передач секрета для раскрытия секрета, вырабатываемого при последующих передачах.

Как замечено в предыдущем параграфе,

$$kP = (k \bmod N)P.$$

Поэтому вычисление константы k можно осуществлять как в модульной арифметике кольца Z_N, так и в кольце Z.

Соответственно,возможны две эквивалентные модификации протокола.

В первой используются модульная арифметика целых чисел, вторая основана на циклическом свойстве подгруппы точек эллиптической кривой.

В случае использования модульной арифметики над такими числами могут выполняться операции сложения и умножения по модулю n порядка эллиптической кривой, в случае использования арифметики эллиптической кривой на такие числа могут умножаться точки эллиптической кривой (тогда цикличность определяется порядком подгруппы точек эллиптической кривой и знание порядка эллиптической кривой или этой подгруппы для выполнения операций не требуется).

Во всех случаях абоненты A и B располагают точкой P эллиптической кривой порядка N, над которой и осуществляются все вычисления. Кроме того они знают долговременные и кратковременные ключи друг друга: ключи абонента B :

$$\begin{array}{l} Q_B = d_B P = (a_B, b_B), \\ R_B = k_B P = (x_B, y_B) \end{array} \tag{6.12}$$

известны абоненту A, а ключи абонента A

$$\begin{array}{l} Q_A = d_A P = (a_A, b_A), \\ R_A = k_A P = (x_A, y_A) \end{array} \tag{6.13}$$

известны абоненту B.

Рассмотрим описание и обоснование протокола с использованием как модульной арифметики, так и циклического свойства эллиптической кривой.

Протоколом предусматривается три этапа, симметрично выполняемых каждой из сторон.

На первом этапе A и B вычисляют числа

$$\begin{array}{l} s_A = (k_A + x_A a_A d_A) \bmod N \\ (s_B = (k_B + x_B a_B d_B) \bmod N) \end{array} \tag{6.14}$$

(при этом они используют свои секретные данные $k_A,\ d_A$ и $k_B,\ d_B$ соответственно, а также *интерпретируемые как числа* координаты точек эллиптической кривой.

На втором этапе они вычисляют точки эллиптической кривой

$$\begin{array}{l} s_A = (k_A + x_A a_A d_A) \bmod N \\ (s_B = (k_B + x_B a_B d_B) \bmod N) \end{array} \tag{6.15}$$

Здесь также используются конвертируемые в числовой формат координаты точек эллиптической кривой. На третьем этапе A и B вычисляют общую для них точку эллиптической кривой

$$\begin{array}{l} W = s_A U_A, \\ W = s_B U_B. \end{array} \tag{6.16}$$

Эта точка действительно общая, так как результаты вычислений A и B совпадают:

$$s_A U_A = s_B U_B.$$

Действительно, в соответствии с использованными обозначениями (2.4)-(2.7)@ получим на стороне A :

$$\begin{array}{l} s_A U_A = (k_A + x_A a_A d_A) \bmod N(R_B + x_B a_B Q_B) = \\ = (k_A + x_A a_A d_A) \bmod N(k_B P + x_B a_B d_B P) = \\ = (k_A + x_A a_A d_A) \bmod N(k_B + x_A a_B d_B) P = \\ = (k_A + x_A a_A d_A)(k_B + x_B a_B d_B) P. \end{array}$$

Аналогично получим для стороны B:

$$\begin{array}{l} s_B U_B = (k_B + x_B a_B d_B) \bmod N(R_A + x_A a_A Q_A) = \\ (k_B + x_B a_B d_B) \bmod N(k_A P + x_A a_A d_A P) = \\ (k_B + x_B a_B d_B) \bmod N(k_A + x_A a_A d_A) P = \\ (k_B + x_B a_B d_B)(k_A + x_A a_A d_A) P. \end{array}$$

Как видим, в рассмотренной интерпретации протокола модульная числовая арифметика сочетается с арифметикой эллиптической кривой: точка W вычисляется абонентом A, в конечном итоге, по формуле

$$W = ((k_A + x_A a_A d_A) \bmod N) U_A, \tag{6.17}$$

где используется точка U_A, вычисляемая по формуле (2.7)@. Абонент B получает точку W аналогично.

В варианте, не использующем модульную арифметику, та же точка W получается абонентом A по следующему алгоритму:

1. Вычислить точку U_A по формуле (2.7).
2. Вычислить точку W по формуле

$$W = k_A U_A + x_A(a_A(d_A U_A)).$$

Не трудно видеть, что с учетом модульного свойства умножения точки на константу эта формула эквивалентна формуле (2.9).

Действия абонента B аналогичны.

По окончании исполнения протокола A и B располагают секретной точкой W эллиптической кривой, координаты которой могут быть использованы для построения бинарного кода секретного ключа симметричной системы.

Пример 6.2.4 Рассмотрим имплементацию протокола с использованием той же эллиптической кривой, что и в примере в параграфе 4.1.1. и той же точки P в качестве системного параметра. Порядок кривой (см. 3.3.2) есть N =1169201309864722334562948350719689669665823714812 6.

Пусть абоненты A и B выбрали числа

$$d_A = 345,$$
$$d_B = 4567,$$

и зарегистрировали на сервере свои долговременные ключи

$$Q_A = 345P = (a_A, b_A)$$

$$Q_B = 4567P = (a_B, b_B)$$

Пусть выбраны числа $k_A = 12$ и $k_B = 123$ и вычислены кратковременные ключи

$$R_A = k_A P = 12P = (x_A, y_A)$$

$$R_B = k_B P = 123P = (x_B, y_B)$$

(приведены в 4.1.1), которыми абоненты обменялись. Они получили также долговременные ключи друг друга с сервера.

Затем они вычисляют W вторым из описанных выше способов. Им потребуется конвертировать координаты точек в числовой формат. Это не предусмотрено в алгебраическом процессоре. Придется использовать MATLAB или другую систему символьных вычислений.

6.3 Протоколы цифровой подписи

6.3.1 Электронная цифровая подпись

Электронная цифровая подпись (*Digital signature*) под сообщением m представляет собой некоторый зависящий от этого сообщения, *ключа подписи* k и,

возможно, рандомизатора (случайного кода) r цифровой код $\mathrm{Sign}(m,k,r)$. Обозначим $M,\ K,\ R,\ S$ – множества возможных сообщений, ключей подписи, рандомизаторов и значений цифровой подписи. Тогда цифровую подпись можно рассматривать как отображение

$$\mathrm{Sign}(M,K,R) : M \times K \times R \to S.$$

При фиксированных $m \in M$ и $r \in R$ и при фиксированных $m \in M$ и $k \in K$ отображения $\mathrm{Sign}(m,K,r) \to S$ и $\mathrm{Sign}(m,k,R)$ являются инъекциями, а отображение $\mathrm{Sign}(M,k,r) \to S$ сюръективно. Однако и этому отображению можно придать свойство инъективности, если рассматривать отображения $\mathrm{Sign}(M,K,R)$ вида $\mathrm{Sign}(@H(M),K,R)$, где

$$h(M) : M \to H$$

– хэш-функция, отображающая множество M сообщений в множество H кодов фиксированной длины, и цифровую подпись формировать как значение отображения

$$\mathrm{Sign}(H,K,R).$$

В этом случае подпись рассматривается как подпись под парой

$$(m, h(m)),$$

которую иногда называют сообщением, подготовленным к подписи. Множество $\mathcal{M}_{\mathcal{S}}$ таких пар обладает рядом свойств важных в криптогафическом отношении [7].

1)Мощность множества H много меньше мощности множества $\mathcal{M}_{\mathcal{S}} : |H| << |M \times H| = |\mathcal{M}_{\mathcal{S}}|$ (мощность области определения хэш-функции много больше мощности области ее значений).

2) Каждый элемент h множества H имеет большое число прообразов:

$$|H| << |\mathrm{Im}\ m^{-1}(h)| = |\{m : (m,h) \in \mathcal{M}\}|.$$

3)Легко получить элемент этого множества с заданной первой координатой m, для этого надо вычислить значение $h(m)$ второй координаты по алгоритму вычисления значения хэш-функции. Тем самым легко проверить, принадлежит ли данная пара элементов $(m,h),\ m \in M,\ h \in H$ множеству $\mathcal{M}$. В то же время вычисление первой координаты элемента этого множества по заданной второй координате практически невозможно в следствие свойства односторонности хэш-функции. [8]

[7]Поскольку множество $\mathcal{M}_{\mathcal{S}}$, по существу, является графиком хэш-функции h, то перечисляемые его свойства соответствуют и ее свойствам.

[8]Односторонние функции f определяются в классе функций $f_n : \Sigma^n \to \Sigma^m,\ m = m(n)$, где $m(n)$ – некоторый полином.

4)При заданном элементе $(m, h(m)) \in \mathcal{M}$, практически невозможно подобрать элемент $(m', h(m')) = (m', h(m)) \in \mathcal{M},\ m' \neq m$, то есть элемент, отличающийся от заданного только второй координатой (подобрать второе значение m' аргумента хэш функции при котором она получает то же значение, что и при заданном значении m аргумента.)

5) Практически невозможно подобрать два произвольных элемента (m, h) и (m', h) с одинаковыми значениями второй координаты (два сообщения m и m' с одним и тем же значением хэш-функции $h(m) = h(m')$.

Цифровая подпись сообщения $h(m)$ на ключе подписи k допускает проверку с использованием опубликованного *ключа проверки* k', алгебраически связанного с k.

Проверка основана на предикате $P(S, K')$ проверки, где K' – множество ключей проверки. Цифровая подпись $\mathrm{Sign}(h(m), k, r)$ удостоверяется с использованием ключа проверки k', если

$$P(\mathrm{Sign}(h(m), k, r), k') = 1.$$

Предикат проверки цифровой подписи *с возвратом сообщения* [9] описывается как

$$P(S, K') = \{((m', h'(m)) \in \mathcal{M}_{\mathcal{S}}\},$$

где m' проверяемое сообщение, а $h'(m)$ – хэш-значение подписанного сообщения, извлеченное из цифровой подписи $\mathrm{Sign}(h(m), k, r)$ на заданном ключе проверки $k',\ k' \in K'$.

Отображение

$$\mathrm{Sign}(M, K, R)$$

обладает рядом свойств, гарантирующих возможность и достоверность подтверждения подлинности подписи и, тем самым гарантирующих невозможность отказа от авторства подписавшим документ, как и невозможность вскрытия ключа подписи:

Функция называется *честной,* если существует полином $q(n)$, такой, что $n \leq q(m(n))$. Это означает, что такая функция не слишком сильно «сжимает» входные значения. Честная функция f называется *односторонней,* если

1) Существует полиномиальный алгоритм (алгоритм, исполняющий не более $P(n)$ элементарных операций при вычислении значения функции, $P(n)$ есть некоторый полином) , вычисляющий ее значение $f(x)$ при любом x. Ниже для сокращения подобное условие мы будем формулировать так: «$f(x)$ легко вычислить для всякого x», а если такой алгоритм не существует, будем говорить «практически невозможно вычислить»

?????ДЛИННЫЕ СНОСКИ ВЫЛЕЗАЮТ ЗА КРАЙ 2) Для любого вероятностного алгоритма A и случайно выбранной строки $x \in_R \Sigma^n$ и любого полинома $p(n)$

$$Pr\{f(A(f(x))) = f(x)\} < 1/p(x).$$

(См.[4].)@

[9] Основана на возможности извлечения "отпечатка"$h(m)$ сообщения m из цифровой подписи $\mathrm{Sign}(m, k, r)$.

1) односторонность отображения

$$\mathrm{Sign}(h(m), K, R) :$$

по значению отображения $s = \mathrm{Sign}(h(m), k, r)$, такого, что $P(s, k')$, практически невозможно (при известных $h(m)$ и k') узнать ключ k, а по значению $h(m)$ практически невозможно подобрать другое сообщение m' с тем же хэш-значением $h(m') = h(m)$. Таким образом, при заданном значении $\mathrm{Sign}(h(m), k, r)$ практически невозможно вычислить сообщение m, отображаемое в это значение.

2) для заданного сообщения m и известного значения цифровой подписи $s = \mathrm{Sign}(h(m), k, r),\ P(s, k') = 1$, практически невозможно подобрать другое (фальсифицированное) сообщение m' с тем же значением $\mathrm{Sign}(h(m'), k, r) = \mathrm{Sign}(h(m), k, r)$ цифровой подписи.

3)Практически невозможно найти два произвольных сообщения m и m' с одинаковым значением подписи s, таким, что

$$P(s, k') = 1,$$

то есть удовлетворяющих предикат проверки на заданном ключе проверки k'.

4)Не зная ключ подписи, практически невозможно найти произвольное сообщение m и правильное значение цифровой подписи под ним.

Эти свойства обеспечиваются использованием криптографически стойкой хэш-функции, обладающей перчисленными выше свойствами, а также биективных преобразований, соответствующих трудным алгебраическим проблемам.

6.3.2 Обобщенная схема электронной подписи Эль Гамаля

Обобщенная схема электронной подписи Эль Гамаля работает в любой абелевой группе.

Публикуемыми системными параметрами являются описание циклической группы G порядка N и образующий элемент α этой группы.

Ключ подписи выбирается как целое число k, $0 < k < N$.

Публикуемый ключ проверки вычисляется как элемент $k' = \beta = \alpha^k$ ($k' = \beta = k\alpha$) группы G.[10]

Цифровой подписью под документом m является пара

$$(c, d),$$

где

$$c = \rho = \alpha^r (c = r\alpha)$$

– случайный элемент группы G, определяемый случайным выбором рандомизатора $r,\ 0 < r < N - 1$;

$$d = r^{-1}\{h(m) - kh(c)\} \bmod N$$

[10]Предполагается, что группа G мультипликативна, в скобках здесь и ниже указываются определения для случая аддитивной группы G

– число, вычисляемое с использованием ключа подписи k, того же рандомизатора r и значений $h(m)$ хэш-функции от сообщения m и $h(c)$ от соответствующего рандомизатору случайного элемента $c = \rho$ группы G (код описания этого случайного элемента рассматривается как значение аргумента хэш-функции).

Предикат проверки цифровой подписи на документе m, полученной на ключе подписи k, описывается следующим образом:

$$P(S, K') = \{(c, d) : c \in G, 0 < d < N - 1, \beta^{h(c)} c^d = \alpha^{h(m)}\}$$
$$(P(S, K') = \{(c, d) : c \in G, 0 < d < N - 1, h(c)\beta + dc = h(m)\alpha\}).$$

Если подпись вычислена абонентом, владеющим секретным ключом k, то данный предикат выполняется. Действительно, в этом случае

$$d \equiv r^{-1}\{h(m) - kh(c)\} \bmod N.$$

Умножив обе части сравнения на r, получим

$$rd \equiv h(m) - kh(c) (\bmod\ N),$$

что эквивалентно сравнению

$$h(m) \equiv kh(c) + rd\ (\bmod\ N).$$

Отсюда следует

$$\alpha^{h(m)} = \alpha^{kh(c)+rd} =$$
$$= (\alpha^k)^{h(c)}(\alpha^r)^d = \beta^{h(c)} c^d.$$
$$(h(m)\alpha = (kh(c) + rd)\alpha = h(c)(k\alpha) + d(r\alpha) = h(c)\beta + dc).$$

Если же ключ подписи другой, то предикат имеет значение 0.

Чтобы подделать подпись под сообщением m' злоумышленник вынужден взять случайное число r и вычислить $c = \alpha^r$ с тем, чтобы затем определить $@d = r^{-1}h(m) - kr$. Для этого надо из $\beta = \alpha^k$ найти ключ подписи k, то есть найти дискретный логарифм от β по основанию α, что практически невозможно. Остается выбирать d наугад с вероятностью успеха $\frac{1}{N}$.

Заметим, что генерация подписи требует вычислений как в группе G, так и в группе Z_n, в то же время проверка подписи связана с вычислениями только в группе G.

Число r должно уничтожаться сразу после вычисления подписи, так как по этому числу и значению подписи (c, d) под известным сообщением m вычисляется секретный ключ:

$$k = (h(m) - rd)^{-h(c)} \bmod N$$

Это же возможно в случае повторного использования числа r, так как в этом случае оно с большой вероятностью вычисляется: пусть с использованием

одного и того же числа r получены две подписи (c_1, d_1) и (c_2, d_2), $c_1 = c_2 = \alpha^r = c$ под сообщениями m_1 и m_2. При этом

$$d_1 = r^{-1}\{h(m_1) - kh(c)\}\mathrm{mod}N,$$

$$d_2 = r^{-1}\{h(m_2) - kh(c)\}\mathrm{mod}\ N.$$

Тогда

$$(d_1 - d_2)@k \equiv (h(m_1) - h(m_2))(\mathrm{mod}\ N).$$

При $d_1 \neq d_2$ получаем $@k = (d_1 - d_2)^{-1}(h(m_1) - h(m_2))\ \mathrm{mod}\ N$.

В алгоритме подписи используется не само сообщение, а значение хэш-функции от него. Иначе возможен подбор сообщения с известным значением подписи (то есть не выполняется четвертое свойство цифровой подписи). Например, можно выбрать случайные числа $i, j,\ 1 < i < N,\ 1 < j < N,\ (j, N) = 1$, и положить

$$c = \alpha^i\beta^j = \alpha^{i+kj};$$

$$d = -h(c)j^{-1}\mathrm{mod}\ N,$$

Тогда пара (c, d) является подписью под сообщением

$$m = di\ \mathrm{mod}\ N = -h(c)ij^{-1}\mathrm{mod}\ N,$$

так как

$$(\alpha^m\alpha^{-kh(c)})^{d^{-1}} = \alpha^i\beta^j = c.$$

Действительно,

$$@(\alpha^m\alpha^{-kh(c)})^{d-1} = (\alpha^{-rij^{-1}}\alpha^{-ah(c)})^{(-h(c)j^{-1})^{-1}} =$$

$$= \alpha^{-h(c)ij^{-1}(-h(c))^{-1}j}\alpha^{-kh(c)(-h(c))^{-1}j} = \alpha^i\alpha^{kj} = \alpha^{i+kj} = c.$$

Теперь можно получить

$$\alpha^x\alpha^{a^r} \equiv r^s,$$

откуда следует подтверждение подписи (напоминаем, что $\alpha^a = y$):

$$y^r r^s \equiv \alpha^{ar} :$$

В предикате проверки подписи предусматривается проверка, что $c \in G$. Если эту проверку не делать, то в некоторых случаях, например, при построении протокола на основе группы Z_p^* (порядка $N = p - 1$). злоумышленник может подписать выбираемое им сообщение m', если располагает подписанным на секретном ключе k сообщением m. Пусть (c, d) – подпись под сообщением m. Допустим, что существует $m^{-1}\ \mathrm{mod}\ (p-1)$. Тогда можно вычислить

$$u = m' \cdot m^{-1}.$$

Затем по китайской теореме об остатках можно вычислить

$$d' = du \bmod N$$

и c', такое, что

$$c' \equiv cu \bmod N \text{ и } c' \equiv c \pmod{N+1}.$$

Пара (c', d') является подписью под сообщением m', которая подтверждается предикатом проверки подписи и сообщение m' будет принято, если указанная проверка игнорируется.

Рассмотренная схема наиболее удачно может быть реализована при использовании в качестве группы G группы точек эллиптической кривой над конечным полем $\mathcal{F}_q$. Проблема дискретного логарифма в этой группе гораздо сложнее, чем в мультипликативной группе конечного поля $\mathcal{F}_q$. Отсюда следует, что может быть выбрано меньшее q, чем в случае имплементации в группе $\mathcal{F}_q^*$.

В этом случае системными параметрами являются уравнение эллиптической кривой $\mathcal{E}$, и описание поля $\mathcal{F}$, точка P кривой известного большого порядка N – образующий элемент подгруппы $G \subseteq \mathcal{EF}$ группы точек эллиптической кривой. Публикуемым ключом проверки k' является точка $Q = kP$ эллиптической кривой. Цифровая подпись

$$(c, d) = (R, d),$$

где

$$c = R = rP$$

– случайная точка, элемент группы G, определяется случайным выбором рандомизатора $r, 0 < r < N - 1$;

$$d = r^{-1}\{h(m) - kh(R)\} \bmod N$$

число, вычисляемое с использованием ключа подписи k, того же рандомизатора r и значений $h(m)$ хэш-функции от сообщения m и $h(R)$ от соответствующей рандомизатору случайной точки $R = (x, y)$ группы G (конкатенация $x||y$ координат x и y этой точки рассматривается как значение аргумента хэш-функции).

Предикат проверки цифровой подписи на документе m, полученной на ключе подписи k, описывается следующим образом:

$$(P(S, K') = \{(R, d) : R \in G, 0 < d < N - 1, h(c)Q + dR = h(m)P\}).$$

Если генерация подписи требует вычислений как в группе $\mathcal{EF}$ так и в группе Z_n, то проверка подписи связана с вычислениями только в группе $\mathcal{EF}$.

Алгебраически и криптографически эквивалентным рассмотренному варианту цифровой подписи является вариант, отличающийся тем, что вместо точки $c = R$ эллиптической кривой в качестве первого элемента цифровой подписи

берется число $c = h(R)$, хэш-значение от этой точки. При вычислении второго числа подписи не производится умножение на число, r^{-1} :

$$d = (h(m) - kh(R)).$$

Соответственно упрощается и предикат проверки подписи:

$$(P(S, K') = \{(c, d) : 0 < s < N - 1, 0 < d < N - 1, cQ + dP = h(m)P\}).$$

В данном случае алгоритм получения подписи использует только операции модульной арифметики, а алгоритм проверки- только операции группы точек эллиптической кривой.

Последний вариант подписи используется в алгоритме ECDSA, применяемом в американском стандарте электронной подписи.

Пример 6.3.1 Пусть группа G есть мультипликативная группа Z_p^*, α – образующий элемент этой группы, k – секретный ключ подписи, $(\alpha, \beta = \alpha^k)$ – публикуемый ключ проверки подписи. $k \in Z_p^*$, r – случайно выбранный элемент группы Z_p^*. Тогда цифровая подпись под сообщением m $s = (c, d)$ это пара чисел,

$$c = \alpha^r, d = (m - ks)r^{-1}.$$

Множество истинности предиката проверки $P(S, K')$ есть множество пар

$$\{((c, d), k') : \ c \in Z_n * n, \ d \in Z_n^*, \ \beta^c \times c^d = \alpha^m, \ \beta \in K'\}.$$

Как отмечено в [?] другие схемы цифровой подписи, аналогичные рассмотренной, отличаются проверяемым сравнением вида

$$\alpha^A \beta^B \equiv r^C (\mathrm{mod}\ p),$$

где тройка (A, B, C) совпадает с одной из перестановок чисел $\pm m, \pm s, \pm r$ при некотором выборе знака. Например, описанная схема цифровой подписи Эль-Гамаля получается при $A = m, \ B = -s, \ C = t$.

В американском стандарте DSS используются значения $A = m, \ B = s, \ C = t$.

В российском стандарте $A = -m, \ B = s, \ C = t$.

В схемах даного семейства возможно сокращение длины подписи путём замены пары чисел (r, s) парой $(r \bmod qs \bmod q)$, где q является некоторым делителем числа $p - 1$. При этом проверяемое сравнение заменяется модифицированным равенством

$$(\alpha^A y^B \mathrm{mod}\ p) \mathrm{mod}\ q = r^C \mathrm{mod}\ q.$$

Это применено в американском стандарте DSS.

6.3.3 Электронная подпись Эль Гамаля с возвратом сообщения. Схема Nyberg-Rueppel электронной подписи с использованием группы точек эллиптической кривой

Пусть $\mathcal{EF}$ группа точек эллиптической кривой, P – базовая точка открытого ключа, N – порядок этой точки, k – секретный ключ подписывающего документ

участника. Открытым ключом последнего является точка

$$Q = kP. \tag{6.18}$$

Пусть $e = h(m)$ - значение хеш-функции h для документа m.

Алгоритм генерации подписи следующий:

1) Взять случайное число $r,\ 0 < r < N$, Такое, что x-компонента точки

$$R = rP = (x, y) \tag{6.19}$$

не равна 0.

2) Используя x-компоненту точки R как целое число, вычислить

$$c = x + e \ (\text{mod } N), \tag{6.20}$$

$$d = r - kc(\text{mod } N). \tag{6.21}$$

Если $c = 0$ или $d = 0$, то вернуться к шагу 1.

Пара (c, d) является подписью для документа m, такого, что $h(m) = e$.

Для проверки, что $h(m)$ является корректным хеш-значением, выполняются следующие действия:

1) Проверить, что $1 < c < N - 1,\ 1 < d < N - 1$.

2) Вычислить

$$R' = dP + cQ. \tag{6.22}$$

3) Интерпретируя x-компоненту точки R' как двоичную запись целого числа, вычислить

$$e' = c - x'(\ \text{mod } n). \tag{6.23}$$

3) если полученное значение e' совпадает с хеш-значением $h(m')$, вычисленным для полученного сообщения m', то последнее удостоверяется.

Поясним этот протокол следующим примером (при заведомо малых значениях параметров).

Пример 6.3.2 Выберем несуперсингулярную кривую

$$Y^2 + XY = X^3 + X^2 + 1$$

над полем $GF(2)(\lambda)$, где λ – корень неприводимого многочлена $1 + X^2 + X^5$, и базовую точку

$$P = (00101, 10110)$$

этой кривой. Непосредственной проверкой убедимся, что порядок N этой точки равен 22. Пусть значение хэш-функции сообщения m есть $e = h(m) = (1011) = 13$.[11] Допустим, что секретным ключом является двоичный код k=(111)=7, тогда открытым ключом является точка

$$Q = kP = 7(00101, 10110) = (10011, 10111).$$

[11] Элементы двоичных кодов располагаются в порядке возрастания степеней или весовых эквивалентов

Для получения подписи сначала базовая точка P умножается на случайно выбираемый рандомизатор r, пусть $r = 5$, и получается точка

$$R = 5P = (10111, 11011);$$

x-компонента (10111)=29 этой точки R также является случайным числом.

Прибавление по (3.3) этой точки к хеш-значению $e = h(m) = 13$ по модулю $N = 22$ (порядка точки P) эффективно маскирует это хеш-значение,в результате получается первое число кода цифровой подписи

$$c = (x + e) \bmod N = 29 + 13 \bmod 22 = 20.$$

Второе число d кода цифровой подписи получается по (3.4) с использованием секретного ключа k :

$$d = (r - kc) \bmod N = (5 - 7 \cdot 20) \bmod 22 = 19.$$

Цифровая подпись под значением $e = h(m)$ на ключе подписи $k = 7$ есть пара чисел $(c, d) = (20, 19)$.

Этап верификации по 3.5 позволяет восстановить точку R и, следом, по 3.6 получить замаскированное хеш-значение e : Если подпись корректна, то получится $R' = R$:

$$R' = dP + cQ = dP + ckP = (d + ck) \bmod N \; P =$$

$$= (r - (x + e)k + (x + e)k) \bmod N \; P = rP = R.$$

В нашем примере

$$R' = dP + cQ = 19(00101, 1011) + 20(10011, 10111) = (10111, 11011) = R.$$

Используя x-координату $x' = (10111 = 29)$ точки R, восстановим хэш-значение

$$e' = c - x' = 20 - 29 \bmod 22 = 13.$$

Если это восстановленное значение e' совпадает с хеш-значением $h(m')$, вычисленным по полученному сообщению m', то можно считать, что последнее мог подписать только обладатель секретного ключа s и что ни сообщение, ни его хэш-значение не было изменено активным криптоаналитиком или вследствие ошибок при передаче или хранении.

Заметим, что при верификации операции умножения модульной арифметики не используются.

В основе российского стандарта цифровой подписи с возвратом сообщения используется другая схема генерации и проверки подписи:

цифровой подписью под сообщением m со значением хэш-функции $e = h(m)$ на ключе подписи k является пара чисел (c, d), где c есть отличное от нуля число, определяемое $x-$координатой точки $R = kP$, а число d вычисляется как

$$d = xk + re \bmod N.$$

Для проверки цифровой подписи восстанавливают точку R, используя операции арифметики эллиптической кривой и модульной арифметики:

$$R' = z_1 P + z_2 Q, \tag{6.24}$$

где

$$z_1 = d\nu \bmod N,$$

$$z_2 = -c\nu \bmod N$$

при $\nu = e^{-1} \bmod N$.

В данном случае на этапе проверки цифровой подписи приходится выполнять операции умножения и мультипликативного обращения в группе Z_N^*. Порядок N базовой точки P в данном случае есть простое число.

Пример 6.3.3 Возьмем базовую точку $P = (0001, 11111)$ той же кривой, что и в предыдущем примере и образуем цифровую подпись под сообщением m с хэш-значением $e = h(m) = 2$, (то же, что и в предыдущем примере, но приведенное по модулю $N = 11$) на том же ключе подписи $k = 7$ и с тем же рандомизатором $r = 5$. Порядок N точки P есть простое число 11.

Точка ключа проверки есть

$$Q = kP = 7(0001, 11111) = (01111, 10101)$$

"Точка возврата"

$$R = rP = 5(0001, 11111) = (0101, 01001).$$

Первое число цифровой подписи $c = (0101)_2 = (10)_{10}$,
Второе число цифровой подписи
$d = (ck + re) \mod N = (10 \cdot 7 + 5 \cdot 2) \bmod 11 = 3$.
Цифровая подпись есть пара чисел

$$(c, d) = (10, 3)$$

Для проверки цифровой подписи вычислим
$\nu = e^{-1} \bmod N = 2^{-1} \bmod 11 = 6$
$z_1 = d\nu \bmod N = 3 \cdot 6 \bmod 11 = 7$,
$z_2 = -c\nu \bmod N = -10 \cdot 6 \bmod 11 = 6$.
Восстановленная точка возврата вычисляется в соответствии с (3.7)
$R' = z_1P + z_2Q = 7(0001, 11111) + 6(01111, 10101) = (01111, 10101) + (11001, 10101) = (0101, 01001) = (x', y')$. Как видим, $x' = c$.

6.4 Скрытая передача

6.4.1 Скрытая передача с использованием мультипликативной группы

Скрытой передачей называется общение между двумя участниками по следующим правилам.

1. Участник A передает участнику B два сообщения m_1 и m_2, из которых B может прочитать только одно.

2. Участник A не знает, какое именно сообщение прочитал B.

3. Оба участника уверены, что условия 1 и 2 выполнены.

Для реализации скрытой передачи A и B используют конечную группу $G = F_q^*$ с трудной проблемой дискретного логарифма, образующий элемент

$b \in G$, а также получаемый обоими участниками от центра доверия элемент C с неизвестным им дискретным логарифмом.

Используется отображение $\psi : F_q^* \to F_2^n$, имеющее обратное отображение $\psi^{-1}\mathrm{Im}_\psi \to F_q^*$. (Например, $\psi(x)$ есть двоичная запись элемента x группы, если q-простое или конкатенация двоичных записей его координат, если $q = p^m$, где p – характеристика поля F_q,)

Протокол скрытой передачи следующий.

1. B выбирает секретный ключ $(x, i), 0 < x < q-1,\ i \in \{1, 2\}$, вычисляет $\beta_i = b^x$ и $\beta_{3-i} = Cb^{-x}$ и пересылает к A открытый ключ (β_1, β_2).

(B не знает дискретный логарифм xk' от элемента β_{3-i}, иначе он знал бы и дискретный логарифм $k = x + k' \bmod q - 1$ от элемента $C = kb = \beta_1\beta_2$.)

2. A выбирает два разных числа $0 < y_1, y_2 < q-1$, два разных двоичных сообщения m_1 и m_2 и посылает к B две пары элементов группы F_q^*

$$(c_1, c_2) = (b^{y_1}, b^{y_2})$$

и

$$(\alpha_1, \alpha_2) = (m_1 + \psi(\beta_1^{y_1}), m_2 + \psi(\beta_2^{y_2}).$$

3. B вычисляет $m_i = \alpha_1 + \psi(c_i^x) = \alpha_i + \psi(b^{y_i x}) = \alpha_i + \psi(\beta_i^{y_i})$.

Примечание B не может вычислить $\alpha_{3-i} + \psi(\beta_{3-i}^{y_{3-i}})$, так как он знает только $b^{x'} = Cb^{-x}$ и $b^{y_{3-i}}$, и чтобы узнать $\beta_{3-i}^{y_{3-i}} = b^{x' y_{3-i}}$, ему необходимо решить проблему Диффи-Хеллмана.

Удостоверившись, что $C = \beta_1\beta_2$, участник A может проверить, что B действует, не зная дискретного логарифма от C. С другой стороны, сохраняя в секрете x и i, B уверен, что A не может различить, каким из двух элементов является элемент β_1, то есть условия 1 и 2 выполнены.

6.4.2 Использование группы точек эллиптической кривой

Для реализации скрытой передачи A и B используют группу точек эллиптической кривой $\mathcal{EF}$ над полем $\mathcal{F}$, образующий элемент $P \in \mathcal{EF}$ высокого порядка N, q а также получаемую обоими участниками от центра доверия точку $Q = kP$ с неизвестным им дискретным логарифмом k.

Используется отображение $\psi : \mathcal{EK} \to F_2^n$, где $n = \lceil \log_2 |F| \rceil$, имеющее обратное отображение $\psi^{-1}\mathrm{Im}_\psi \to \mathcal{EF}$. (Например, $\psi(x)$ есть двоичная запись элемента x точки (x, y), Тогда $\psi^{-1}(x) = (x, y) \in \mathcal{EF}$, где (x, y) вычисляется алгоритмом размещения данных x на кривой $\mathcal{EF}$.

Протокол скрытой передачи следующий.

1. B выбирает секретный ключ $(x, i), 0 < x < N,\ i \in \{1, 2\}$, вычисляет $\beta_i = xP$ и $\beta_{3-i} = Q - xP$ и пересылает к A открытый ключ (β_1, β_2).

(B не знает дискретный логарифм x' от элемента $\beta_{3-i} = k'P$, иначе он знал бы и дискретный логарифм $k = x + k'$ от элемента $Q = kP = \beta_1 + \beta_2 = (x + k')P$.

2. A выбирает два разных числа $0 < y_1, y_2 < q-1$, два разных двоичных сообщения m_1 и m_2, выбирая их из $GF(2^n)$, и посылает к B пару элементов группы $\mathcal{EK}$

$$(C_1, C_2) = (y_1 P,\ y_2 P)$$

и пару полиномов из $GF(2^n)$

$$(\alpha_1, \alpha_2) = (m_1 + \psi(y_1\beta_1), m_2 + \psi(y_2\beta_2)).$$

3. B вычисляет

$$m_i = \alpha + \psi(xC_i) = \alpha_i + \psi(xy_iP) = \alpha_i + \psi(y_i\beta_i).$$

Примечание. B не может вычислить $m_{3-i} = \alpha_{3-i} + \psi(y_{3-i}\beta_{3-i})$, так как он знает только $k'P = Q - xP$ и $y_{3-i}P$, и чтобы узнать $y_{3-i}\beta_{3-i} = k'y_{3-i}P$, ему необходимо решить проблему Диффи-Хеллмана.

Удостоверившись, что $Q = \beta_1 + \beta_2$ участник A может проверить, что B действует, не зная дискретного логарифма от Q. С другой стороны, сохраняя в секрете x и i, B уверен, что A не может различить, каким из двух элементов является элемент β_1, то есть условия 1 и 2 выполнены.

Пример 6.4.1 Выберем несуперсингулярную кривую

$$Y^2 + XY = X^3 + X^2 + 1$$

над полем $GF(2)(\lambda)$, где λ – корень неприводимого многочлена $1 + X^2 + X^5$, и базовую точку

$$P = (00101, 10110)$$

этой кривой. Непосредственной проверкой убедимся, что порядок N этой точки равен 22. Допустим, что секретным ключом доверенного центра является двоичный код k=(111)=7, тогда открытым ключом является точка

$$Q = kP = 7(00101, 10110) = (10011, 10111).$$

Дискретный логарифм k этой точки по основанию P участникам неизвестен.

Рассмотрим возможную реализацию протокола скрытой передачи с указанными системными параметрами.

1)Пусть участник B выбрал ключ $(x, i) = (19, 1)$ и вычислил

$$\beta_i = \beta_1 = xP = 19(00101, 10110) = (01101, 00101)$$

$$\beta_{3-i} = \beta_2 = Q - xP = (10011, 10111) + (01101, 01000) = (01111, 11010).$$

Он посылает участнику A открытый ключ

$$(\beta_1, \beta_2) = (01101, 00101), (01111, 11010)).$$

2) Участник A выбирает два числа

$$y_1 = 5,\ y_2 = 16$$

и два двоичных сообщения

$$m_1 = (10001),\ m_2 = (11100)$$

длины n=5. Он вычисляет пару точек

$$(C_1, C_2) = (y_1P, y_2P) =$$

$$(5(00101, 10110), 16(00101, 10110) = ((10111, 11011), (11001, 00111))$$

и пару полиномов

$$(\alpha_1, \alpha_2) = (m_1 + \psi(y_1\beta_1),\ m_2 + \psi(y_2\beta_2))) =$$

$$= (10001 + \psi(5(01101, 00101)), 11100 + \psi(16(01111, 11010))) =$$

$$(10001 + \psi((10011, 10111)), 11100 + \psi(11001, 1111))$$

$$((10001 + 10011), (11100 + 11001)) = (00010, 00101).$$

Эти две пары он посылает участнику B.

3. B вычисляет

$$m_i = \alpha_i + \psi(xC_i) = \alpha_1 + \psi(xC_1) =$$

$$= 00010 + \psi(19(10111, 11011)) = 00010 + \psi((10011, 10111)) =$$

$$00010 + 10011 = 10001.$$

Рассмотрим два примера использования канала скрытой передачи.

Пример 6.4.2 Электронная жеребьевка. Участники A и B договариваются (с помощью доверенного центра) о трактовке бинарного результата жеребьевки и осуществляют протокол скрытой передачи. Заметим, что к моменту получения единственного сообщения от A участник B уже не может изменить выбранного им ранее значения бита i. Для определения результата жеребьевки участник B возвращает участнику A полученное им сообщение m_i, тем самым он сообщает ему значение случайно выбранного параметра i : участник A узнает это значение, сравнив полученное от B сообщение m_i с m_1 и m_2. Результатом жеребьевки является бит $j = (X^{i-1} + m_i \bmod 1 + X$. Этот результат вычисляет каждый участник. Например, если после получения сообщения 10001 участник B отправит его участнику A, то A, сравнив его с отправленными сообщениями, узнает, что $i = 1$, теперь оба участника вычислят результат жеребьевки $j = 1 + 10001 \bmod (1 + X) = X^5 \bmod 1 + X = 1$.

Пример 6.4.3 Доказательство с нулевым разглашением знания факторизации числа $n = pq$.

1. B выбирает случайно целое число x и вычисляет $y = x^2 \bmod n$ и посылает y к A.

2. A вычисляет четыре квадратных корня $\pm x,\ \pm x'$ и выбирает один из них, обозначая его x_0.

3. A выбирает случайное число $r \neq 0$ и посылает к B число $s = r^2 \bmod n$. Далее A вычисляет два сообщения $m_1 = r \bmod n$ и $m_2 = x_0 r \bmod n$ и пересылает их к B по каналу скрытой передачи.

4. B вычисляет m_i и проверяет, что $m_i^2 = s$, если $i = 1$, или $m_i^2 = ys$, если $i = 2$.

5.Шаги $1 - 4$ повторяются (с различными открытыми ключами (β_1, β_2)). Если A выдерживает T проверок, то B убежается в том, что A знает разложение n с вероятностью $1 - T^{-1}$.

6.4.3 Неинтерактивное доказательство с нулевым разглашением

Передачи открытого ключа от B могут быть осуществлены заранее в доверенный центр, который по очереди передает их к A, сопровождая вопросами, направляемыми к A и B (при доказательстве знания разложения это вычеты $x^2 \bmod n$). В каждой итерации данные передаются только от A к B.

Упражнение. Описать неинтерактивные протоколы доказательства знания одного дискретного логарифма и разложения n. Описать процедуры моделирования, доказывающие, что эти доказательства имеют свойство нулевого разглашения.

Глава 7

Компьютерный практикум

Предметный указатель

Литература

[1] Алферов А.П., Зубов А.Ю., Кузьмин А.С., Черемушкин А.В. Основы криптографии. М.: Гелиос АРВ, 2001.

[2] Анин Б. Защита компьютерной информации. СПб.: BHV, 2000.

[3] Арлазоров В.Л., Диниц Е.А., Кронрод М.А., Фараджев И.А. Об экономном построении транзитивного замыкания графа. ДАН СССР, 194 N 3(1970), 487-488.

[4] Айерленд К., Роузен М. Классическое введение в современную теорию чисел. М.: «Мир»,1987.

[5] Ахо А.,Хопкрофт Д.,Ульман Д. Построение и анализ вычислительных алгоритмов. М.: «Мир»,1979.

[6] Бабаш А.В., Шанкин Г.П. Криптография. М.: «Солон-Р», 2002.

[7] Бабаш А.В., Шанкин Г.П. История криптографии. М.: Гелиос АРВ, 2002.

[8] Берлекемп Э. Алгебраическая теория кодирования. Мир, Москва, 1971.

[9] Биркгоф Г., Барти Т. Современная прикладная алгебра.// М.: «Мир».1972.

[10] Блейхут Р. Быстрые алгоритмы цифровой обработки сигналов.//М.:«Мир» 1989.

[11] Болотов А.А., Гашков С.Б., Фролов А.Б., Часовских А.Б. Алгоритмические основы эллиптической криптографии. Изд-во МЭИ, Москва, 2000.

[12] Болотов А.А., Гашков С.Б., Фролов А.Б., Часовских А.Б. О методах имплементации арифметических операций в криптографических системах. Известия РАН. Теория и системы управления (2002) N1, 86-96.

[13] Болотов А.А., Гашков С.Б., Фролов А.Б., Часовских А.Б. Программные и схемные методы умножения многочленов для эллиптической криптографии. Известия РАН. Теория и системы управления (2000) N5, 66-75.

[14] Болотов А.А., Гашков С.Б. О быстром умножении в нормальных базисах конечных полей Дискретная математика т.13 3(2001), 3-31.

[15] Брейсуэлл Р., Преобразование Хартли.//М.:«Мир» 1990.

[16] Брассар Ж. Современная криптология. М.: Полимед, 1999.

[17] Василенко О.Н. Теоретико-числовые алгоритмы в криптографии, МЦНМО, 2003.

[18] Вельшенбах М. Криптография на C и C++ в действии. М.: «Триумф», 2003.

[19] Виноградов И.М. Теория чисел. М.: Наука, 1990.

[20] Власенко В.А., Лаппа Ю.М., Ярославский Л.П. Методы синтеза быстрых алгоритмов свертки и спектрального анализа сигналов.//М.:«Наука» 1990.

[21] Влэдуц С.Г., Ногин Д.Ю., Цфасман М.А. Алгеброгеометрические коды. Основные понятия. МЦНМО 2003.

[22] Гашков С.Б. О сложности интегрирования рациональных дробей.//Труды Матем. Института им. Стеклова, т.218, с.122-133. М.«Наука»,1997

[23] Гашков С.Б. Замечания о быстром умножении многочленов, преобразовании Фурье и Хартли. Дискретная математика, т.12 N 3 (2000), 124-153.

[24] Гашков С.Б. Кочергин В.В. Об аддитивных цепочках векторов, вентильных схемах и сложности вычисления степеней. Дискретный анализ, 52(1992), 22-40.

[25] Гашков С.Б. Чубариков В.Н. Арифметика. Алгоритмы. Сложность вычислений. Высшая школа, Москва, 2000.

[26] Гельфонд А.О., Линник Ю.В. Элементарные методы в аналитической теории чисел. М.: ФМлит, 1962.

[27] Глухов М.М., Елизаров В.П., Нечаев А.А. Алгебра, т. 1,2. М.: Гелиос АРВ, 2003.

[28] Грушо А.А., Тимонина Е.Е. Теоретические основы защиты информации. М.: Изд. агентства «Яхтсмен», 1996.

[29] Грушо А.А., Тимонина Е.Е., Применко А.А. Анализ и синтез криптоалгоритмов. Йошкар-Ола, МФ МОСУ, 2000.

[30] Грушо А.А., Тимонина Е.Е., Применко А.А. Криптографические протоклолы. Йошкар-Ола, МФ МОСУ, 2001.

[31] Жельников В., Криптография от папируса до компьютера, Москва, ABF, 1996.

[32] Защита программного обеспечения /под. ред. Д.Гроувера. М.: Мир, 1992.

[33] Зубов А.Ю. Совершенные шифры. М.: Гелиос АРВ, 2003.

[34] Иванов М.А. Криптографические методы защиты информации в компьютерных системах и сетях. М.: «Кудиц-образ», 2002.

[35] Зензин О.С., Иванов М.А. Стандарт криптографической защиты AES. Конечные поля. М.: «Кудиц-образ», 2002.

[36] Иванов М.А., Чугунков И.В. Теория, применение и оценка качества генераторов псевдослучайных последовательностей . М.: «Кудиц-образ», 2003.

[37] Иванов М.А. и др. Поточные шифры. М.: «Кудиц-образ», 2003.

[38] Карацуба А.А.,Офман Ю.П., Умножение многозначных чисел на автоматах.// ДАН СССР. 1962. 145 N 2.с.293-294.

[39] Кнут Д. Искусство программирования на ЭВМ, т.2., Вильямс, 2000.

[40] Кнэпп.Э. Эллиптические кривые. М.: Факториал Пресс, 2004.

[41] Коблиц Н. Курс теории чисел и криптографии. М.: «ТВП», 2001.

[42] Коблиц Н. Введение в эллиптические кривые и модулярные формы. М.: «Мир>, 1988.

[43] Коновальцев И.В. Об одном алгоритме решения линейных уравнений в конечных полях. Наука, Москва, Проблемы кибернетики (1967), вып.19, 269-274.

[44] Эллиптические функции. М.: Наука, 1984.

[45] Лидл Р., Нидеррейтер Х. Конечные поля. Мир, Москва, 1988.

[46] Лупанов О.Б. О вентильных и контактно-вентильных схемах. ДАН СССР, 111 N 6(1956), 1171-1174.

[47] Лупанов О.Б. Асимптотические оценки сложности управляющих систем. Изд. МГУ, 1984.

[48] Макклеллан Дж.,Рейдер Ч., Применение теории чисел в цифровой обработке сигналов.//М.:«Радио и связь» 1983.

[49] Молдовян Н.А. Проблематика и методы криптологии. СПб.: изд. СПбГУ, 1998.

[50] Молдовян Н.А., Молдовян А.А., Советов Б.Я. Криптография. СПб.: «Лань», 2000.

[51] Московский университет и развитие криптографии в России. М.: МЦНМО, 2003.

[52] Нечаев В.И. Элементы криптографии. М.: Высшая школа, 1999.

[53] Ноден П., Китте К. Алгебраическая алгоритмика. М.: Мир, 1999.

[54] Острик В.В., Цфасман М.А. Алгебраическая геометрия и теория чисел. Рациональные и эллиптические кривые. МЦНМО 20001.

[55] Прахар К. Распределение простых чисел. М.: Мир, 1967.

[56] Прасолов В.В., Соловьев Ю.П. Эллиптические функции и алгебраические уравнения. М.: «Факториал», 1997.

[57] Саломаа А. Криптография с открытым ключом. М.: Мир, 1996.

[58] Слоэн Н.Дж.А. Коды, исправляющие ошибки, и криптография. - «Математический цветник», М.: Мир, 1983.

[59] Соколов А.В., Степанюк О.М. Методы информационной защиты обьектов и компьютерных сетей. М.: «АСТ», СПб.: «Полигон», 2000.

[60] Соловьев Ю.П., Садовничий В.А., Шавгулидзе Е.Т., Белокуров В.В. Эллиптические кривые и современные алгоритмы теории чисел. Москва-Ижевск 2003.

[61] Степанов С.А. Арифметика алгебраических кривых. М.: Наука, 1991.

[62] Уокер Р. Алгебраические кривые. М.: ИЛ, 1952.

[63] Фомичев В.М. Дискретная математика и криптология. Диалог-МИФИ, 2003.

[64] Хассе Г. Лекции по теории чисел. М.: ИЛ, 1953.

[65] Хофман Л. Современные методы защиты информации. М.: Сов. Радио, 1980.

[66] Харин Ю.С., Берник В.И., Матвеев Г.В., Агиевич С.В. Математические и компьютерные основы криптологии. Минск, «Новое знание», 2003

[67] Чмора А.Л. Современная прикладная криптография. М.: М.: Гелиос АРВ, 2001.

[68] Черемушкин А.В. Лекции по арифметическим алгоритмам в криптографии. М.: МЦНМО, 2002.

[69] Шафаревич И.Р. Алгебраическая геометрия. М.: Наука 1988.

[70] Саломаа А. Криптография с открытым ключом.М.: Мир: 1996.

[71] Шнейер Б. Прикладная криптография. М.: «Триумф», 2002.

[72] Шнейер Б. Секреты и ложь. Безопасность данных в цифровом мире. «Питер», 2003.

[73] Штрассен Ф. Алгоритм Гаусса не оптимален.// Кибернетический сборник вып.7 М.: «Мир», 1971.

[74] Шенхаге А.,Штрассен Ф. Быстрое умножение больших чисел.// Кибернетический сборник вып.10, М.: «Мир» 1973.

[75] Щербаков А., Домашев А. Прикладная криптография. М.: «Русская редакция », 2003.

[76] Введение в криптографию / под ред. В.В. Ященко. МЦНМО-ЧеРо, 1998.

[77] Adleman L., Lenstra H. Finding irreducible polinomials over finite fields. In Proc. 18th ACM Symp. Theory of computing (1986) 350-355.

[78] Agnew G.B., Beth T., Mullin R.C., Vanstone S.A. Arithmetic Operations in $GF(2^m)$. Journal of Cryptology. 1993. 6, 3-13.

[79] Agnew G.B., Mullin R.C., Onyszchuk I.M., Vanstone S.A. An implementation for a fast public-key cryptosystem. Journal of Cryptology. 1991. 3, N 2, 63-79.

[80] Ash D.W., Blake I.F., Vanstone S.A. Low complexity normal bases. Discrete Applied Mathematics (1989), vol.25, 191-210.

[81] Artin E. Collected papers, Addison-Wesley, 1965.

[82] Bach E. Explicit bounds for primality testing and related problems. Math.Comp. 55(1989), 355-380.

[83] Bach E., Shallit J. Factoring with cyclotomic polynomials. Math.Comp.(1989) vol. 52, 201-219.

[84] Bailey D.V., Paar C. Optimal extension fields for fast arithmetic in public-key algorithms. CRYPTO 98, Lecture Notes in Computer Science No 1462(1998), 472-485.

[85] Bailey D.V., Paar C. Efficient arithmetic in finite field extensions with application in elliptic curve cryptography. To appear in J. of Cryptology.

[86] Brent R.P., Kung H. Fast algorithms for manipulating formal power series. J. Assoc. Comput. Mach. (1978), vol. 25, 581-595.

[87] Brent R.P., Larvala S., Zimmerman P., A fast algoritm for testing irreducibity of trinomials mod 2, Preliminary report, Oxford University Computing Laboratory.

[88] Brickell E., Gordon D., McCurley K., Wilson D. Fast exponentiation with precomputation. In Proc. Eurocript'1992, Balatonfüred, Hungary.

[89] Brillhart J.,Lehmer D.,Selfridge J.,Tuckerman B.,Wagstaff S.,Jr, Factorization of $b^n \pm 1, b = 2, 3, 5, 6, 7, 10, 11, 12$ up to high powers, 2nd ed, Contemp.Math.,vol.22,Amer.Math.Soc.,Providence,RI,1988.

[90] Cantor D. On arithmetic algorithm over finite fields. J. of combinatorial theory, Series A 50 (1989), 285-300.

[91] Cantor D., Kaltofen E. On fast multiplication of polynomials over arbitrary algebras. Acta Informatica 28(1991), 693-701.

[92] Coppersmith D. Fast evaluation of logarithms in fields of characteristic two, IEEE Trans.Inform.Theory 30(1984) 587-594.

[93] Diffie W. and Hellman M. new directions in cryptography// IEEE Transactions and Information Theory. 1976. IT-22.

[94] Dewaghe I. Remarks on the Schoof-Elkies-Atkin algorithm. Mathematics of computation, v.67, N.223 (1998) 1247-1252.

[95] Fong K., Hankerson D., Lopez J.H., Menezes A. Field inversion and point halving revisited.

[96] Fouqet M., Gaudry P., Harley P. On Satoh's algoritm and its implementation (2000) preprint.

[97] Gao S., Lenstra H.W. Optimal normal bases. Design, Codes and Cryptography (1992), vol.2, 315–323.

[98] Gao S., Vanstone S.A. On orders of optimal normal basis generators. Mathematics of Computation (1995), vol.64, 1227-1233.

[99] Gao S., von zur Gathen J., Panario D. Gauss periods: orders and cryptografical applications. Mathematics of Computation (1998), vol.67, 343-352.

[100] Gao S., von zur Gathen J., Panario D., Shoup V. Algorithm for exponentiation in finite field J. of Symbolic Computation (2000), vol.29, 879-889.

[101] von zur Gathen J., Gerhard J. Modern computer algebra. Cambridge University Press, 1999.

[102] von zur Gathen J., Gerhard J. Arithmetic and factorization of polynomials over $GF(2)$. in Proc. ISAAC 96, Zürich, 1-9.

[103] von zur Gathen J., Giesbrecht M. Constructing normal bases in finite fields. J. Symbolic Computation (1990), 10, 547-579.

[104] von zur Gathen J. Pappalardi F. On the reverse Artin problem for primitive roots. Preprint (1995).

[105] Gao S., Shoup V. Computing Frobenius maps and factoring polynomials. Comput. complexity 2(1992), 187-224.

[106] Gauss C. Disquisitiones Arithmeticae. Braunschweig, 1801.

[107] Hankerson D., Lopez J.H., Menezes A. Software implementation of elliptic curve cryptography over binary fields. CHES 2000, Lecture Notes in Computer Science No 1965(2000), 1-23.

[108] Hooley C. On Artin's conjecture. J. reine angew. Math. 226 (1967), 209-220.

[109] Jungnickel D. Finite fields: Structure and arifmetics. Wissenschfttsverlag, 1995.

[110] IEEE 1363, Standard specification for public-key cryptography,2000. http://grouper.ieee.org/groups/1363/index.html

[111] ISO/IEC 14888-3 Information technology–Security techniques– Digital signatures with appendix (1998).

[112] ISO/IEC 15946 Information technology–Security techniques– Cryptograhic techniques, based on elliptic curves (1999).

[113] Kaliski B.,Jr. One-way permutations on elliptic curves. J. Cryptology (1991) N 3, 187-199.

[114] Knudsen E. Elliptic scalar multiplication using point having. ASIACRYPT'99 Lecture Notes in Computer Science 1716, 135-149..

[115] Koblitz N. Algebraic aspects of Cryptography. Springer Verlag.1998.

[116] Koblitz N. Constructing elliptic curves cryptosystems in characteristic 2. Crypto'90, Lecture Notes in Computer Science 537, 156-167.

[117] Koblitz N. CM-curves with good cryptographic properties. Crypto'91, Lecture Notes in Computer Science 576, 279-287.

[118] Kunihiro N., Koyama K. Equivalence of counting the number of points on elliptic vurve over the ring Z_n and factoring n. Eurocript-98, Lecture Notes in Computer Science 1403, 47-58.

[119] Lenstra H.W.Jr. Factoring integers with elliptic curves. Ann. of Math. v. 126 (1987) 649-673.

[120] Lercier R. Computing isogenies in $GF(2^n)$. Algorithmic number theory, Lecture Notes in Computer Science 1122(1996), 197-212.

[121] Lercier R. Finding good random ellilptic curves for cryptosystems defined $GF(2^n)$. Eurocript-97, Lecture Notes in Computer Science 1233(1997), 379-392.

[122] Lercier R., Morain F. Counting the number of points on ellilptic curves over finite fields: strategies and performances. Eurocript-95, Lecture Notes in Computer Science 921(1995), 79-94.

[123] Lay G., Zimmer H. Constructing elliptic curves with given group order over large finite fields. Algorithmic number theory, Lecture Notes in Computer Science 877(1994), 250-263.

[124] Lim G.H., Lee P.G. More flexible exponentiation with precomputation. CRYPTO-94, Springer-Verlag, pp.95-107.

[125] Lopez J., Dahab R. Improved algorith for elliptic curve over $GF(2^n)$ SAC'98, Lecture Notes in Computer Science No 1556(1999), 201-212.

[126] Lopez J., Dahab R. Fast meltiplication on elliptic curve over $GF(2^n)$ without precomputation. CHES 2000, Lecture Notes in Computer Science No 1965(2000), 317-327.

[127] Massey J.L., Omura J.K., Apparatus for finite fields computation, US patent 4587627 (1986).

[128] Mc Elice R.J. Finite Fields for Computer Scientists and Engineers. New York: Kluwer Academic Publishers, 1987.

[129] Menezes A.J. Elliptic Curve Public Key Cryptosystems. Kluwer Academic Publishers, 1993.

[130] Menezes A.J., Vanstone S. Elliptic Curve Cryptosystems and their implementation. Journal of Cryptology. 1993. 6, 209-224.

[131] Menezes A.J., Vanstone S., Okamoto T. Reducing elliptic curve logarithms to logarithms in a finite field. IEEE Trans. Inform. Th. v.39, N.5 (1993), 1639-1647.

[132] Menezes A.J., Vanstone S., Zuccherato R. Counting points on elliptic curves over $GF(2^n)$. Mathematics of computation, v.60, N.201 (1993) 407-420.

[133] Menezes A., van Oorshot P., Vanstone S. Handbook of applied cryptography.

[134] Meneses A., Blake I., Gao X., Mullin R., Vanstone S. and Yaghoobian T. Application of Finite Filds. New York: Kluwer Academic Publishers, 1992.

[135] Montgomery P. Speeding the Pollard and elliptic curve methods of factorization. Mathematics of computation, v.48 (1987) 243-264.

[136] Morain F. Calcul du nombre de points sur une courbe dans un corps fini: aspects algoritmiques. J.Theor.Nombres Bordeaux 7(1995), 255-282.

[137] Morain F. Building cyclic elliptic curves modulo large prime. Eurocript-91, Lecture Notes in Computer Science 547(1991), 328-336.

[138] Mullin R.C., Onyszchuk I.M., Vanstone S.A., Wilson R.M. Optimal normal bases in $GF(p^n)$. Discrete Applied Mathematics (1988/89), vol.22, 149–161.

[139] Moenk R., Fast algorithm of GCD's.// Proceedings of the 5th Annual ACM Symposium on Theory of Computing. 1973. p.142-151.

[140] Nussbaumer H. Fast polynomial transform algorithms for for digital convolution. IEEE Signal Processing 28(1980), 205-215.

[141] Paar C., Fan J.L., Efficient inversion in tower fields of characteristic two, ISIT, Ulm, Germany, 1997.

[142] Pippenger N. On the evaluation of powers and monomials. SIAM J. on Computing 9(1980) 230-250.

[143] Reyhani-Masoleh A., Hasan M.A. On effective normal basis multiplication, IndiaCRYPT, 2000.

[144] Саломаа А. КРиптография с открытым ключом. М.: Мир, 1976.

[145] Satoh T. The canonical lift of an ordinary elliptic curve over prime field and its point counting (1999) preprint.

[146] Schonhage A., Schnelle berechnung von kettenbruchentwicklungen//Acta Informatica 1 (1971), 139-144.

[147] Schonhage A. Schnelle Multiplication von Polynomen ueber Koerpern der Charakteristik 2. Acta Informatica (1977), vol.7, 395-398.

[148] Schoof R. Elliptic curves over finite fields and the computation of square roots *modp*. Math. Comp. 44(1985), 483-494.

[149] Schoof R. Counting points on elliptic curves over finite fields. J.Theor.Nombres Bordeaux 7(1995), 219-254.

[150] Schroeppel R., Orman H., Malley S'O., Spatschek O. Fast key exchange with elliptic curve systems. CRYPTO 95, Lecture notes in Computer Science No 963, 43-56.

[151] Schroeppel R. Automatically solving equations in finite fields. US Patent application No 09/834,363, publication number US2002/0055962 A1.

[152] Schroeppel R. Elliptic curve point having wins big. 2nd Midwest Arithmetical Geometry in Cryptograhy Workshop, Urbana, Illinois, November 2000.

[153] Standards for efficient cryptography group SEC-1: Elliptic curve cryptography, http://www.secg.org

[154] Standards for efficient cryptography group SEC-2: Recommended elliptic curve domain parameters, http://www.secg.org

[155] Silverman J. Fast multiplication in Finite Fields $GF(2^n)$. CHES 2000, Lecture Notes in Computer Science No 1965(2000), 123-133.

[156] Simon Singh. The code book. Doubleday, 1999.

[157] Solinas J. Efficient arithmetic on Koblitz curves. Designs, Codes and Cryptography 19(2000) 195-249.

[158] Stinson D.R. Cryptography: Theory and practice. CRC Press, 1995.

[159] Turk J.W.M. Fast arithmetic operations on numbers and polynomials, in Computational methods in number theory I, Mathematish Centrum, Amsterdam (1982), 43-54.

[160] Wagstaff S.,Jr, Update 2.6 to the second edition of factorization of $b^n \pm 1$,1993.

[161] Wassermann A. Zur Arithmetik in endlihen Körpern. Bayreuter Mathematishe Schriften (1993) vol. 44, 147-251.

[162] Wen Bao Han, The coefficients of primitive polynomials over finite fields, Math. Comp., 65(1996) 213, 331-340.

[163] Wu H. Montgomery multiplier and Squarer in $GF(2^n)$. CHES 2000, Lecture Notes in Computer Science No 1965(2000), 264-275.

[164] Wu H., Hasan M.A., Blake I.A. Higly regular Architectures for finite field computation using redundant basis. CHES 2000, Lecture Notes in Computer Science No 1965(2000), 269-279.

[165] Zirler N.,Brillhart J., On primitive trinomials (mod 2), Inform.Contr.,13(1968), 541-554.

[166] Zirler N., On $x^n + x + 1$ over $GF(2)$, Inform.Contr.,16(1970), 502-505.

[167] Zirler N.,Brillhart J., On primitive trinomials, Inform.Contr.,14(1969), 566-569.

[168] Zivkovic M., A table of primitive binary polynomials, Math. Comp. 62 (1994),385-386.

[169] Zivkovic M., A table of primitive binary polynomials.II, Math. Comp. 63 (1994),301-306.

[170] Yacobi Y. Exponentiation faster with addition chains. In Eurocript'90, Lecture notes in computer science (473), 222-229.

[171] Yamamichi M., Mambo M., Shizuya H. On the complexity of constructing an elliptci curve of a given order. IEICE Trans. fundamentals, v. E84-A,N 1, 2001.

Оглавление

Приложение А

Алгоритмы с матрицами

А.1 Приведение матрицы к треугольному виду

Пусть вектор $A = (P_1(x), P_2(x), \ldots, P_n(x))$, компонентами которого являются многочлены, задает матрицу с n строками и $m = \max\{\deg(P_i(x)) | i = 1, 2, \ldots, n\}$ столбцами, в i-ой строчке и j-ом столбце которой находится коэффициент при x^{j-1} многочлена $P_i(x)$.

В рассматриваемом представлении матриц удаление ее i-ой строчки означает удаление i-ой компоненты $P_1(x)$ вектора, а прибавление i-ой строки к j-ой означает прибавление к j-ой компоненте $P_j(x)$ этого вектора его i-ой компоненты $P_i(x)$.

Метод Гаусса приведения рассматриваемой матрицы A к треугольному виду выглядит следующим образом. Введем вспомогательную матрицу B, которая сначала не содержит элементов (вектор с 0 компонентами). В векторе A выберем компоненту $P_i(x)$ с наибольшей степенью. Пусть эта степень равна t. Если $t < 0$ (все многочлены вектора A равны 0), то вектор B является результатом приведения матрицы A к треугольному виду. В противном случае к каждой компоненте вектора A кроме i-ой компоненты, имеющей степень t, прибавим вектор $P_i(x)$. Из полученного вектора удалим i-ю компоненту и добавим в вектор B новую компоненту, равную $P_i(x)$. Получим новые матрицы A и B. С этой парой векторов проведем те же действия, что и с исходной парой и так будем действовать до тех пор, пока в векторе A все многочлены не станут нулевыми. После этого матрица B будет доставлять треугольный вид исходной матрицы A.

@Пример. Приведем к треугольному виду матрицу $A = (x + x^4, 1 + x^3, x^2 + x^4, x + x^3)$. Одна из компонент вектора A, имеющих максимальную степень, - первая. Выбрав первую компоненту матрицы A, после первой итерации алгоритма Гаусса, имеем: $A = (1 + x^3, x + x^2, x + x^3)$, $B = (x + x^4)$. Далее выбираем первую компоненту полученного вектора A и после второй итерации алгоритма Гаусса имеем: $A = (x + x^2, 1 + x)$ $B = (x + x^4, 1 + x^3)$. После третьей итерации - $A = (1 + x)$, $B = (x + x^4, 1 + x^3, x + x^2)$, а после четвертой итерации вектор A не содержит компонент, а вектор B имеет четыре компоненты $B = (x + x^4, 1 + x^3, x + x^2, 1 + x)$. Таким образом, ранг исходной матрицы A равен 4.

Перед выполнением очередной итерации алгоритма Гаусса рекомендуется выбирать компоненту вектора A с наибольшей степенью, которая задается наименьшим числом ненулевых элементарных многочленов. Это сократит время работы алгоритма, если принять во внимание следующие соображения. Пусть кроме элементарных многочленов, задающих рассматриваемый многочлен, нам известен массив индексов ненулевых элементарных многочленов. Тогда прибавление рассматриваемого многочлена к другому можно выполнить изменяя лишь элементарные многочлены с указанными индексами второго многочлена. Кроме того, реализовав операцию перестановки столбцов матрицы, можно сначала выбирать в векторе A ненулевую компоненту, задаваемую наименьшим числом элементарных многочленов, а затем,

переставив нужным образом столбцы матрицы A, добиться, чтобы степень этой компоненты была наибольшей.

А.2 Представление матрицы.

Двоичную матрицу M размеров $k \times n$ будем представлять таблицей T,

$$T = \begin{matrix} t_{11} & t_{12} & \dots & t_{1n} \\ t_{21} & t_{22} & \dots & t_{2n} \\ \cdot & \cdot & \cdot & \cdot \\ t_{k1} & t_{k2} & \dots & t_{kn} \end{matrix}$$

с k строками и $\lceil \frac{n}{s} \rceil$ столбцами, элементы которой t_{ij} - @*так называемые элементарные многочлены степени меньше* s @**целые неотрицательные числа, меньшие** 2^s. [1]Мы полагаем, что эти многочлены соответствуют "отрезкам"длины s, определяемой длиной машинного слова (целого типа данных). Мы полагаем также, что коэффициенты многочленов (они же разряды целого числа) расположены в слева направо в порядке возрастания степеней переменной (числового эквивалента двоичного разряда в данной позиции). "Отрезки"образуются отсчетом по s разрядов справа налево. Если последний "отрезок"t_{i1} оказывается не полным, то он пополняется младшими нулями до длины s.

@Пример. Пусть $m = k = 5$ и $s = 4$. Рассмотрим бинарную матрицу

$$\begin{matrix} 1 & 1 & 0 & 1 & 1 \\ 1 & 0 & 1 & 0 & 1 \\ 0 & 0 & 1 & 1 & 0 \\ 1 & 1 & 1 & 0 & 0 \\ 0 & 0 & 1 & 1 & 1 \end{matrix}$$

Имеем $n = \lceil \frac{m}{s} \rceil = 2$, поэтому таблица T будет иметь 2 столбца:

$$T = \begin{matrix} 1 & 11 \\ 1 & 5 \\ 0 & 6 \\ 1 & 12 \\ 0 & 7 \end{matrix}$$

Заметим, что при переходе к таблицам, элементами которой являются числа типа unsigned long, количество столбцов исходной матрицы сокращается в 32 раза.

А.3 Умножение матрицы на вектор

Рассмотрим $k \times n$ таблицу T и вектор-столбец V с n компонентами, элементами которых являются целые неотрицательные числа, не превосходящие $2^s - 1$. Таблица T построена из некоторой бинарной $k \times m$ матрицы описанным выше способом, а вектор-столбец V - результат преобразования некоторого бинарного вектора размерности m рассмотренным выше способом с последующим транспонированием.

$$T = \begin{matrix} t_{11} & t_{12} & \dots & t_{1n} \\ t_{21} & t_{22} & \dots & t_{2n} \\ \cdot & \cdot & \cdot & \cdot \\ t_{k1} & t_{k2} & \dots & t_{kn}, \end{matrix}$$

$$V = \begin{matrix} v_1 \\ v_2 \\ \cdots \\ v_n \end{matrix}$$

Рассмотрим новый вектор V', положив

$$V' = \begin{matrix} \sum_{i=1}^{n} t_{1i}v_i \\ \sum_{i=1}^{n} t_{2i}v_i \\ \cdots \\ \sum_{i=1}^{n} t_{ki}v_i, \end{matrix}$$

где сложение и умножение являются логическими (поразрядными). Например, $10 \cdot 12 + 9 \cdot 11 = 1$. Далее, из вектора-столбца V' получим бинарный вектор-столбец V'' высоты k,

$$V'' = \begin{matrix} v'_1 \\ v'_2 \\ \cdots \\ v'_k, \end{matrix}$$

где v'_j - результат суммирования по модулю 2 двоичных разрядов числа $\sum_{i=1}^{n} t_{ji}v_i$.

Транспонируем вектор V'', описанным в первом параграфе способом сопоставим полученному вектору новый вектор с $]k/s[$ компонентами, элементами которого являются целые неотрицательные числа не большие $2^s - 1$. Полученный вектор снова транспонируем. Новый вектор-столбец является результатом умножения T на V.

@Пример. Пусть, как и в предыдущем примере, $m = k = 5$ и $s = 4$,

$$T = \begin{matrix} 1 & 11 \\ 1 & 5 \\ 0 & 6 \\ 1 & 12 \\ 0 & 7 \end{matrix}$$

Требуется перемножить бинарную матрицу, которой соответствует приведенная выше таблица T, на вектор $(1, 0, 1, 0, 1)$. Таким образом, описанным выше способом нужно умножить T на вектор-столбец $(1, 5)^T$.(Через M^T обозначается таблица, получаемая из M транспонированием.) Имеем:

$$V' = \begin{matrix} 1 \cdot 1 + 11 \cdot 5 \\ 1 \cdot 1 + 5 \cdot 5 \\ 0 \cdot 1 + 6 \cdot 5 \\ 1 \cdot 1 + 12 \cdot 5 \\ 0 \cdot 1 + 7 \cdot 5 \end{matrix}$$

или $V' = (0, 4, 4, 5, 5)^T$, откуда $V'' = (0, 1, 1, 0, 0)^T$. Таким образом, $TV = (0, 12)^T$.

При умножении $k \times n$-таблицы на n-вектор используется kn логических умножений, $k(n-1)$ логических сложений, sk выделений разрядов и бинарных сложений, а также $O(k)$ операций при переходе от вектора V'' к результату умножения. При умножении же бинарной таблицы на бинарный вектор пришлось бы выполнить $O(snk)$ операций.

A.4 Алгоритм GAUS-MATRIX-TRIAN

Опишем алгоритм Гауса (GAUS-MATRIX-TRIAN) при работе с такими таблицами. Пусть задана таблица T (см. выше), а также пустая таблица T_1, Все строки таблицы T считаем

активными. После окончания работы алгоритма матрица из нулей и единиц, соответствующая таблице T_1 доставляет треугольный вид соответствующей матрицы для таблицы T.

Пусть активны строки с номерами $i_0, i_0{+}1, \ldots, k$. Среди чисел в активных строках первого столбца а выберем наибольшее. Пусть, например, это будет число $t_{j1}, \quad j \geq i_0$. Если $t_{j1} = 0$, то удаляем из T первый столбец и добавляем этот столбец в таблицу T_1 справа. Далее снова рассматриваем элементы первого столбца уже полученной матрицы.

В противном случае определяем такое целое неотрицательное число r, что $2^r \leq t_{j1} < 2^{r+1}$. Если $t_{i_0 1} < 2^r$, то к строке с номером i_0 таблицы T прибавляем (в логическом смысле) строку с номером j. (Логическая сумма двух целых чисел это результат их поразрядного сложения по модулю 2. Например, $(1,1,0,0) + (1,0,1,0) = (0,1,1,0)$.) **Используемую операцию прибавления одной строки таблицы кдругой выделим в в отдельную процедуру, которую назовем SUM-MATRIX'LINE.**

Далее для каждого $i, \quad i = i_0 + 1, i_0 + 2, \ldots, k$ в случае $t_{i1} \geq 2^r$ i-ю строку матрицы T, заменяем логической суммой i-й и i_0-й строк. i_0-ую строку таблицы T далее считаем неактивной.

С полученной таблицей T далее проделываем описанные выше действия, начиная с поиска активной строки с наибольшим первым слева элементом и т.д. до тех пор, пока в таблице T не останется элементов или не останется активных строк. Если в таблице T не осталось активных строк, но она не пуста, то приписываем ее справа к таблице T_1.

После этого работа алгоритма GAUS-MATRIX-TRIAN заканчивается.

Пример. Пусть $s = 4$ и

$$T = \begin{matrix} 7 & 15 & 3 \\ 8 & 10 & 4 \\ 11 & 2 & 6 \\ 4 & 13 & 1 \\ 10 & 3 & 2 \end{matrix}$$

Третья строка таблицы T имеет наибольший первый элемент (11), $2^3 \leq 11 < 2^4$ и $r = 3$. Так как $7 < 2^r$, то к первой строчке таблицы T прибавляем (в логическом смысле) третью. Получим новую таблицу

$$T = \begin{matrix} 12 & 13 & 5 \\ 8 & 10 & 4 \\ 11 & 2 & 6 \\ 4 & 13 & 1 \\ 10 & 3 & 2 \end{matrix}$$

Так как первые элементы второй, третьей и пятой строк полученной таблицы не меньше $2^3 = 8$, то к этим строкам прибавляем первую строку. Получим.

$$T = \begin{matrix} 12 & 13 & 5 \\ 4 & 7 & 1 \\ 7 & 15 & 3 \\ 4 & 13 & 1 \\ 6 & 14 & 7 \end{matrix}$$

Первую строчку полученной таблицы далее считаем неактивной, а таблица T_1 остается пустой.

В полученной матрице T наибольший первый справа элемент среди активных строк - в третьей строке. Имеют место неравенства $2^2 \leq 7 < 2^3$. Поэтому $r = 2$. Т.к. $4 \geq 2^2$, то вторую строку не меняем, а ко всем строкам, начиная с третьей прибавляем вторую строку. Получаем

новую таблицу T,

$$T = \begin{matrix} 12 & 13 & 5 \\ 4 & 7 & 1 \\ 3 & 8 & 2 \\ 0 & 10 & 0 \\ 2 & 9 & 6 \end{matrix}$$

Вторую строку таблицы T далее считаем неактивной. Таблица T_1 остается пока пустой.

В полученной таблице T среди активных строк, имеющих наибольший первый элемент - третья и $2^1 \leq 3 < 2^2, \quad r = 2$. Прибавляя эту строку к пятой, получим

$$T = \begin{matrix} 12 & 13 & 5 \\ 4 & 7 & 1 \\ 3 & 8 & 2 \\ 0 & 10 & 0 \\ 1 & 1 & 4 \end{matrix}$$

Третью строку таблицы T далее считаем неактивной. Таблица T_1 по-прежнему остается пустой.

На следующем шаге к четвертой строчке таблицы T прибавляется пятая, а после этого к пятой строке прибавляется новая четвертая. Получается

$$T = \begin{matrix} 12 & 13 & 5 \\ 4 & 7 & 1 \\ 3 & 8 & 2 \\ 1 & 11 & 4 \\ 0 & 10 & 0 \end{matrix}$$

Таблица T_1 снова остается пустой, а четвертая строка становится неактивной.

Единственная оставшаяся активной строка таблицы T имеет первый элемент равный нулю. Поэтому переносим первый столбец из T в T_1. Получаем

$$T = \begin{matrix} 13 & 5 \\ 7 & 1 \\ 8 & 2 \\ 11 & 4 \\ 10 & 0 \end{matrix}$$

$$T_1 = \begin{matrix} 12 \\ 4 \\ 3 \\ 1 \\ 0 \end{matrix}$$

Далее так как в T осталась единственная активная строка, правый элемент которой не равен нулю, то таблица T не изменяется, пятая строка становится неактивной, В таблице T не осталось активных строк. Поэтому приписываем эту таблицу к T_1 справа. Получаем

$$T_1 = \begin{matrix} 12 & 13 & 5 \\ 4 & 7 & 1 \\ 3 & 8 & 2 \\ 1 & 11 & 4 \\ 0 & 10 & 0 \end{matrix}$$

Алгоритм заканчивает работу. Таблица T_1 - результат приведения T к треугольному виду.

При приведении таблицы T с n строками и n столбцами к треугольному виду по приведенному выше алгоритму GAUS-MATRIX-TRIAN используется:

1. не более чем $k^2]k/s[$ операций логического сложения s-разрядных чисел;
2. не более чем k^2 операций сравнения при определении наибольших элементов в столбцах;
3. не более чем k^2 операций деления на 2 при определении чисел r;
4. не более чем k^2 сравнений правых элементов в строках с числами 2^r при определении необходимости складывать строки;
5. не более чем k операций удаления столбцов, а также добавления столбцов и придания неактивного статуса строкам;
6. не более чем $3k$ операций с компонентами вектора V.

Всего операций не более $k^2]k/s[+3k^2+6k$. Поэтому при $s=32, \quad k \approx 200$ алгоритм GAUS-MATRIX-TRIAN дает существенный выигрыш по времени.

A.5 Проверки невырожденности матрицы

Пусть T - таблица, описанная в параграфе "Представление матриц", а T_1 - результат обработки таблицы T алгоритмом GAUS-MATRIX-TRIAN. Элемент таблицы T_1, находящийся в i-ой строке и j-ом столбце обозначим t'_{ij}. Пусть также таблице T_1 соответствует квадратная матрица $M=(m_{ij})$ из нулей и единиц.

Опишем, как по таблице T проверить, является ли матрица M, невырожденной. Для этого будем работать с s-разрядными числами (элементами таблицы T_1). Для каждого $i, \quad i=1,2,\ldots,k$ рассмотрим строку таблицы T_1 с номером i. Пусть @j $=]frack-i+1s[$ - номер столбца (считая справа), в котором должен находиться бит равный m_{ii}. Через r_i обозначим число $k-i+1-(j-1)s$. Если число $l_i=[t_{ij}/2^{r_i-1}]$ делится на 2, то $m_{ii}=0$ и таблица T вырождена. Если для каждого $i, \quad i=1,2,\ldots,k$, число l_i не делтся на 2, то матрица M невырождена.

@Пример. Пусть $s=2, \quad k=5$ и результатом работы алгоритма GAUS-MATRIX-TRIAN является таблица T_1,

$$T_1=\begin{matrix}1&2&1\\0&2&3\\0&1&1\\0&0&3\\0&0&1,\end{matrix}$$

Матрица M' из нулей и единиц, которая соответствует таблице T_1, имеет вид

$$M'=\begin{matrix}1&1&0&0&1\\0&1&0&1&1\\0&0&1&0&1\\0&0&0&1&1\\0&0&0&0&1.\end{matrix}$$

Имеем $n=3$. Для первой строки таблицы T_1 выполнено: $j=3, \quad r_1=1, \quad l_1=1$. Число l_1 не делится на 2. Далее проверяем, что l_2,l_3,l_4,l_5 не делятся на 2. Таким образом, таблица T_1 соответствует невырожденной матрице из нулей и единиц.

Нетрудно видеть, что проверку невырожденности таблицы T, алгоритм NONSIGN-MATRIX выполняет за $O(k)$ действий.

A.6 Приведение матрицы к диагональному виду.

Пусть далее матрица M невырождена. Приведем ее к диагональному виду. Пусть индекс i пробегает последовательность $k, k-1, \ldots, 3, 2$. Для каждого i положим $j =]frack - i + 1s[, \quad r_i = k - i + 1 - (j-1)s$. Пусть i' пробегает значения $i+1, i+2, \ldots, k$. Положим $l_{i'j} = [t_{i'j}/2^{r_i - 1}]$. Если $l_{i'j}$ не делится на 2, то элемент $m_{i'i}$ не равен нулю и нужно к строке таблицы T, имеющей номер i' прибавить (в логическом смысле) строку с номером i.

Пусть выполнены все указанные действия. Тогда таблица T_1 соответствует бинарной единичной $k \times k$ матрице E,

$$E = \begin{matrix} 1 & 0 & \ldots & 0 \\ 0 & 1 & \ldots & 0 \\ . & . & . & . \\ 0 & 0 & \ldots & 1 \end{matrix}$$

являющейся результатом приведения исходной бинарной матрицы M к треугольному виду.

@Пример. Пусть, как и в предедущем примере,$s = 2, \quad k = 5$ и

$$T_1 = \begin{matrix} 1 & 2 & 1 \\ 0 & 2 & 3 \\ 0 & 1 & 1 \\ 0 & 0 & 3 \\ 0 & 0 & 1, \end{matrix}$$

Т.к. последний элемент всех строчек (на пятую строку можно не смотреть) не делится на 2, то к каждой строке (кроме пятой) прибавляем пятую строку. Получаем

$$T_1 = \begin{matrix} 1 & 2 & 0 \\ 0 & 2 & 2 \\ 0 & 1 & 0 \\ 0 & 0 & 2 \\ 0 & 0 & 1, \end{matrix}$$

Далее, последний элемент в первой и третьей строках таблицы T_1 делится на 4, поэтому четвертую строку прибавляем (в логическом смысле) только ко второй. Получаем таблицу T_1,

$$T_1 = \begin{matrix} 1 & 2 & 0 \\ 0 & 2 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 2 \\ 0 & 0 & 1, \end{matrix}$$

Далее, предпоследний элемент первой и второй строк делится на 2, поэтому, не меняя таблицу T_1, переходим к рассмотрению ее второй строки. Предпоследний элемент первой строки не делится на 4. Поэтому к первой строке прибавляем вторую. Получаем

$$T_1 = \begin{matrix} 1 & 0 & 0 \\ 0 & 2 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 2 \\ 0 & 0 & 1, \end{matrix}$$

Полученная таблица T_1 соответствует диагональному виду матрицы M'.

Нетрудно видеть, что приведение к диагональному виду алгоритм GAUS-MATRIX-DIAGON выполняет за $O(n^2)$ действий, т.к. прибавляемая строка содержит ровно один ненулевой элемент.

А.7 Обращение матрицы

Рассмотрим таблицу T,

$$T = \begin{matrix} t_{11} & t_{12} & \dots & t_{1n} \\ t_{21} & t_{22} & \dots & t_{2n} \\ \cdot & \cdot & \cdot & \cdot \\ t_{k1} & t_{k2} & \dots & t_{kn}, \end{matrix}$$

соответствующую квадратной $k \times k$ матрице из нулей и единиц.

Обозначим через E единичную $k \times k$ матрицу,

$$E = \begin{matrix} 1 & 0 & 0 & \dots & 0 \\ 0 & 1 & 0 & \dots & 0 \\ 0 & 0 & 1 & \dots & 0 \\ \cdot & \cdot & \cdot & \cdot & \cdot \\ 0 & 0 & 0 & \dots & 1. \end{matrix}$$

Ей соответствует таблица

$$E' = \begin{matrix} t'_{11} & t'_{12} & \dots & t'_{1n} \\ t'_{21} & t'_{22} & \dots & t'_{2n} \\ \cdot & \cdot & \cdot & \cdot \\ t'_{k1} & t'_{k2} & \dots & t'_{kn}, \end{matrix}$$

с элементами из E_2^s.

Применим алгоритм GAUS-MATRIX-TRIAN к таблице T. При этом, если в процессе работы алгоритма некоторая строка таблицы T прибавляется к другой ее строке, то ту же самую операцию производим с таблицей E'.

Если матрица, соответствующая таблице T оказалась невырожденной, и таблица T_1 - результат применения к T алгоритма GAUS-MATRIX-TRIAN, то применим к T_1 алгоритм GAUS-MATRIX-DIAGON. При этом опять, если в процессе работы алгоритма некоторая строка таблицы T_1 прибавляется к другой ее строке, то ту же самую операцию производим с таблицей полученной из E'.

После приведения таблицы T_1 к диагональному виду результат преобразования таблицы E' соответствует матрице обратной к рассматриваемой $k \times k$ матрице.

Пример. Пусть $k = 5, s = 2$ и

$$T = \begin{matrix} 1 & 2 & 0 \\ 0 & 2 & 1 \\ 1 & 0 & 2 \\ 0 & 1 & 1 \\ 1 & 1 & 2, \end{matrix}$$

Имеем

$$E' = \begin{matrix} 1 & 0 & 0 \\ 0 & 2 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 2 \\ 0 & 0 & 1. \end{matrix}$$

После обработки первой строки таблицы T получим

$$T = \begin{matrix} 1 & 2 & 0 \\ 0 & 2 & 1 \\ 0 & 2 & 2 \\ 0 & 1 & 1 \\ 0 & 3 & 2, \end{matrix}$$

$$E' = \begin{matrix} 1 & 0 & 0 \\ 0 & 2 & 0 \\ 1 & 1 & 0 \\ 0 & 0 & 2 \\ 1 & 0 & 1. \end{matrix}$$

После обработки второй строки:

$$T = \begin{matrix} 1 & 2 & 0 \\ 0 & 2 & 1 \\ 0 & 0 & 3 \\ 0 & 1 & 1 \\ 0 & 1 & 3, \end{matrix}$$

$$E' = \begin{matrix} 1 & 0 & 0 \\ 0 & 2 & 0 \\ 1 & 3 & 0 \\ 0 & 0 & 2 \\ 1 & 2 & 1. \end{matrix}$$

После прибавления четвертой строки к третьей и обработки третьей строки:

$$T = \begin{matrix} 1 & 2 & 0 \\ 0 & 2 & 1 \\ 0 & 1 & 2 \\ 0 & 0 & 3 \\ 0 & 0 & 1, \end{matrix}$$

$$E' = \begin{matrix} 1 & 0 & 0 \\ 0 & 2 & 0 \\ 1 & 3 & 2 \\ 1 & 3 & 0 \\ 0 & 1 & 3. \end{matrix}$$

Таким образом, таблица T уже имеет треугольный вид и далее применяем к ней алгоритм GAUS-MATRIX-DIAGON. После обработки пятой строки таблицы T получаем:

$$T = \begin{matrix} 1 & 2 & 0 \\ 0 & 2 & 0 \\ 0 & 1 & 2 \\ 0 & 0 & 2 \\ 0 & 0 & 1, \end{matrix}$$

$$E' = \begin{matrix} 1 & 0 & 0 \\ 0 & 3 & 3 \\ 1 & 3 & 2 \\ 1 & 3 & 0 \\ 0 & 1 & 3. \end{matrix}$$

После обработки четвертой строки:

$$T = \begin{matrix} 1 & 2 & 0 \\ 0 & 2 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 2 \\ 0 & 0 & 1, \end{matrix}$$

$$E' = \begin{matrix} 1 & 0 & 0 \\ 0 & 3 & 3 \\ 0 & 1 & 1 \\ 1 & 2 & 3 \\ 0 & 1 & 3. \end{matrix}$$

После обработки третьей строки ничего не изменяется, а после обработки второй получаем

$$T = \begin{matrix} 1 & 0 & 0 \\ 0 & 2 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 2 \\ 0 & 0 & 1, \end{matrix}$$

$$E' = \begin{matrix} 1 & 3 & 3 \\ 0 & 3 & 3 \\ 0 & 1 & 1 \\ 1 & 2 & 3 \\ 0 & 1 & 3. \end{matrix}$$

Полученная таблица E' соответствует бинарной матрице

$$M^{-1} = \begin{matrix} 1 & 1 & 1 & 1 & 1 \\ 0 & 1 & 1 & 1 & 1 \\ 0 & 0 & 1 & 0 & 1 \\ 1 & 1 & 0 & 1 & 1 \\ 0 & 0 & 1 & 1 & 1, \end{matrix}$$

которая является обратной к исходной:

$$M = \begin{matrix} 1 & 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 1 \\ 1 & 0 & 0 & 1 & 0 \\ 0 & 0 & 1 & 0 & 1 \\ 1 & 0 & 1 & 1 & 0. \end{matrix}$$

A.8 Умножение вектор-строки на матрицу

Пусть задана таблица T с k строчками и n столбцами, на пересечении i-ой строки и j-го столбца которой находится целое неотрицательное число t_{ij}, не превосходящие $2^s - 1$, $(n-1)s < k \le ns$. Таким образом, таблица T задает квадратную бинарную матрицу M с k строками и k столбцами, на пересечении i-ой строки и j-го столбца которой находится число a_{ij}. Введем операцию "U"над таблицами. Результатами применения этой операции к таблице T являются таблица $U(T)$, имеющая n строк и k столбцов.

Содержательно мы преобразуем сначала таблицу T к матрице M и тем самым определяем элементы a_{ij}. Через q и r обозначим соответственно числа $]j/s[$ и $j-(q-1)s$. Тогда число a_{ij} равно остатку от деления $[t_{iq}/2^{r-1}]$ на 2.

Определим таблицу $U(T)$, на пересечении i-ой строки и j-го столбца которой находится целое неотрицательное число u_{ij}, для каждого j, $1 \le j \le k$ и для каждого i, $1 \le i \le n$, полагая

$$u_{ij} = \sum_{i'=(i-1)s}^{\min(is-1,k)} a_{i'j} 2^{i'-(i-1)s}.$$

Важное замечание! Не следует преобразовывать таблицу T в матрицу M целиком. Следует последовательно вычислять и сразу использовать необходимые для вычисления элемента

$u(i,j)$ элементы матрицы M, **не сохраняя их!** Ниже в примере матрица M формируется целиком. В порядке упражнения предлагается произвести вычисления без ее предварительного формирования.

@Пример. Пусть $k=3$ и $s=2$. Рассмотрим таблицу T,

$$T = \begin{matrix} 0 & 2 \\ 1 & 0 \\ 1 & 3. \end{matrix}$$

Сначала она преобразуется в матрицу M,

$$M = \begin{matrix} 0 & 1 & 0 \\ 1 & 0 & 0 \\ 1 & 1 & 1. \end{matrix}$$

Затем, просматривая столбцы матрицы M сверху вниз и формируя группы по s элементов (последняя группа при необходимости пополняется недостающими до длины s нулями), получаем таблицу $U(T)$:

$$U(T) = \begin{matrix} 2 & 1 & 0 \\ 1 & 1 & 1. \end{matrix}$$

Далее, пусть $v=(t_1,t_2,\ldots,t_n)$, где t_i - целое неотрицательное число не превосходящее $2^s-1, \quad i=1,2,\ldots,n$. Для каждого $j, \quad j=1,2,\ldots,k$ положим

$$v'_j = \sum_{i=1}^{n} t_i u_{ij},$$

где произведение целых неотрицательных чисел t_i и u_{ij} осуществляется покомпонентно, т.е. если $t_i = \sum_{i_1=0}^{s-1} a_{i_1} 2^{i_1}, \quad u_{ij} = \sum_{i_1=0}^{s-1} b_{i_1} 2^{i_1}$, то $t_i u_{ij} = \sum_{i_1=0}^{s-1} a_{i_1} b_{i_1} 2^{i_1}$; @суммируются эти произведения также покомпонентно по модулю 2. В полученном векторе $v'=(v'_1,v'_2,\ldots,v'_k)$ каждая компонента v'_j является целым неотрицательным числом не превосходящим 2^s-1. Положим $c_j=0$, если число единиц в двоичном разложении v'_j четно и $c_j=1$ в противном случае. Вектор $c=(c_1,c_2,\ldots,c_k)$ преобразуем в вектор $u=(u_1,u_2,\ldots,u_n)$, положив

$$u_j = \sum_{i_1=s(j-1)+1}^{\min(sj,k)} c_{i_1} 2^{i_1-s(j-1)-1}.$$

Результатом умножения вектор-строки v на таблицу T явдяется вектор u.

Продолжая рассматриваемый в текущем параграфе @пример, перемножим вектор $v=(2,1)$ на таблицу T, После вычисления таблицы $U(T)$ определяем вспомогательный вектор $v'=(2,1,1)$. Тогда $c=(0,1,1)$. Таким образом, результатом умножения вектора v на T в данном случае будет вектор $(2,1)$.

Приложение B

Таблица размерностей ОНБ

Ниже приведена составленная на основе компьютерных вычислений таблица размерностей полей $GF(2^n)$ при $1000 < n10000$, для которых существуют оптимальные нормальные базисы (ОНБ). В скобках после размерности указан тип базиса (если для рассматриваемого поля одновременно существуют базисы типа 1 и 2, то в скобках стоит 12; другие варианты не встречаются).

Таблицу для $n \leq 1000$ мы не приводим, так как она имеется, например, в [?],[133]. Также мы не приводим вычисляемые программой для каждого оптимального нормального базиса матрицы A, T и порождающие элементы этих базисов.

```
-----------------------------------------------------------
1013(2) 1014(2) 1018(1) 1019(3) 1026(2) 1031(3) 1034(2)
-----------------------------------------------------------
1041(2) 1043(3) 1049(2) 1055(3) 1060(1) 1065(2) 1070(2)
-----------------------------------------------------------
1090(1) 1103(3) 1106(2) 1108(1) 1110(2) 1116(1) 1118(2)
-----------------------------------------------------------
1119(3) 1121(2) 1122(1) 1133(2) 1134(2) 1146(2) 1154(2)
-----------------------------------------------------------
1155(3) 1166(2) 1169(2) 1170(1) 1178(2) 1185(2) 1186(1)
-----------------------------------------------------------
1194(2) 1199(3) 1211(3) 1212(1) 1218(2) 1223(3) 1228(1)
-----------------------------------------------------------
1229(2) 1233(2) 1236(1) 1238(2) 1251(3) 1258(1) 1265(2)
-----------------------------------------------------------
1269(2) 1271(3) 1274(2) 1275(3) 1276(1) 1278(2) 1282(1)
-----------------------------------------------------------
1289(2) 1290(1) 1295(3) 1300(1) 1306(1) 1310(2) 1323(3)
-----------------------------------------------------------
1329(2) 1331(3) 1338(2) 1341(2) 1346(2) 1349(2) 1353(2)
-----------------------------------------------------------
```

```
1355(3) 1359(3) 1370(2) 1372(1) 1380(1) 1394(2) 1398(2)
-------------------------------------------------------
1401(2) 1409(2) 1418(2) 1421(2) 1425(2) 1426(1) 1430(2)
-------------------------------------------------------
1439(3) 1443(3) 1450(1) 1451(3) 1452(1) 1454(2) 1463(3)
-------------------------------------------------------
1469(2) 1478(2) 1481(2) 1482(1) 1492(1) 1498(1) 1499(3)
-------------------------------------------------------
1505(2) 1509(2) 1511(3) 1518(2) 1522(1) 1530(1) 1533(2)
-------------------------------------------------------
1539(3) 1541(2) 1548(1) 1559(3) 1570(1) 1583(3) 1593(2)
-------------------------------------------------------
1601(2) 1618(1) 1620(1) 1626(2) 1636(1) 1649(2) 1653(2)
-------------------------------------------------------
1659(3) 1661(2) 1666(1) 1668(1) 1673(2) 1679(3) 1685(2)
-------------------------------------------------------
1692(1) 1703(3) 1706(2) 1730(2) 1732(1) 1733(2) 1734(2)
-------------------------------------------------------
1740(1) 1745(2) 1746(1) 1749(2) 1755(3) 1758(2) 1763(3)
-------------------------------------------------------
1766(2) 1769(2) 1773(2) 1778(2) 1779(3) 1785(2) 1786(1)
-------------------------------------------------------
1790(2) 1791(3) 1806(2) 1811(3) 1818(2) 1821(2) 1829(2)
-------------------------------------------------------
1835(3) 1838(2) 1845(2) 1850(2) 1854(2) 1859(3) 1860(1)
-------------------------------------------------------
1863(3) 1866(12) 1876(1) 1883(3) 1889(2) 1898(2) 1900(1)
-------------------------------------------------------
1901(2) 1906(1) 1923(3) 1925(2) 1926(2) 1930(1) 1931(3)
-------------------------------------------------------
1938(2) 1948(1) 1953(2) 1955(3) 1958(2) 1959(3) 1961(2)
-------------------------------------------------------
1965(2) 1972(1) 1973(2) 1978(1) 1983(3) 1986(1) 1994(2)
-------------------------------------------------------
1996(1) 2001(2) 2003(3) 2006(2) 2009(2) 2010(2) 2026(1)
-------------------------------------------------------
2028(1) 2039(3) 2045(2) 2046(2) 2049(2) 2052(1) 2055(3)
-------------------------------------------------------
2063(3) 2066(2) 2068(1) 2069(2) 2078(2) 2079(3) 2082(1)
-------------------------------------------------------
2098(1) 2109(2) 2114(2) 2115(3) 2121(2) 2126(2) 2129(2)
-------------------------------------------------------
2130(12) 2140(1) 2141(2) 2163(3) 2174(2) 2178(2) 2181(2)
```

```
---------------------------------------------------------
2186(2) 2195(3) 2198(2) 2212(1) 2220(1) 2223(3) 2225(2)
---------------------------------------------------------
2231(3) 2236(1) 2241(2) 2242(1) 2246(2) 2253(2) 2258(2)
---------------------------------------------------------
2266(1) 2268(1) 2273(2) 2291(3) 2292(1) 2295(3) 2301(2)
---------------------------------------------------------
2308(1) 2310(2) 2318(2) 2319(3) 2332(1) 2338(1) 2339(3)
---------------------------------------------------------
2345(2) 2351(3) 2356(1) 2361(2) 2370(1) 2388(1) 2391(3)
---------------------------------------------------------
2393(2) 2394(2) 2399(3) 2406(2) 2415(3) 2436(1) 2438(2)
---------------------------------------------------------
2451(3) 2458(1) 2459(3) 2466(12) 2471(3) 2475(3) 2476(1)
---------------------------------------------------------
2478(2) 2483(3) 2486(2) 2493(2) 2501(2) 2505(2) 2511(3)
---------------------------------------------------------
2519(3) 2525(2) 2529(2) 2530(1) 2538(12) 2543(3) 2548(1)
---------------------------------------------------------
2549(2) 2553(2) 2556(1) 2559(3) 2573(2) 2578(1) 2585(2)
---------------------------------------------------------
2589(2) 2594(2) 2613(2) 2615(3) 2620(1) 2630(2) 2651(3)
---------------------------------------------------------
2654(2) 2658(1) 2666(2) 2675(3) 2676(1) 2682(1) 2692(1)
---------------------------------------------------------
2693(2) 2698(1) 2699(3) 2703(3) 2706(1) 2715(3) 2721(2)
---------------------------------------------------------
2738(2) 2739(3) 2740(1) 2741(2) 2750(2) 2753(2) 2759(3)
---------------------------------------------------------
2763(3) 2778(2) 2781(2) 2786(2) 2788(1) 2795(3) 2796(1)
---------------------------------------------------------
2802(1) 2811(3) 2818(1) 2819(3) 2823(3) 2825(2) 2829(2)
---------------------------------------------------------
2836(1) 2841(2) 2842(1) 2846(2) 2850(12) 2858(2) 2860(1)
---------------------------------------------------------
2870(2) 2871(3) 2874(2) 2889(2) 2891(3) 2895(3) 2903(3)
---------------------------------------------------------
2906(2) 2908(1) 2913(2) 2919(3) 2921(2) 2925(2) 2934(2)
---------------------------------------------------------
2938(1) 2939(3) 2951(3) 2956(1) 2961(2) 2962(1) 2963(3)
---------------------------------------------------------
2969(2) 2993(2) 3005(2) 3010(1) 3014(2) 3018(1) 3023(3)
---------------------------------------------------------
```

```
3026(2) 3033(2) 3036(1) 3050(2) 3065(2) 3066(1) 3071(3)
-------------------------------------------------------
3082(1) 3086(2) 3098(2) 3099(3) 3101(2) 3105(2) 3114(2)
-------------------------------------------------------
3123(3) 3131(3) 3134(2) 3138(2) 3143(3) 3149(2) 3155(3)
-------------------------------------------------------
3158(2) 3161(2) 3171(3) 3179(3) 3183(3) 3186(1) 3189(2)
-------------------------------------------------------
3194(2) 3198(2) 3202(1) 3234(2) 3245(2) 3252(1) 3273(2)
-------------------------------------------------------
3275(3) 3298(1) 3299(3) 3303(3) 3306(1) 3309(2) 3318(2)
-------------------------------------------------------
3322(1) 3326(2) 3329(2) 3345(2) 3346(1) 3350(2) 3351(3)
-------------------------------------------------------
3354(2) 3359(3) 3366(2) 3370(1) 3381(2) 3389(2) 3390(2)
-------------------------------------------------------
3401(2) 3411(3) 3412(1) 3413(2) 3414(2) 3431(3) 3434(2)
-------------------------------------------------------
3441(2) 3449(2) 3453(2) 3455(3) 3458(2) 3460(1) 3466(1)
-------------------------------------------------------
3468(1) 3473(2) 3474(2) 3485(2) 3490(1) 3491(3) 3495(3)
-------------------------------------------------------
3498(1) 3506(2) 3509(2) 3513(2) 3516(1) 3519(3) 3521(2)
-------------------------------------------------------
3532(1) 3534(2) 3538(1) 3539(3) 3546(1) 3551(3) 3554(2)
-------------------------------------------------------
3556(1) 3563(3) 3570(1) 3579(3) 3580(1) 3593(2) 3603(3)
-------------------------------------------------------
3605(2) 3609(2) 3612(1) 3614(2) 3618(2) 3621(2) 3623(3)
-------------------------------------------------------
3626(2) 3636(1) 3641(2) 3642(1) 3653(2) 3658(1) 3665(2)
-------------------------------------------------------
3674(2) 3676(1) 3690(1) 3700(1) 3705(2) 3708(1) 3725(2)
-------------------------------------------------------
3729(2) 3732(1) 3738(2) 3749(2) 3753(2) 3758(2) 3761(2)
-------------------------------------------------------
3770(2) 3773(2) 3774(2) 3778(1) 3779(3) 3786(2) 3791(3)
-------------------------------------------------------
3794(2) 3795(3) 3796(1) 3801(2) 3802(1) 3803(3) 3810(2)
-------------------------------------------------------
3821(2) 3834(2) 3843(3) 3845(2) 3850(1) 3851(3) 3852(1)
-------------------------------------------------------
3858(2) 3863(3) 3876(1) 3878(2) 3879(3) 3894(2) 3906(1)
```

```
-------------------------------------------------------
3911(3) 3914(2) 3916(1) 3922(1) 3926(2) 3930(1) 3938(2)
-------------------------------------------------------
3939(3) 3941(2) 3946(1) 3950(2) 3953(2) 3959(3) 3966(2)
-------------------------------------------------------
3974(2) 3975(3) 3988(1) 4002(1) 4012(1) 4018(1) 4019(3)
-------------------------------------------------------
4020(1) 4026(2) 4034(2) 4043(3) 4046(2) 4055(3) 4058(2)
-------------------------------------------------------
4061(2) 4073(2) 4085(2) 4089(2) 4090(1) 4092(1) 4098(1)
-------------------------------------------------------
4109(2) 4110(2) 4115(3) 4118(2) 4121(2) 4131(3) 4132(1)
-------------------------------------------------------
4134(2) 4138(1) 4145(2) 4146(2) 4155(2) 4156(1) 4181(2)
-------------------------------------------------------
4193(2) 4211(2) 4214(2) 4215(3) 4218(1) 4221(2) 4223(2)
-------------------------------------------------------
4228(1) 4233(2) 4242(1) 4252(1) 4256(2) 4258(1) 4260(1)
-------------------------------------------------------
4263(3) 4269(2) 4271(3) 4281(2) 4282(1) 4286(2) 4298(2)
-------------------------------------------------------
4299(2) 4304(2) 4311(3) 4313(2) 4323(3) 4331(3) 4334(2)
-------------------------------------------------------
4338(2) 4346(2) 4348(1) 4349(2) 4356(1) 4362(1) 4365(2)
-------------------------------------------------------
4370(2) 4372(1) 4373(2) 4391(3) 4396(1) 4401(2) 4403(2)
-------------------------------------------------------
4409(2) 4410(2) 4418(2) 4419(3) 4430(2) 4433(2) 4443(2)
-------------------------------------------------------
4450(1) 4461(2) 4466(2) 4475(3) 4481(2) 4482(1) 4484(2)
-------------------------------------------------------
4485(2) 4492(1) 4499(2) 4503(3) 4505(2) 4506(1) 4514(2)
-------------------------------------------------------
4516(1) 4524(2) 4529(2) 4546(1) 4551(3) 4563(3) 4575(2)
-------------------------------------------------------
4580(2) 4586(2) 4590(2) 4599(3) 4601(2) 4602(1) 4610(2)
-------------------------------------------------------
4613(2) 4619(3) 4620(1) 4628(2) 4636(1) 4641(2) 4646(2)
-------------------------------------------------------
4655(3) 4659(3) 4661(2) 4668(2) 4670(2) 4671(2) 4674(2)
-------------------------------------------------------
4685(2) 4690(1) 4695(3) 4698(2) 4709(2) 4710(2) 4716(2)
-------------------------------------------------------
```

```
4718(2) 4722(1) 4733(2) 4739(2) 4745(2) 4766(2) 4769(2)
-------------------------------------------------------
4773(2) 4775(3) 4786(1) 4788(1) 4793(2) 4800(2) 4806(2)
-------------------------------------------------------
4809(2) 4812(1) 4814(2) 4821(2) 4830(2) 4838(2) 4839(3)
-------------------------------------------------------
4848(2) 4866(2) 4871(3) 4874(2) 4876(1) 4883(2) 4895(3)
-------------------------------------------------------
4901(2) 4916(2) 4925(2) 4929(2) 4932(1) 4935(3) 4941(2)
-------------------------------------------------------
4943(3) 4950(2) 4953(2) 4956(1) 4961(2) 4970(2) 4972(1)
-------------------------------------------------------
4974(2) 4983(3) 4986(1)
-------------------------------------------------------


-------------------------------------------------------
5002(1) 5010(1) 5018(2) 5033(2) 5034(2) 5039(3) 5045(2)
-------------------------------------------------------
5046(2) 5049(2) 5050(1) 5051(3) 5055(2) 5058(1) 5066(2)
-------------------------------------------------------
5069(2) 5070(2) 5075(2) 5076(1) 5081(2) 5088(2) 5090(2)
-------------------------------------------------------
5098(1) 5106(1) 5111(2) 5123(2) 5126(2) 5129(2) 5133(2)
-------------------------------------------------------
5135(3) 5136(2) 5144(2) 5146(1) 5150(2) 5156(2) 5165(2)
-------------------------------------------------------
5170(1) 5171(2) 5178(1) 5184(2) 5188(1) 5195(3) 5199(2)
-------------------------------------------------------
5213(2) 5226(1) 5229(2) 5231(3) 5238(2) 5243(3) 5249(2)
-------------------------------------------------------
5250(2) 5260(1) 5279(3) 5283(2) 5294(2) 5300(2) 5303(2)
-------------------------------------------------------
5306(2) 5308(1) 5315(2) 5319(3) 5328(2) 5332(1) 5333(2)
-------------------------------------------------------
5343(2) 5345(2) 5354(2) 5361(2) 5366(2) 5386(1) 5394(2)
-------------------------------------------------------
5399(2) 5415(3) 5418(2) 5423(3) 5426(2) 5429(2) 5430(2)
-------------------------------------------------------
5433(2) 5441(2) 5442(1) 5444(2) 5445(2) 5454(2) 5474(2)
-------------------------------------------------------
5476(1) 5482(1) 5486(2) 5489(2) 5493(2) 5500(1) 5501(2)
-------------------------------------------------------
```

```
5506(1) 5513(2) 5523(2) 5528(2) 5534(2) 5535(2) 5541(2)
-------------------------------------------------------
5543(2) 5546(2) 5556(1) 5562(1) 5565(2) 5572(1) 5579(3)
-------------------------------------------------------
5585(2) 5588(2) 5598(2) 5606(2) 5619(2) 5630(2) 5636(2)
-------------------------------------------------------
5639(2) 5650(1) 5655(3) 5658(12) 5675(3) 5682(1) 5684(2)
-------------------------------------------------------
5691(3) 5692(1) 5699(3) 5700(1) 5711(3) 5716(1) 5718(2)
-------------------------------------------------------
5721(2) 5740(1) 5741(2) 5748(12) 5751(3) 5759(3) 5763(2)
-------------------------------------------------------
5774(2) 5778(1) 5789(2) 5793(2) 5796(2) 5808(2) 5810(2)
-------------------------------------------------------
5812(1) 5816(2) 5826(1) 5838(2) 5842(1) 5844(2) 5849(2)
-------------------------------------------------------
5850(1) 5858(2) 5859(2) 5868(1) 5871(3) 5889(2) 5891(3)
-------------------------------------------------------
5894(2) 5900(2) 5903(3) 5906(2) 5910(2) 5913(2) 5915(3)
-------------------------------------------------------
5919(3) 5922(1) 5933(2) 5938(1) 5943(2) 5948(2) 5951(3)
-------------------------------------------------------
5954(2) 5963(2) 5966(2) 5969(2) 5984(2) 5986(1) 5990(2)
-------------------------------------------------------
5993(2) 6005(2) 6010(1) 6020(2) 6021(2) 6024(2) 6028(1)
-------------------------------------------------------
6035(2) 6048(2) 6052(1) 6053(2) 6059(2) 6066(1) 6071(3)
-------------------------------------------------------
6074(2) 6078(2) 6080(2) 6098(2) 6100(1) 6101(2) 6105(2)
-------------------------------------------------------
6113(2) 6119(2) 6125(2) 6126(2) 6130(1) 6131(2) 6134(2)
-------------------------------------------------------
6138(2) 6140(2) 6150(2) 6161(2) 6172(1) 6173(2) 6186(2)
-------------------------------------------------------
6189(2) 6195(2) 6196(1) 6200(2) 6202(1) 6206(2) 6210(12)
-------------------------------------------------------
6216(2) 6218(2) 6225(2) 6228(1) 6236(2) 6243(3) 6245(2)
-------------------------------------------------------
6255(3) 6268(1) 6269(2) 6273(2) 6276(12) 6288(2) 6294(2)
-------------------------------------------------------
6298(1) 6305(2) 6306(2) 6309(2) 6316(1) 6318(2) 6322(1)
-------------------------------------------------------
6323(2) 6326(2) 6329(2) 6348(2) 6369(2) 6371(3) 6372(1)
```

```
---------------------------------------------------------
6378(1) 6381(2) 6388(1) 6390(2) 6395(3) 6396(1) 6399(2)
---------------------------------------------------------
6404(2) 6410(2) 6414(2) 6426(2) 6444(2) 6449(2) 6453(2)
---------------------------------------------------------
6455(3) 6458(2) 6459(2) 6461(2) 6468(1) 6470(2) 6479(2)
---------------------------------------------------------
6483(2) 6489(2) 6490(1) 6491(3) 6500(2) 6518(2) 6521(2)
---------------------------------------------------------
6524(2) 6546(1) 6551(2) 6554(2) 6573(2) 6575(2) 6579(2)
---------------------------------------------------------
6581(2) 6591(2) 6593(2) 6614(2) 6618(1) 6620(2) 6636(1)
---------------------------------------------------------
6645(2) 6652(1) 6654(2) 6656(2) 6658(1) 6663(3) 6665(2)
---------------------------------------------------------
6669(2) 6690(1) 6698(2) 6700(1) 6705(2) 6708(1) 6725(2)
---------------------------------------------------------
6728(2) 6731(2) 6732(1) 6734(2) 6738(2) 6743(2) 6756(2)
---------------------------------------------------------
6761(2) 6762(1) 6768(2) 6778(1) 6780(1) 6783(2) 6788(2)
---------------------------------------------------------
6798(2) 6802(1) 6806(2) 6809(2) 6813(2) 6824(2) 6826(1)
---------------------------------------------------------
6828(1) 6839(2) 6843(2) 6845(2) 6848(2) 6854(2) 6855(3)
---------------------------------------------------------
6861(2) 6864(2) 6868(1) 6878(2) 6879(2) 6881(2) 6882(1)
---------------------------------------------------------
6898(1) 6899(2) 6903(2) 6906(1) 6914(2) 6915(2) 6916(1)
---------------------------------------------------------
6920(2) 6929(2) 6938(2) 6939(2) 6941(2) 6946(1) 6948(1)
---------------------------------------------------------
6950(2) 6953(2) 6960(2) 6965(2) 6966(2) 6970(1) 6983(3)
---------------------------------------------------------
6998(2) 6999(2) 7004(2) 7005(2) 7012(1) 7014(2) 7016(2)
---------------------------------------------------------
7018(1) 7025(2) 7026(1) 7040(2) 7042(1) 7043(3) 7053(2)
---------------------------------------------------------
7068(1) 7071(2) 7074(2) 7076(2) 7079(3) 7086(2) 7103(3)
---------------------------------------------------------
7108(1) 7110(2) 7121(2) 7146(2) 7151(2) 7170(2) 7173(2)
---------------------------------------------------------
7186(1) 7193(2) 7194(2) 7203(2) 7205(2) 7209(2) 7210(1)
---------------------------------------------------------
```

7211(2) 7218(1) 7223(2) 7228(1) 7230(2) 7236(1) 7242(1)

7252(1) 7259(2) 7266(2) 7268(2) 7271(3) 7274(2) 7275(3)

7278(2) 7282(1) 7306(1) 7310(2) 7313(2) 7314(2) 7319(2)

7326(2) 7328(2) 7330(1) 7334(2) 7348(1) 7349(2) 7358(2)

7361(2) 7368(2) 7370(2) 7373(2) 7379(2) 7385(2) 7389(2)

7398(2) 7406(2) 7410(1) 7413(2) 7415(2) 7421(2) 7425(2)

7433(2) 7434(2) 7439(2) 7443(3) 7445(2) 7450(1) 7458(1)

7461(2) 7464(2) 7469(2) 7473(2) 7476(1) 7478(2) 7484(2)

7491(2) 7498(1) 7506(1) 7515(2) 7516(1) 7522(1) 7526(2)

7530(2) 7538(2) 7540(1) 7541(2) 7545(2) 7546(1) 7548(1)

7550(2) 7553(2) 7565(2) 7569(2) 7572(1) 7574(2) 7580(2)

7586(2) 7588(1) 7593(2) 7602(1) 7613(2) 7620(1) 7629(2)

7631(2) 7634(2) 7642(1) 7643(2) 7649(2) 7656(2) 7659(3)

7665(2) 7668(1) 7674(2) 7686(2) 7688(2) 7690(1) 7691(2)

7695(3) 7706(2) 7713(2) 7716(1) 7719(2) 7721(2) 7730(2)

7746(2) 7748(2) 7755(3) 7756(1) 7763(2) 7784(2) 7788(1)

7790(2) 7803(2) 7814(2) 7820(2) 7821(2) 7823(2) 7824(2)

7828(1) 7830(2) 7833(2) 7835(2) 7841(2) 7852(1) 7863(2)

7865(2) 7868(2) 7869(2) 7874(2) 7876(1) 7880(2) 7882(1)

7883(3) 7886(2) 7893(2) 7895(2) 7898(2) 7900(1) 7901(2)

7904(2) 7906(1) 7911(2) 7929(2) 7932(1) 7940(2) 7943(3)

7948(1) 7950(2) 7953(2) 7959(2) 7961(2) 7979(3) 7985(2)

```
-------------------------------------------------------
7995(2) 8000(2) 8003(3) 8016(2) 8031(2) 8033(2) 8034(2)
-------------------------------------------------------
8043(3) 8051(2) 8052(1) 8063(2) 8068(1) 8069(2) 8092(1)
-------------------------------------------------------
8093(2) 8094(2) 8096(2) 8108(2) 8111(2) 8114(2) 8115(3)
-------------------------------------------------------
8116(1) 8122(1) 8126(2) 8133(2) 8136(2) 8146(1) 8150(2)
-------------------------------------------------------
8159(2) 8166(2) 8169(2) 8170(1) 8174(2) 8178(1) 8181(2)
-------------------------------------------------------
8190(2) 8205(2) 8210(2) 8213(2) 8218(1) 8220(1) 8223(3)
-------------------------------------------------------
8226(2) 8236(1) 8238(2) 8240(2) 8242(1) 8243(3) 8246(2)
-------------------------------------------------------
8259(2) 8264(2) 8268(1) 8273(2) 8280(2) 8283(3) 8286(2)
-------------------------------------------------------
8290(1) 8292(1) 8301(2) 8303(3) 8309(2) 8310(1) 8325(2)
-------------------------------------------------------
8336(2) 8346(2) 8349(2) 8351(2) 8362(1) 8373(2) 8381(2)
-------------------------------------------------------
8386(1) 8393(2) 8411(2) 8414(2) 8421(2) 8422(1) 8428(1)
-------------------------------------------------------
8441(2) 8442(1) 8446(1) 8450(2) 8451(3) 8465(2) 8466(1)
-------------------------------------------------------
8471(3) 8481(2) 8489(2) 8490(2) 8493(2) 8505(2) 8510(2)
-------------------------------------------------------
8512(1) 8513(2) 8523(2) 8526(2) 8538(1) 8546(2) 8549(2)
-------------------------------------------------------
8553(2) 8561(2) 8562(1) 8568(2) 8572(1) 8579(3) 8583(2)
-------------------------------------------------------
8591(3) 8594(2) 8596(1) 8598(1) 8601(2) 8603(2) 8604(2)
-------------------------------------------------------
8608(1) 8619(2) 8626(1) 8628(2) 8649(2) 8658(2) 8663(3)
-------------------------------------------------------
8666(2) 8668(1) 8675(3) 8676(1) 8679(2) 8692(1) 8693(2)
-------------------------------------------------------
8694(2) 8698(1) 8709(2) 8715(2) 8721(2) 8730(1) 8735(2)
-------------------------------------------------------
8738(2) 8740(1) 8741(2) 8746(1) 8754(2) 8759(3) 8786(2)
-------------------------------------------------------
8789(2) 8798(2) 8799(3) 8802(1) 8806(1) 8811(2) 8813(2)
-------------------------------------------------------
```

8818(1) 8820(1) 8834(2) 8836(1) 8841(2) 8853(2) 8856(2)

8866(1) 8873(2) 8874(2) 8886(1) 8891(2) 8894(2) 8903(3)

8913(2) 8918(2) 8919(3) 8922(1) 8925(2) 8931(3) 8932(1)

8945(2) 8951(3) 8954(2) 8955(3) 8961(2) 8962(1) 8968(1)

8969(2) 8970(1) 8978(2) 8979(3) 8988(2) 8990(2) 8994(2)

8998(1) 9006(2) 9010(1) 9021(2) 9023(2) 9028(1) 9029(2)

9038(2) 9044(2) 9048(1) 9058(1) 9059(2) 9066(2) 9071(3)

9074(2) 9084(2) 9090(2) 9095(3) 9105(2) 9114(2) 9125(2)

9126(2) 9134(2) 9144(2) 9150(1) 9155(2) 9160(1) 9164(2)

9172(1) 9180(1) 9183(3) 9189(2) 9202(1) 9206(2) 9213(2)

9216(2) 9220(1) 9221(2) 9226(1) 9230(2) 9240(1) 9246(2)

9256(1) 9260(2) 9270(2) 9276(2) 9282(1) 9292(1) 9293(2)

9296(2) 9318(2) 9322(1) 9330(2) 9335(3) 9336(1) 9340(1)

9342(1) 9348(1) 9350(2) 9356(2) 9359(3) 9365(2) 9370(1)

9371(3) 9374(2) 9378(2) 9386(2) 9393(2) 9396(1) 9418(1)

9419(3) 9420(1) 9429(2) 9432(1) 9434(2) 9436(1) 9449(2)

9455(3) 9458(2) 9459(3) 9466(1) 9473(2) 9478(1) 9479(3)

9486(2) 9489(2) 9490(1) 9504(2) 9506(2) 9518(2) 9525(2)

9532(1) 9534(2) 9538(1) 9539(2) 9543(2) 9546(1) 9569(2)

9570(2) 9578(2) 9581(2) 9586(1) 9590(2) 9600(1) 9603(3)

9606(2) 9609(2) 9612(1) 9615(2) 9618(1) 9624(2) 9628(1)

9629(2) 9642(1) 9650(2) 9659(3) 9660(1) 9666(2) 9676(1)

```
---------------------------------------------------------
9686(2) 9689(2) 9693(2) 9695(2) 9696(1) 9701(2) 9711(2)
---------------------------------------------------------
9713(2) 9716(2) 9720(2) 9731(2) 9732(1) 9734(2) 9741(2)
---------------------------------------------------------
9748(1) 9750(2) 9753(2) 9765(2) 9766(1) 9770(2) 9771(2)
---------------------------------------------------------
9776(2) 9785(2) 9791(2) 9798(2) 9801(2) 9802(1) 9830(2)
---------------------------------------------------------
9832(1) 9848(2) 9849(2) 9850(1) 9854(2) 9858(1) 9863(2)
---------------------------------------------------------
9869(2) 9875(3) 9881(2) 9882(1) 9900(1) 9906(1) 9909(2)
---------------------------------------------------------
9922(1) 9926(2) 9930(2) 9933(2) 9940(1) 9944(2) 9945(2)
---------------------------------------------------------
9948(1) 9959(2) 9963(2) 9968(2) 9974(2) 9981(2) 9986(2)
---------------------------------------------------------
9989(2) 9995(3) 9998(2)
---------------------------------------------------------
```

Приложение C

Дискретное преобразование Фурье

C.1 Быстрое преобразование Фурье над полем комплексных чисел и над простым конечным полем $GF(2^n + 1)$

Рассмотрим несколько способов вычисления значений многочлена в заданной точке или в нескольких различных точках, выбираемых из соображений упрощения вычислений. При вычислении значения многочлена в заданной точке, как известно, чтобы избежать повторений при вычислении степеней точки используют схему Горнера.

Пусть $p = a_0 + a_1x + \cdots + a_nx^n \in R[x]$, где $R[x]$ – коммутативное кольцо с единицей над полем F. Если $r \in F$, то

$$p(r) = a_0 + a_1r + \cdots + a_nr^n = a_0 + r(a_1 + \cdots + r(a_{n-1} + ra_n))...).$$

Например, для $p = 3 + 4x + 5x^2 + 6x^3 + 7x^4$ и $r = 2$ получим

$$p(2) = 3 + 2(4 + 2(5 + 2(6 + 7 \cdot 2))) = 191.$$

Если требуется вычислять значения многочлена в нескольких точках, выбор которых свободен, то лучше использовать другой способ. Рассмотрим его на примере кольца многочленов над полем C комплексных чисел.

Пусть $p = a_0 + a_1x + \cdots + a_{n-1}x^{n-1} \in C[x]$ и допустим, что надо вычислить значения многочлена в n точках, в выборе которых мы свободны. Продемонстрируем идею на примере.

Пусть $p = a_0 + a_1 + a_2x^2 + a_3x^3$. Выберем одну точку $c_1 \in C$ произвольно, и выберем c_2 как $c_2 := -c_1$. Представим $p = p_{even} + p_{odd}$, где $p_{even} = a_0 + a_2x^2$, $p_{odd} = a_1 + a_3x^3$. Ясно, что

$$p(c_1) = p_{even}(c_1) + p_{odd}(c_1), p(c_2) = p_{even}(c_2) + p_{odd}(c_2).$$

Теперь $p_{even} = (a_0 + a_2x) * x^2 := q_{even}(x^2)$, где $q_{even} = a_0 + a_2x$, и аналогично

$p_{odd} = x((a_1 + a_3x) * x^2 := xq_{odd}(x^2)$, где $q_{odd} = a_1 + a_3x$.

Как видим, q_{even} и q_{odd} – многочлены степени ≤ 2 (в общем случае $\leq n/2$) и

$$p(c_1) = q_{even}(c_1^2) + c_1q_{odd}(c_1^2), p(c_2) = q_{even}(c_1^2) + c_1q_{odd}(c_1^2).$$

Так вычисление значений двух многочленов степени $\leq n/2$ позволяет получить два значения многочлена p в точках c_1 и $-c_1$. Применяя такой же подход к вычислению значений многочленов q_{even} и q_{odd}, понижая степень до $\approx n/4$ и так далее. Следовательно, вместо вычисления p в c_1 и $-c_1$ мы делаем это в точках $c_1, ic_1, -c_1, -ic_1$.

Развивая этот подход, используя свободу выбора, выберем $c_1 = 1$, обозначим ω примитивный корень n-ой степени из единицы (например, $\omega = e^{\frac{2\pi i}{n}}$) и вычислим значения многочлена в точках $1, \omega, \omega^2, \ldots, \omega^{n-1}$. Эти точки различны, так как ω – примитивный корень.

Это стратегия и процедура знаменитого алгоритма Cooly-Tukey и они работают лучшим образом, если n является степенью двойки. Вычисление значений многочлена p степени n требует n сложений и n умножений, если мы используем идею 1. Сложность однократного вычисления асимптотически равна $O(n)$, если мы вычислять n значений, то сложность будет $O(n^2)$. При использовании алгоритма $Cooly - Tukey$ для числа $CT(n)$ выполняемых операций мы имеем рекуррентную формулу $CT(n) \leq 2CT(\frac{n}{2})$ (для вычисления чётных и нечётных частей) $+\frac{3n}{2}$ (для соединения результатов при получении значений для полинома в целом). Ясно, что $CT(1) = 0$, и мы получаем $CT(n) \leq 1.5n \log_2 n$, – по индукции:

$$CT(2n) \leq 2CT(n) + 3 \leq 3n \log n + 3n = 1.5(2n) \log n + 1.5(2n) \leq 1.5(2n) \log(2n).$$

Cooley-Tuckey имеет сложность $O(n \log n)$, которая реально лучше сложности $O(n^2)$: при $n = 1\ 000\ 000$, $n^2 = 10^{12}$ более чем в 50 000 раз превышает значение $n \log n$.

аналогично можно поступать и при использовании любого другого поля F (вместо C), имеющего примитивный корень n-ой степени из единицы. При этом обобщении не возникает никаких проблем. Пусть F есть такое поле и ω есть корень n-ой степени из единицы. Для практических целей, возможно, неудобно сохранять соглашение и о разбиениях $p = p_{even} + p_{odd}$. Мы исправим это положение следующим образом.

Теорема С.1.1 *Пусть $p \in F[x]$. Если r является остатком от деления p на $x - \omega^i$, то*

$$p(\omega^i) = r(\omega^i).$$

Доказательство. Если $p = (x-\omega^i)q+r$, мы имеем $p(\omega^i) = (\omega^1-\omega^i)q(\omega^i)+r\omega^i = r(\omega^i)$.

Теперь напомним, что $x^n - 1 = \Pi_{i=0}^{n-1}(x - \omega^i)$. Так в $C[x]$, например,

$$x^4 - 1 = (x^2 - 1)(x^2 - i^2) = (x - 1)(x - (-1))(x - i)(x - (-1)).$$

Если k и m четны, то

$$x^k - \omega^m = (x^{k/2} - \omega^{m/2})(x^{k/2} + \omega^{m/2}) = (x^{k/2} - \omega^{m/2})(x^{k/2} - \omega^{m/2+n/2}).$$

Теорема С.1.2 *Если последовательно делить p на $x^k - \omega^m$ и затем полученный остаток на $x^{k/2} - \omega^{m/2}$ (k, m четны), то получится тот же остаток, как при однократном делении на $x^{k/2} - \omega^{m/2}$.*

Теорема С.1.3 *При делении $p = a_{n-1}x^{n-1} + \cdots + a_1x + a_0 \in F[x]$ на $x^k - c,\ k = n/2$, получается остаток $r = c(a_{n-1}x^{k-1} + \cdots + a_{k+1}x + a_k) + (a_{k-1}x^{k-1} + \cdots + a_1x + a_0) = \sum_{i=0}^{k-1}(a_j + ca_{j+k})x^j$.@*

Продемонстрируем использование этого результата двумя примерами.

Пример С.1.1 Пусть

$$p = (1+i)x^3 + 5ix^2 + 4x + 3 \in C[x],\ \text{т.е. } n = 4.$$

$$\begin{array}{cccc}
 & & x^4-1 & \\
 & & /\backslash & \\
 & x^2-1 & & x^2+1 = x^2-i^2 \\
 & /\backslash & & /\backslash \\
x-1 = x-d^0 = x-\omega^{\tilde{d^0}} = x-\omega^0 & & x-(-1) = x-d^1 = x-\omega^{\tilde{d^1}} = x-\omega^2 & \\
 & & x-i = x-d^2 = x-\omega^{\tilde{d^2}} = x-\omega^1 & x-(-i) = x-d^3 = x-\omega^{\tilde{d^3}} = x-\omega^3
\end{array}$$

Здесь $\tilde{d^i}$ – число, двоичная двух битовая запись которого получается инвертированием двоичной записи (d_0^i, d_1^i) числа $d^i : \tilde{d^i} = (\tilde{d_0^i}, \tilde{d_1^i}) = (d_1^i, d_0^i)$. Деление p на $x^4 - 1,\ x^2 - 1,\ x^2 - i^2,\ x - 1$, и т. д., дает остатки (вновь древовидная форма) с использованием последней теоремы:

$$\begin{array}{cccc}
 & & p & \\
 & & /\backslash & \\
 & ((1+i)x+5i)+(4x+3) & & -((1+i)x+5i)+(4x+3) \\
 & = (5+i)x+(3+5i) & & (3-i)x+(3-5i) \\
 & /\backslash & & /\backslash \\
8+6i = p(\omega^0) = p(1) = & & -2+4i = p(\omega^2) = p(-1) = & \\
 & & 4-2i = p(\omega^1) = p(i) = & 2-8i = p(\omega^3) = p(-i) =
\end{array}$$

На нижнем уровне мы имеем требуемые значения $p(1),\ p(-1),\ p(i),\ p(-i)$.

Пример С.1.2 Возьмём $p = 4x^2 + 3x + 2 \in Z_5[x]$. Дополняя многочлен p до степени x^{2^m-1}, запишем

$$p = 0x^3 + 4x^2 + 3x + 2,$$

таким образом, $n = 4$, и мы находим $\omega = 2$ как примитивный корень четвертой степени из единицы, то есть как производящий элемент циклической группы Z_5^*. Мы получаем $2^1 = 2,\ 2^2 = 4,\ 2^3 = 3,\ 2^4 = 1$, так что

$$\begin{array}{cccc}
 & & x^4-1 & \\
 & & /\backslash & \\
 & x^2-1 & & x^2-4 \\
 & /\backslash & & /\backslash \\
x-1 = x-\omega^0 & & x-4 = x-\omega^2 & \\
 & & x-2 = x-\omega^1 & x-3 = x-\omega^3.
\end{array}$$

Используя последнюю теорему еще раз, получим

$$\begin{array}{ccc} & p & \\ & /\backslash & \\ 4+(3x+2) & & -4+(3x+2) \\ =3x+1 & & =3x+3 \\ /\backslash & & /\backslash \\ p(\omega^0)= \quad p(\omega^2)= & & p(\omega^1)= \quad p(\omega^3)= \\ p(1)= \quad p(4)= & & p(2)= \quad p(3)= \\ 4 \quad 3 & & 4 \quad 2 \end{array}$$

Таким образом, $p(1)=4,\ p(4)=3,\ p(2)=4,\ p(3)=2.$

Отметим, что использованные бинарные древовидные структуры можно построить при любом n, являющемся степенью двойки. При этом листья дерева разностей будут линейно упорядочены по правилу $i<j \to (x-\omega^{\tilde{i}})$ предшествует разности $(x-\omega^{\tilde{j}})$.

Теорема С.1.4 *Обозначим*

$$q_{lm} = \Pi_{i=l}^{l+2m-1}(x-c_j),$$

где $c_j = \omega^{\tilde{j}}$ *тогда*

$$q_l m = x^{2^m} - \omega^{i/\tilde{2}^m}.$$

Таким образом, дерево разностей q_{lm} можно построить, поднимаясь от листьев к корню.

Примечание. Предлагается подумать, как изменить дерево, если n не является степенью двойки.

Заметим, что p однозначно определяется значениями $p(1) = p(\omega^0), p(\omega^1), \ldots, p(\omega^{(n-1)}$. Таким образом можно заменить p последовательностью

$$(p(1), p(\omega^1), \ldots, p(\omega^{(n-1)}).$$

Определение 1.0. Пусть $p = a_0 + a_1x + \cdots + a_{n-1}x^{n-1} \in F[x]$ и $\omega \in F$ является примитивным корнем n-ой степени из единицы. Тогда

$$\hat{p} := p(\omega^0) + p(\omega^1)x + \cdots + p(\omega^{n-1})x^{n-1} =: \hat{a}_0 + \hat{a}_1 + \cdots + \hat{a}_{n-1}x^{n-1}$$

называется дискретным преобразованием Фурье (ДПФ) многочлена p, а коэффициенты $\hat{a}_i$ называются коэффициентами Фурье многочлена p.

Напомним, что посредством интерполяции можно восстановить p по $\hat{p}$. Любой вектор $(a_0, \ldots, a_{n-1}) \in F^n$ можно записать в виде многочлена $p = a_0 + a_1x + \cdots + a_{n-1}x^{n-1} \in F[x]$. Отсюда понятно, что такое *преобразование Фурье* вектора или последовательности. Так как для

$$\mathbf{a} = (a_0, \ldots, a_{n-1}) = a_0 + \ldots + a_{n-1}x^{n-1}$$

мы имеем

$$p(\omega^k) = a_0 + a_1\omega^k + a_2\omega^{2k} + \cdots + a_{n-1}\omega^{(n-1)k},$$

можно записать

$$\hat{\mathbf{a}} = \begin{pmatrix} \hat{a}_0 \\ \hat{a}_1 \\ \hat{a}_2 \\ \cdot \\ \cdot \\ \cdot \\ \hat{a}_{n-1} \end{pmatrix} = \mathbf{D_n a} =$$

$$= \begin{pmatrix} 1 & 1 & 1 & 1 & 1 \\ 1 & \omega & \omega^2 & \cdots & \omega^{n-1} \\ 1 & \omega^2 & \omega^4 & \cdots & \omega^{2(n-1)} \\ \cdots & \cdots & \cdots & \cdots & \cdots \\ 1 & \omega^{n-1} & \omega^{2(n-1)} & \cdots & \omega^{(n-1)(n-1)} \end{pmatrix} \begin{pmatrix} a_0 \\ a_1 \\ a_2 \\ \cdot \\ \cdot \\ \cdot \\ a_{n-1} \end{pmatrix}.$$

Определение 1.0. Матрица $\mathbf{D_n}$ называется *матрицей дискретного преобразования Фурье* порядка n (ДПФ-матрицей). Вектор $\hat{\mathbf{a}}$ называется *спектральным вектором* для *сигнального вектора* $\mathbf{a}$. Заметим, что D_n симметрична и зависит от выбора ω, при необходимости используется обозначение $\mathbf{D_{n,\omega}}$. Если мы работаем с многочленами, то обозначаем $\hat{p} = \mathbf{D_n} p$. Вычисление $\hat{\mathbf{a}}$ (или $\hat{\mathbf{p}}$) по схеме Горнера (идея 1) имеет сложность $O(n^2)$. Если используются быстрые алгоритмы (идея 3), то говорят о *быстром преобразовании Фурье*.

Для восстановления сигнального вектора по спектральному вектору используется обратное преобразование:

$$\mathbf{a} = \begin{pmatrix} a_0 \\ a_1 \\ a_2 \\ \cdot \\ \cdot \\ \cdot \\ a_{n-1} \end{pmatrix} = \tfrac{1}{n}\mathbf{D_n^*}\hat{\mathbf{a}} =$$

$$= \tfrac{1}{n}\begin{pmatrix} 1 & 1 & 1 & 1 & 1 \\ 1 & \omega^{-1} & (\omega^{-1})^2 & \cdots & (\omega^{-1})^{n-1} \\ 1 & (\omega^{-1})^2 & (\omega^{-1})^4 & \cdots & (\omega^{-1})^{2(n-1)} \\ \cdots & \cdots & \cdots & \cdots & \cdots \\ 1 & (\omega^{-1})^{n-1} & (\omega^{-1})^{2(n-1)} & \cdots & (\omega^{-1})^{(n-1)(n-1)} \end{pmatrix} \begin{pmatrix} \hat{a}_0 \\ \hat{a}_1 \\ \hat{a}_2 \\ \cdot \\ \cdot \\ \cdot \\ \hat{a}_{n-1} \end{pmatrix}.$$

Здесь D_n^* – матрица обратного преобразования Фурье, ω^{-1} – элемент поля, обратный элементу ω.

Пример С.1.3 Построим многочлен $\hat{p}$, применяя дискретное преобразование Фурье к многочлену

$$p = 0x^3 + 4x^2 + 3x + 2,$$

над полем $GF(5)$, используя $\omega = 2$ как примитивный корень четвертой степени из единицы, как в предыдущем примере, напомним, что $\omega^1 = 2,\ \omega^2 = 4,\ \omega^3 = 3,\ \omega^4 = 1$, так что

$$\hat{\mathbf{a}} = \begin{pmatrix} \hat{a}_0 \\ \hat{a}_1 \\ \hat{a}_2 \\ \hat{a}_3 \end{pmatrix} = \begin{pmatrix} 1 & 1 & 1 & 1 \\ 1 & 2 & 4 & 3 \\ 1 & 4 & 1 & 4 \\ 1 & 3 & 4 & 2 \end{pmatrix} \begin{pmatrix} 2 \\ 3 \\ 4 \\ 0 \end{pmatrix} = \begin{pmatrix} 4 \\ 4 \\ 3 \\ 2 \end{pmatrix}$$

Полученному спектральному вектору соответствует многочлен
$\hat{p} = 2x^3 + 3x^2 + 4x + 4$.

Восстановление сигнального вектора по спектральному происходит следующим образом:

$$\mathbf{a} = \begin{pmatrix} a_0 \\ a_1 \\ a_2 \\ a_3 \end{pmatrix} =$$

$$4^{-1} \begin{pmatrix} 1 & 1 & 1 & 1 \\ 1 & 3 & 4 & 2 \\ 1 & 4 & 1 & 4 \\ 1 & 2 & 4 & 3 \end{pmatrix} \begin{pmatrix} 4 \\ 4 \\ 3 \\ 2 \end{pmatrix} = 4^{-1} \begin{pmatrix} 3 \\ 2 \\ 1 \\ 0 \end{pmatrix} = \begin{pmatrix} 2 \\ 3 \\ 4 \\ 0 \end{pmatrix}$$

С.2 Алгоритм умножения многочленов p и q над полем $GF(p)$

Идея состоит в том, чтобы вычислить значения многочленов во многих точках $c_1,\ c_2, \ldots$ и найти многочлен f путем интерполирования в точках $(c_i,\ , p(c_i)q(c_i))$. Тогда $f = pq$.

Если $p = a_0 + a_1 x + \cdots + a_{n-1}x^{n-1},\ q = b_0 + b_1 x + \cdots + b_{m-1}x^{m-1} \in F[x]$, то $pq = c_0 + c_1 + \cdots + c_{n+m-2}$, где

$$c_k = \sum_{i=j=k} a_i b_j = \sum_{i+0}^{k} a_i b_{k-i}.$$

Произведение pq можно назвать также свёрткой многочленов p и q по аналогии со сверткой функций $f * g(t) = \int_{\inf}^{\inf} f(x)g(t-x)dx$ функций f и g. Подобно этому свертка последовательностей $(a_0, \ldots, a_{n-1}) * (b_0, \ldots, b_{n-1})$ определяется как $c_0, \ldots, c_{n+m-2}$, где c_k определяются как описано только что выше.

Будем обозначать

$$(a_0, \ldots, a_{n-1}) \cdot (b_0, \ldots, b_{n-1}) = (a_0 b_0, \ldots, a_{n-1} b_{n-1}),$$

полагая, что старшие $]\frac{n}{2}[$ коэффициентов – нулевые.

Теорема C.2.1 *(Быстрое умножение, быстрая свертка). Пусть* $p, q \in F[x]$, *тогда*

$$pq = \mathbf{D_n}^{-1}(\mathbf{D_n}p \cdot \mathbf{D_n}q),$$

где n *достаточно велико.*

Такое умножение можно выполнить со сложностью

$$O(n \log n),$$

в то время как обычное умножение имеет сложность $O(n^2)$.

Пример C.2.1 Возведем многочлен $2x+3$ над полем $GF(5)$ в квадрат. Для этого достаточно возвести в квадрат компоненты спектрального вектора

$$\hat{\mathbf{a}} = \begin{pmatrix} \hat{a}_0 \\ \hat{a}_1 \\ \hat{a}_2 \\ \hat{a}_3 \end{pmatrix} = \begin{pmatrix} 1 & 1 & 1 & 1 \\ 1 & 2 & 4 & 3 \\ 1 & 4 & 1 & 4 \\ 1 & 3 & 4 & 2 \end{pmatrix} \begin{pmatrix} 3 \\ 2 \\ 0 \\ 0 \end{pmatrix} = \begin{pmatrix} 0 \\ 2 \\ 1 \\ 4 \end{pmatrix}$$

Возводя покомпонентно в квадрат, получим

$$\hat{p^2} = (0, 4, 1, 1)$$

Обратным преобразованием Фурье получим

$$\mathbf{a} = \begin{pmatrix} a_0 \\ a_1 \\ a_2 \\ a_3 \end{pmatrix} =$$

$$= 4^{-1} \begin{pmatrix} 1 & 1 & 1 & 1 \\ 1 & 3 & 4 & 2 \\ 1 & 4 & 1 & 4 \\ 1 & 2 & 4 & 3 \end{pmatrix} \begin{pmatrix} 0 \\ 4 \\ 1 \\ 1 \end{pmatrix} = 4^{-1} \begin{pmatrix} 1 \\ 3 \\ 1 \\ 0 \end{pmatrix} = \begin{pmatrix} 4 \\ 3 \\ 4 \\ 0 \end{pmatrix}.$$

Полученный сигнальный вектор определяет многочлен

$$4x^2 + 2x + 4,$$

являющийся квадратом исходного многочлена.