

Лабораторна робота 1

Тема: Кодування з коригуванням помилок

ТЕОРЕТИЧНА ЧАСТИНА

Помилки у збережених даних можуть виникати з різних причин. Наприклад, сплеск напруги електроживлення зумовлює помилки в оперативній пам'яті або при передаванні даних у мережі, а порушення властивостей матеріалу носія інформації пристрою збереження при нагріванні, електромагнітному або механічному впливі веде до її зміни. Для захисту від таких помилок використовуються коди, що можуть виявляти та виправляти помилки. При цьому кожному слову в пам'яті особливим чином додаються додаткові біти. Коли слово зчитується, додаткові біти застосовуються для перевірки наявності помилок.

При кодуванні відбувається перетворення елементів даних у відповідні їм числа – кодові символи. Кожному елементу відповідає унікальна сукупність кодових символів, яка називається кодовою комбінацією. Множина можливих кодових символів називається кодовим алфавітом, а їх кількість m є основою коду.

Відстань Хемінга – це мінімальна кількість розбіжностей між двома різними припустимими кодовими словами. У перешкодостійкому кодуванні відстань Хемінга відіграє основну роль. Розглянемо наступний 4-бітний код:

Приклад 1. Найпростіший 4-бітний код з відстанню Хемінга, що дорівнює одиниці. Такий код широко використовується в комп'ютерній техніці, незважаючи на неможливість знайти помилки.

0 → 0000;	4 → 0100;	8 → 1000;	12 → 1100;
1 → 0001;	5 → 0101;	9 → 1001;	13 → 1101;
2 → 0010;	6 → 0110;	10 → 1010;	14 → 1110;
3 → 0011;	7 → 0111;	11 → 1011;	15 → 1111.

Це звичайний двійковий код, що вміщує у свої 4 біти 16 символів. За його допомогою можна закодувати 16 літер алфавіту. Як неважко переконатися, два будь-яких символи відрізняються щонайменше на один біт, отже, відстань

Хемінга для такого коду дорівнює одній одиниці, що умовно позначається як $d = 1$

Наведемо інший 4-бітний код:

Приклад 2. Найпростіший 4-бітний код з відстанню Хемінга, що дорівнює двом та здатний знайти одиночні помилки.

0 → 0000;	4 → 1001;
1 → 0011;	5 → 1010;
2 → 0101;	6 → 1100;
3 → 0110;	7 → 1111;

Два будь-яких символи цього коду відрізняються як мінімум у двох позиціях, за рахунок чого інформаційна ємність коду зменшилася з 16 до 8 символів. Вісім комбінацій у коді оголошено забороненими, завдяки чому код здатний виявити одиночні помилки.

Наприклад, у символі «1010» спотворимо один довільний біт. Нехай для визначеності це буде другий зліва біт, тоді змінений символ буде мати такий вигляд: «1110». Оскільки комбінація «1110» є забороненою, декодер зможе виявити наявність помилки, хоча не зможе її виправити. Для виправлення навіть одного-єдиного збійного біту потрібно збільшити відстань Хемінга як мінімум до трьох. Двійковий 4-бітний код с $d = 3$ здатний вмщати в собі лише два різних символи, тому є непрактичним.

Розглянемо 10-бітний код з $d = 5$.

Приклад 3. 10-бітний код, з відстанню Хемінга, що дорівнює 5, який здатний виявити 4-бітні помилки, а виправляти 2-бітні.

0000000000	0000011111	1111100000	1111111111
------------	------------	------------	------------

У символі «0000011111» змінимо два будь-яких біта, та отримаємо наприклад двійкову послідовність «0100110111». Оскільки така комбінація є забороненою, декодер встановлює, що відбулася помилка. Якщо кількість збійних біт менша відстані Хемінга хоч би наполовину, то декодер може гарантовано

відновити вихідний символ. У наведеному коді між двома будь-якими дозволеними символами існує не менше ніж п'ять розбіжностей, тому викривлення двох біт у дозволеному символі приводить до утворення нового символу. Відстань Хемінга між ним та оригінальним символом дорівнює числу біт, що змінено k . Відстань до найближчого сусіднього символу дорівнює: $d - k$ (в нашому випадку 3). Іншими словами, доки $d - k > k$ декодер може гарантовано відновити двійковий код символу. У тих випадках, коли $d > k > d - k$, успішне відновлення вже не гарантується, але за сприятливих обставин усе-таки виявляється принципово вірогідним. Наприклад, якщо в коді символу «0000011111» ушкодити чотири біти так, що буде мати вигляд «0100110101» і спробувати відновити його, то процес відновлення буде мати наступний вигляд:

Приклад 4. Схема відновлення 4-бітної помилки у 10-бітному коді з відстанню Хемінга, що дорівнює 5.

0000000000	0000011111	1111100000	1111111111
0100110101	0100110101	0100110101	0100110101
_____	_____	_____	_____
5 відмінностей	4 відмінності	6 відмінностей	5 відмінностей

Знайшовши помилку, декодер послідовно звіряє ушкоджений символ з усіма дозволеними символами алфавіту, прагнучи знайти символ, який найбільше відповідає ушкодженому символу. Таким символом є той, що має найменше число відмінностей, а саме, кількість відмінностей не повинна перевищувати більше ніж у $d - 1$ позиціях порівняно з ушкодженим символом. У цьому випадку символ, що відновлюється збігається з правильним. Але якщо чотири ушкоджених біти будуть розподілені таким чином: «0**1111**11111», то декодер сприйняв би цей символ за «1111111111» і відновлення за наведеною схемою буде неправильним. Здатність коду до виправлення ушкоджень визначається наступним: для того щоб виявити r помилок, відстань Хемінга повинна бути більшою або дорівнювати $r + 1$, а для корегування r помилок відстань Хемінга повинна бути принаймні на одиницю більша ніж подвійна величина r (табл.1).

Таблиця 1

Здатність простого коду Хемінга до корегування помилок

<i>Характеристика</i>	<i>Величина</i>
виявлення помилок:	$d \geq r + 1$
виправлення помилок:	$d \geq 2r + 1$
інформаційна ємність:	$2^{n/d}$

Теоретично кількість помилок, що можна виявити, необмежена, практично інформаційна ємність кодових слів стрімко зменшуються зі зростанням d .

Розглянемо інший підхід, який запропонував Ричард Хемінг. Нехай усі біти, номери яких є ступенем двійки, будуть відігравати роль контрольних бітів, а ті що залишилися будуть інформаційними бітами повідомлення. Кожний контрольний біт має відповідати за парність суми для деякої належної йому групи бітів. При цьому один і той самий інформаційний біт може входити до різних груп, таким чином, один інформаційний біт буде впливати на декілька контрольних, а інформаційна ємність слова значно зросте. Для того щоб позначити, які біти контролюють інформаційний біт, що стоїть у позиції j , необхідно розкласти число j за ступенями двійки (табл.2). У двійковому коді праворуч записуються цифри молодших розрядів, а ліворуч – старших.

Розглянемо розрахунок контрольної суми 4-бітного символу «1010». Після резервування розрядів для контрольних бітів, які в тексті позначено літерами, символ буде мати вигляд: **D101C0BA**. Тепер відповідно до табл.2 обчислимо значення бітів, які позначено літерами А, В, С и D:

- біт А, контролює біти 3, 5 та 7 і дорівнює нулю, тому що їх сума ($0 + 1 + 1$) парна;
- біт В, контролює біти 3, 6 та 7 і дорівнює одиниці, тому що їх сума ($0 + 0 + 1$) непарна;

- біт С, контролює біти 5, 6 та 7 і дорівнює нулю, тому що їх сума ($1 + 0 + 1$) парна;
- біт D не відіграє ніякої ролі, тому що він може контролювати лише ті біти, які розташовано ліворуч від нього, а вони відсутні.

Таблиця 2

Розподіл бітів на контрольні та інформаційні для одного байта

Позиція	Якими бітами контролюються	
1(A)	$2^0=1$	Контрольний біт
2(B)	$2^1=2$	Контрольний біт
3	$2^0 + 2^1 = 1+2=3$	Контролюються бітами 1 і 2
4(C)	$2^2=4$	Контрольний біт
5	$2^0 + 2^2 = 1+4=5$	Контролюються бітами 1 і 4
6	$2^1 + 2^2 = 2+4=6$	Контролюються бітами 2 і 4
7	$2^0 + 2^1 + 2^2 = 1+2+4=7$	Контролюються бітами 1, 2 і 4
8(D)	$2^3=8$	Контрольний біт

Таким чином, отримане кодове слово буде мати такий вигляд: «1010010», де підкреслено контрольні біти.

Припустимо, що слово було викривлено в одній позиції і стало мати такий вигляд: 1110010, де підкреслено ушкоджений біт. Розглянемо можливість виявлення такої помилки. Біт А має дорівнювати 0, тому що сума 3-го, 5-го та 7-го бітів парна ($0 + 1 + 1$). Це відповідає істині. Біт В має дорівнювати 0, тому що сума 3-го, 6-го та 7-го бітів парна ($0 + 1 + 1$), але в слові він дорівнює одиниці. Біт С має дорівнювати 1, тому що сума 5-го, 6-го та 7-го бітів непарна ($1 + 1 + 1$), а в слові він дорівнює нулю. Отже, контрольні біти в позиціях 2 (біт В) та 4 (біт С) виявляють розбіжність з дійсністю. Їх сума $2 + 4 = 6$ відповідає позиції ушкодженого біту. Якщо його інвертувати, то код слова буде відновлено в первинному вигляді. Позиція помилки успішно виявляється і у випадку ушкодження одного з контрольних бітів, тому що кожний інформаційний біт контролюється декількома бітами, а один контрольний біт буде вказувати сам на себе.

Оскільки номери контрольних бітів становлять ступінь двійки, то із зростанням розрядності кодового слова вони розташовуються все рідше й рідше. Таким чином, із збільшенням розрядності кодового блоку, що оброблюється, ефективність кодів Хемінга зростає (табл.3).

Таблиця 3

Результат розрахунку ефективної інформаційної ємності кодів Хемінга для слів різної довжини

Величина кодового блоку	Кількість інформац. бітів	Ефективність
1	0	0.0 %
2	0	0.0 %
4	1	25.0 %
8	4	50.0 %
16	11	68.8 %
32	26	81.3 %
64	57	89.1 %
128	120	93.8 %
256	247	96.5 %
512	502	98.0 %
1024	1013	98.9 %
2048	2036	99.4 %
4096	4083	99.7 %
8192	8178	99.8 %
16384	16369	99.9 %
32768	32752	100.0 %
65536	65519	100.0 %
131072	131054	100.0 %
262144	262125	100.0 %
524288	524268	100.0 %

З наведеного видно, що при обробці блоків, які досягають 1024 біти та більше, накладними витратами на контрольні біти можна знехтувати.

ЗАВДАННЯ

1. Використовуючи таблицю ASCII кодів перекладіть своє прізвище або ім'я (але менш чим 5 літер) у послідовність цифр 16-річної системи числення, а потім у послідовність двійкових біт. Користуючись викладеним теоретичним матеріалом доповніть послідовність двійкових біт бітами коду Хемінга. Змініть значення N -ного біту отриманої послідовності на протилежне та покажіть можливість його відновлення (де N – це ваш номер за журналом академічної групи). У доповнення до N -того біту також змініть на протилежне значення $(35 - N)$ біту. Чи є можливість тепер відновити інформацію?
2. Розробити додаток, що дозволяє кодувати текстові файли за схемою «ASCII-код \rightarrow двійковий код \rightarrow код Хемінга» та відновлювати їх. Передбачити можливість задавання довжини бітових блоків, на які розбивається вміст файлу при кодування бітами Хемінгу.
3. Перевірте можливість додатку відновити текст файлу, якщо в блоках кодування змінюється не більше одного біту.
4. Підготуйте звіт з виконаними завданнями та необхідними поясненнями.

ЛИТЕРАТУРА

1. Таненбаум Э. Архитектура компьютера. – СПб.: Питер, 2002. – 704 с.
2. Блейхут Р. Теория и практика кодов, контролирующих ошибки. – М.: Мир, 1986. – 576 с.
3. Нортон П., Соухе Д. Язык ассемблера для IBM PC: Пер. с англ. М.: Изд-во "Компьютер", 1993. – 352 с.
4. Зубков С.В. Assembler. Для DOS, Windows и Unix. – М.: ДМК, 1999. – 640 с.
5. Гук М. Аппаратные средства персонального компьютера. Энциклопедия. - СПб: Питер, 2003.- 928с.
6. Мюллер С. Модернизация и ремонт ПК.– М.: Вильямс, 2003. – 1184 с.
7. Фролов А.В., Фролов Г.В. Аппаратное обеспечение персонального компьютера. – М.: ДИАЛОГ-МИФИ, 1998. – 304с.
8. Гук М. Интерфейсы ПК: справочник. – СПб: ПитерКом, 1999. – 416 с.
9. Чепурной В. Устройства хранения информации. – СПб: БХВ, 1998. – 208 с.
10. Барри Пресс. Ремонт и модернизация ПК. Библия пользователя.: Пер. с англ. – К., М., СПб: Диалектика, 1999.– 976с.
11. Дорот В.Л., Новиков Ф.А. Толковый словарь современной компьютерной лексики. – СПб: БХВ, 1999. – 352с.
12. Вебер Р. Конфигурирование ПК на процессорах Pentium, Pentium MMX, AMD. – М: Мир, 1998. – 416с.
13. <http://www.citforum.ru>
14. <http://www.emanual.ru>
15. <http://www.ixbt.com>
16. <http://www.intel.ru>

ДОДАТОК А

Табл. 128 символів ASCII-коду

Системи числення			Символ	Системи числення			Символ	Системи числення			Символ	Системи числення			Символ
8	10	16		8	10	16		8	10	16		8	10	16	
0	0	0	NUL	40	32	20	SPACE	100	64	40	@	140	96	60	`
1	1	1	SOH	41	33	21	!	101	65	41	A	141	97	61	a
2	2	2	STX	42	34	22	"	102	66	42	B	142	98	62	b
3	3	3	ETX	43	35	23	#	103	67	43	C	143	99	63	c
4	4	4	EOT	44	36	24	\$	104	68	44	D	144	100	64	d
5	5	5	ENQ	45	37	25	%	105	69	45	E	145	101	65	e
6	6	6	ACK	46	38	26	&	106	70	46	F	146	102	66	f
7	7	7	BEL	47	39	27	'	107	71	47	G	147	103	67	g
10	8	8	BS	50	40	28	(110	72	48	H	150	104	68	h
11	9	9	HT	51	41	29)	111	73	49	I	151	105	69	i
12	10	0A	LF	52	42	2A	*	112	74	4A	J	152	106	6A	j
13	11	0B	VT	53	43	2B	+	113	75	4B	K	153	107	6B	k
14	12	0C	FF	54	44	2C	,	114	76	4C	L	154	108	6C	l
15	13	0D	CR	55	45	2D	-	115	77	4D	M	155	109	6D	m
16	14	0E	SO	56	46	2E	.	116	78	4E	N	156	110	6E	n
17	15	0F	SI	57	47	2F	/	117	79	4F	O	157	111	6F	o
20	16	10	DLE	60	48	30	0	120	80	50	P	160	112	70	p
21	17	11	DC1	61	49	31	1	121	81	51	Q	161	113	71	q
22	18	12	DC2	62	50	32	2	122	82	52	R	162	114	72	r
23	19	13	DC3	63	51	33	3	123	83	53	S	163	115	73	s
24	20	14	DC4	64	52	34	4	124	84	54	T	164	116	74	t
25	21	15	NAK	65	53	35	5	125	85	55	U	165	117	75	u
26	22	16	SYN	66	54	36	6	126	86	56	V	166	118	76	v
27	23	17	ETB	67	55	37	7	127	87	57	W	167	119	77	w
30	24	18	CAN	70	56	38	8	130	88	58	X	170	120	78	x
31	25	19	EM	71	57	39	9	131	89	59	Y	171	121	79	y
32	26	1A	SUB	72	58	3A	:	132	90	5A	Z	172	122	7A	z
33	27	1B	ESC	73	59	3B	;	133	91	5B	[173	123	7B	{
34	28	1C	FS	74	60	3C	<	134	92	5C	\	174	124	7C	
35	29	1D	GS	75	61	3D	=	135	93	5D]	175	125	7D	}
36	30	1E	RS	76	62	3E	>	136	94	5E	^	176	126	7E	~
37	31	1F	US	77	63	3F	?	137	95	5F	_	177	127	7F	DEL

ДОДАТОК Б

Ричард Хемінг. Бібліографічна довідка

Ричард Веслі Хемінг народився 11 лютого 1915 р. у Чикаго. В 1937-му році він закінчив Чикагський університет і отримав ступінь бакалавра. Він продовжив освіту в університеті штату Небраска, де у 1939 р. йому було присвоєно наступну – магістерську ступінь. В 1942 р. Ричард став доктором філософії з математики в університеті штату Іллінойс.

В 1945 р. Хемінг брав участь в знаменитому Манхетенському дослідницькому проєкті, метою якого було створення атомної бомби.

З 1946 р. в лабораторії Белла він починає займатися конструюванням комп'ютерів. В цьому знаменитому центрі, де також працювали К. Шенон та інші видатні вчені, йому призначено було проробити майже тридцять років.

У 1976 р. Ричард переїжджає в місто Монтерей (шт. Каліфорнія) де очолює наукові дослідження з обчислювальної техніки у Вищому військово-морському училищі. В цьому училищі він викладає та пише книги з теорії ймовірностей і комбінаторики.

Піонерську роботу Хемінга було відзначено багатьма нагородами. В 1968-му році він став почесним членом Інституту інженерів з електротехніки та електроніки (ІЕЕЕ) та був нагороджений премією Т'юринга Асоціації комп'ютерних технологій. За винятковий вклад в розвиток інформаційних наук та систем в 1979 р. йому присуджено премію Еммануїла Піоре. У 1980-му його обрали членом Національної Академії інженерних наук, в 1981-му він одержав премію Гарольда Пендера від Пенсільванського університету, а в 1988-му – почесну медаль ІЕЕЕ. В 1996 р. в Мюнхені за працю з тем кодів, коригувальних помилок, Хемінга було удостоєно престижну премію Едуарда Рейма в розмірі \$130 000.

Помер Ричард Хемінг 7 січня 1998 р. у віці 82 років.

В його честь Інститут інженерів з електротехніки та електроніки заснував медаль, якою нагороджуються вчені, що внесли значний вклад в теорію інформації.

Ричарда Хемінга можна назвати “генієм однієї ідеї”. Він сформулював її в 1950 р. лише в одній науковій статті, присвяченій кодам для корекції помилок. Стаття містила конструкцію блочного коду, що корегує одиночні помилки, які виникають при передачі повідомлень. Ричард Хемінг постійно вів активні наукові дослідження, але знаменитою стала лише одна його робота з теорії інформації, яка складає за своїм об'ємом незначний відсоток його наукової творчості. Ця стаття швидко дістала світову популярність і принесла йому заслужену славу.

Подібно до того, як за відкриттями Фарадею і Максвелу з'явилися численні винаходи з електрозв'язку, які змінили наше життя, так і після створення Клодом Шеноном теорії інформації та Володимиром Котельниковим теорії потенціальної перешкодостійкості відкрилися нові можливості для розвитку телекомунікації. Одним з найважливіших розділів теорії інформації є теорія кодування, основи якої було закладено Хемінгом. Хемінг був першим, хто запропонував конструктивний метод побудови кодів з надмірністю та простим декодуванням. Його труд визначив напрямок більшості робіт в цій галузі, що з'явилися пізніше.