OpenMP

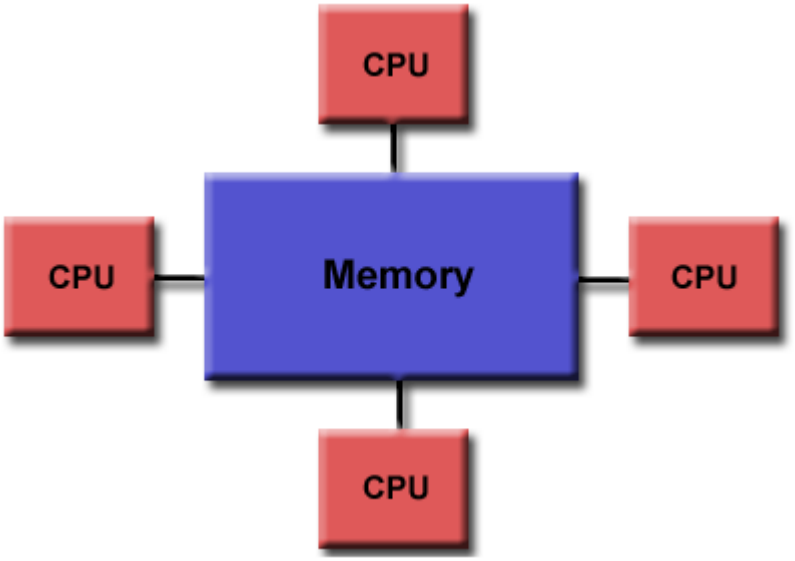
Чопоров Сергей Викторович

OpenMP

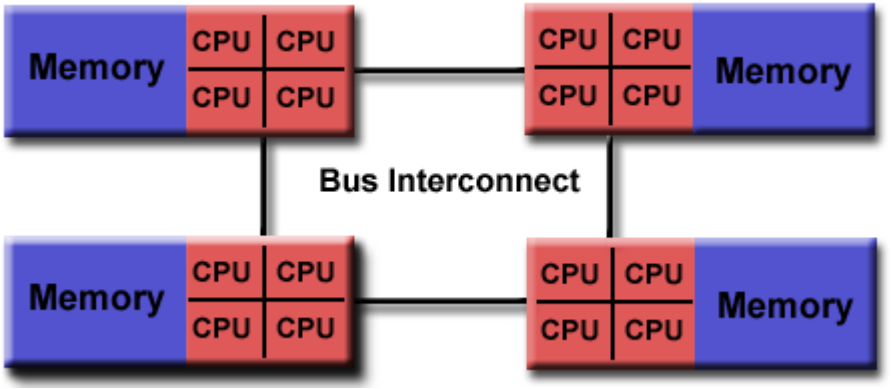
- Интерфейс прикладного программирования (API), который предназначен для программирования многопоточных приложений на многопроцессорных системах с общей памятью
- Состоит из трех основных компонент:
  - Директивы компилятора
  - Процедуры библиотеки времени выполнения
  - Переменные окружения
- Цели:
  - Стандартизация
  - Простота понимания
  - Простота использования
  - Портативность

Модели систем с общей памятью

- UMA — Uniform Memory Access - однородный доступ к памяти



- NUMA — Non-uniform Memory Access — неоднородный доступ к памяти

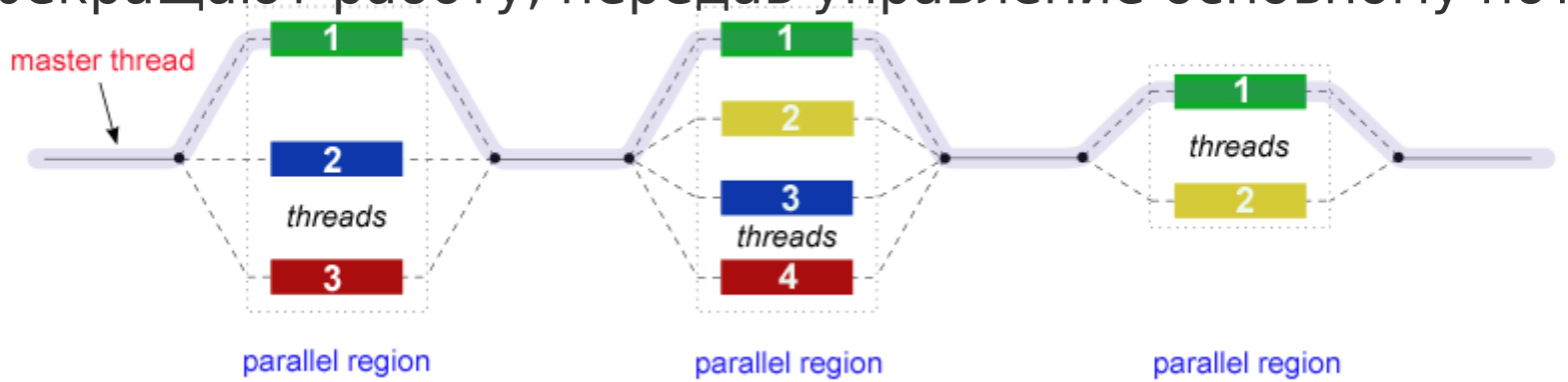


Программная модель

- Параллелизм на основе вычислительных потоков
- Явный параллелизм
- Fork-Join модель параллелизма
- Директивы компилятора вместо команд языка программирования
- Вложенный параллелизм
- Динамические вычислительные потоки
- Нет ограничений на ввод-вывод
- Релаксационная модель консистентности

Fork-Join модель параллелизма

- Все программы запускаются одним процессом: *основным потоком*. Основной поток выполняется последовательно пока конструкция первого параллельного блока не встречена
- Fork: основной поток создает множество параллельных потоков
- Команды из параллельного блока распределяются между потоками
- Join: когда потоки завершают выполнение инструкций из параллельного блока, они синхронизируются и прекращают работу, передав управление основному потоку



Обзор структуры

- Директивы компилятора (44):
  - Порождение параллельных регионов
  - Распределение блоков кода между потоками
  - Распределение итераций цикла между потоками
  - Сериализация секций кода
  - Синхронизация работы потоков

Обзор структуры

- Процедуры runtime-библиотеки (35):
  - Установка и получение количества потоков
  - Получение информации о потоке или наборе потоков
  - Установка и получение динамических свойств потоков
  - Получение информации о нахождении в параллельной секции (на каком уровне)
  - Задание и получение вложенных потоков
  - Установка, инициализация и завершение блокировок и вложенных блокировок

Обзор структуры

- Переменные окружения (13):
  - Установка количества потоков
  - Определения способа разделения итераций цикла
  - Разрешение/запрещение вложенного параллелизма; установка максимального уровня вложенности
  - Разрешение/запрещение динамических потоков
  - Установка размера стека потока
  - Установка политики ожидания потоков

Общая структура кода приложения

```
#include <omp.h>

int main ()
{
   int var1, var2, var3;
   // Serial code

   // Beginning of parallel region. Fork a team of threads.
   // Specify variable scoping

   #pragma omp parallel private(var1, var2) shared(var3)
   {
      // Parallel region executed by all threads
      // Other OpenMP directives
      // Run-time Library calls
      // All threads join master thread and disband
   }
   // Resume serial code
}
```

Компиляция

- *GNU*: gcc, g++, g77, gfortran **-fopenmp**
- *Intel*: icc, icpc, ifort **-openmp** (*Linux, MacOSX*), **-Qopenmp** (*Windows*)
- *Microsoft*: cl.exe **-openmp** (*Настройки проекта в Visual Studio*)

Формат директив C/C++

#pragma omp	directive-name	[clause, ...]	newline
Требуется для всех директив	Директива OpenMP	Необязательные условия/опции в произвольном порядке	Разделитель новой строки определяет блок кода для применения директивы

- Пример:
  - `#pragma omp parallel default(shared) private(beta,pi)`
- Общие правила:
  - Чувствительность к регистру
  - Соглашения C/C++ действительны
  - Одна директива в одной строке
  - Директива применяется к одному логическому блоку