программирование

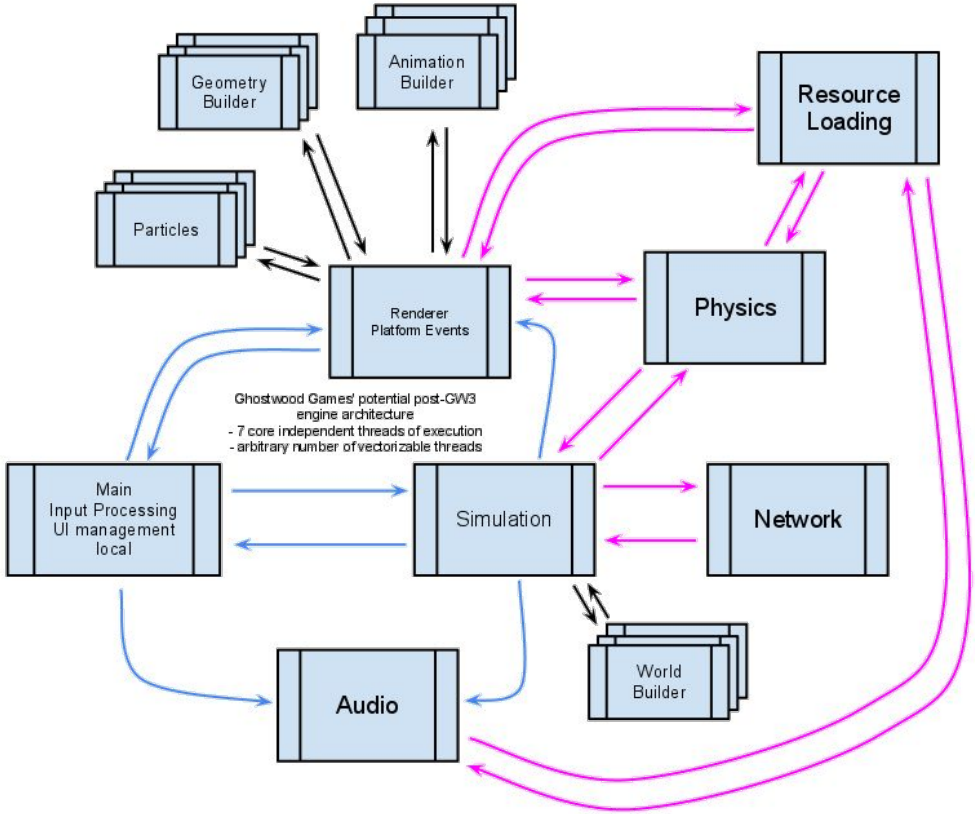
ВЗАИМОДЕЙСТВИЕ МЕЖДУ ПРОЦЕССАМИ

План лекции

- Модели вычислителей с общей и разделяемой памятью
- Реализация вычислений с разделяемой памятью
  - Файлы
  - Пайпы
  - Мэйлслоты
  - Общая память
  - Сокеты

Примеры параллельных вычислений

Локальные



Примеры параллельных вычислений

Глобальные

Galaxy Formation

Planetary Movments

Climate Change

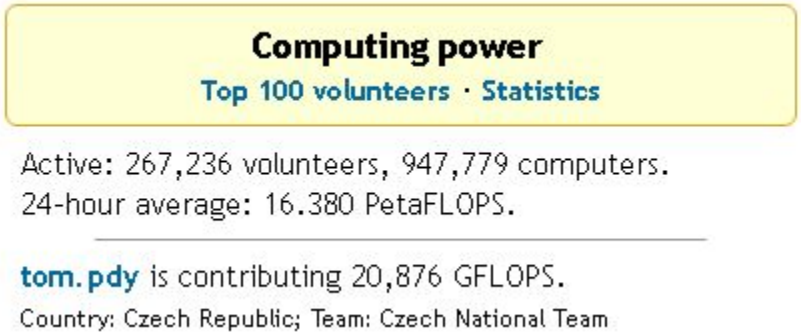
Rush Hour Traffic

Plate Tectonics

Weather

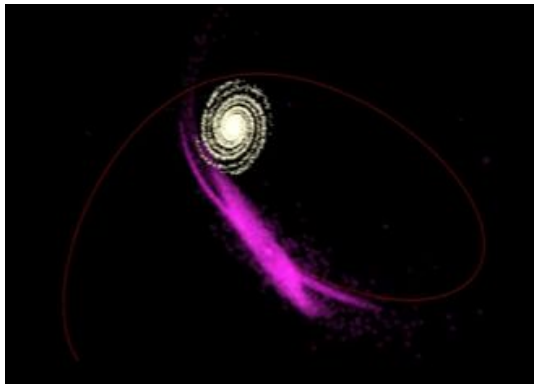
Open-source software for volunteer computing

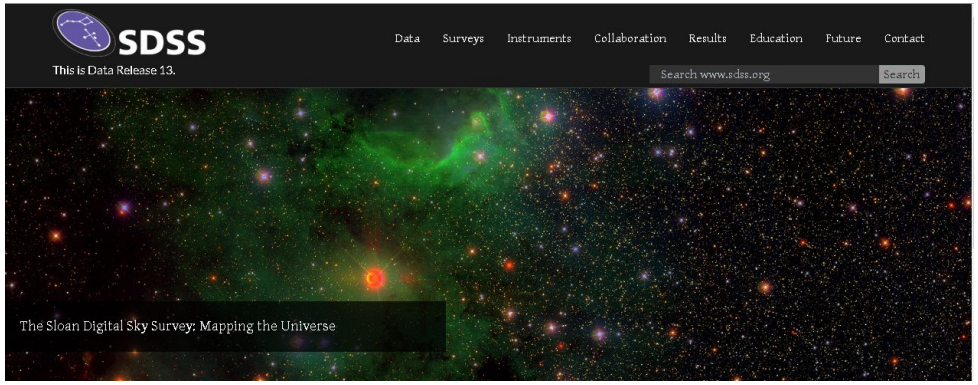
1. Choose projects
2. Download BOINC software
3. Enter an email address and password.



MilkiWay project

Milkyway@Home uses the BOINC platform to harness volunteered computing resources, creating a highly accurate three dimensional model of the Milky Way galaxy using data gathered by the Sloan Digital Sky Survey. This project enables research in both astroinformatics and computer science.

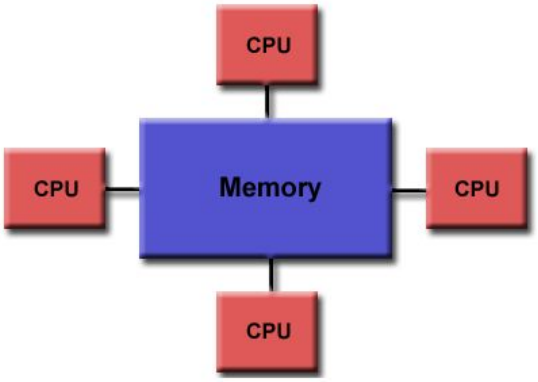


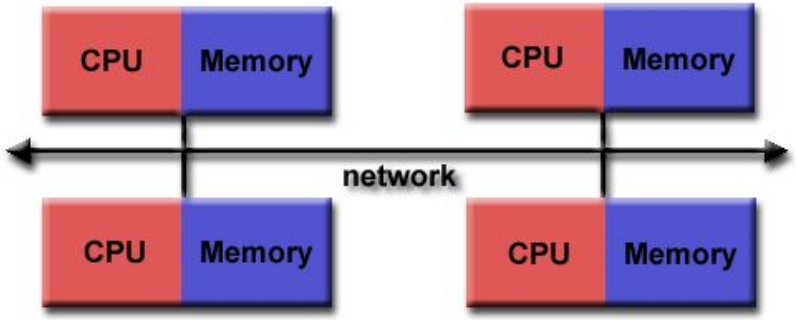


Архитектура памяти параллельных вычислителей

Две основных модели:

- С общей памятью
- С разделяемой памятью





Архитектура памяти параллельных вычислителей

1. Общая память (Shared Memory)

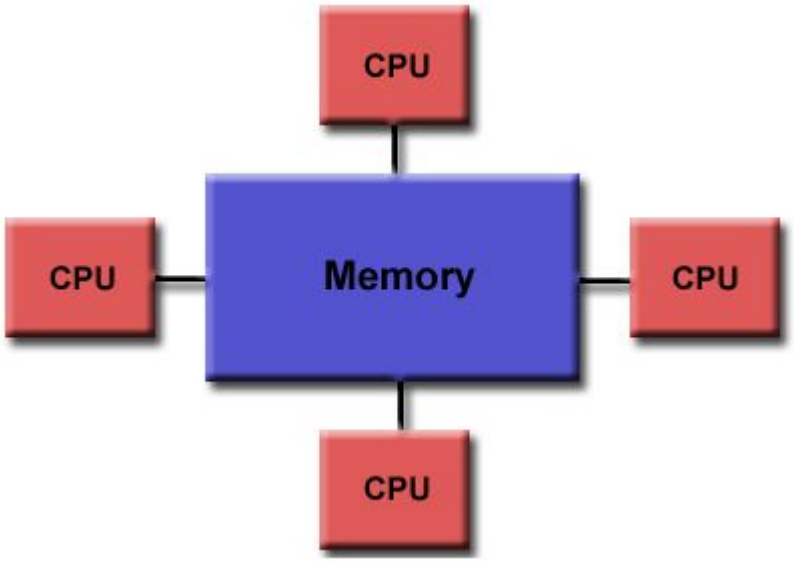
Общая память (1/4)

- Все процессоры имеют доступ к общей памяти
- Процессоры работают независимо
  - Исключение hyperthreading
- Изменение памяти одним из процессоров видят все остальные процессоры

Общая память Uniform Memory Access (2/4)

Несколько однотипных процессоров

Одинаковое время доступа ко всей памяти для всех процессоров

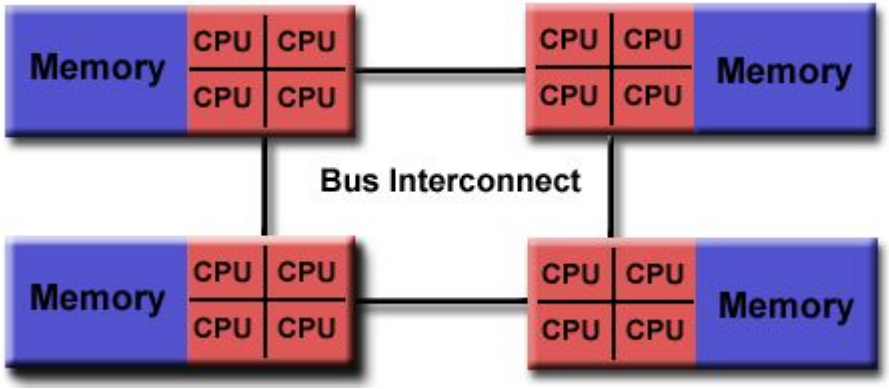


Общая память
Non Uniform Memory Access (3/4)

Несколько однотипных процессоров

Быстрое время доступа к своей памяти

Медленное время доступа к памяти другого процессора



Общая память Достоинства и недостатки (4/4)

Достоинства

- Достаточно простая модель программирования
- Быстрый и одинаковый доступ к любым данным

Недостатки

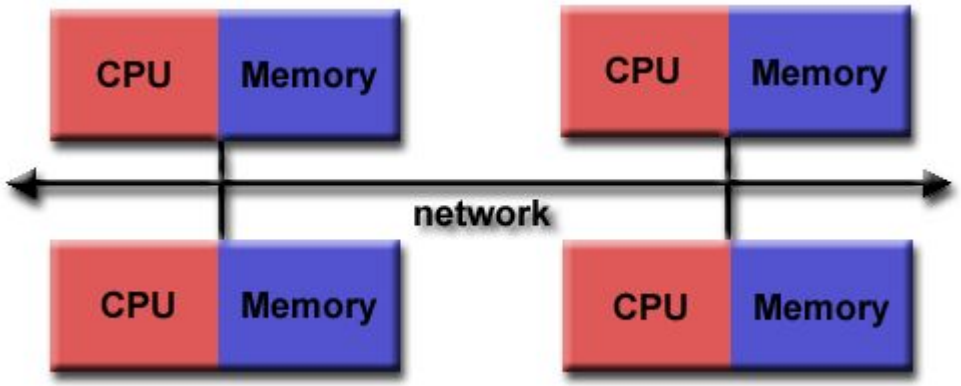
- Необходимо использование примитивов синхронизации для корректной работы с общей памятью
- Слабая масштабируемость (вертикальная)
  - Количество процессоров ограничено
  - Количество памяти ограничено

Архитектура памяти параллельных вычислителей

2. Распределенная память (Distributed Memory)

Распределенная память (1/2)

- Все процессоры имеют локальную память
- Полная изоляция данных между процессорами
- Процессоры работают независимо
- Требуется канал связи между процессорами
- Задача программиста – обеспечить передачу необходимых для вычисления данных к процессору



Распределенная память Достоинства и недостатки (2/2)

Достоинства

- Хорошая масштабируемость (горизонтальная)
  - А также отказоустойчивость
- Быстрый доступ процессора к локальной памяти
  - Пакетная обработка - задача + данные
- Можно использовать широко распространенные недорогие ресурсы
  - Гиперконвергентные среды

Недостатки

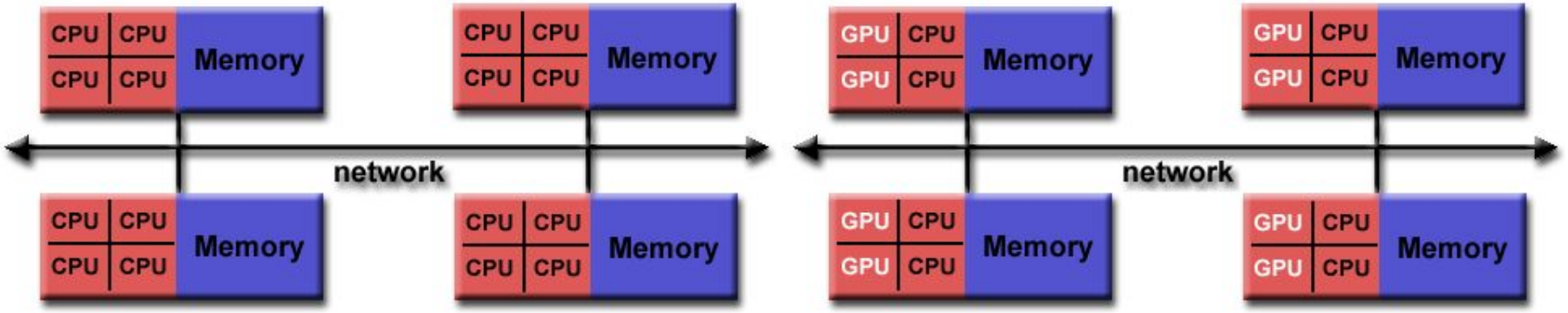
- Более сложная модель программирования
- Сложный переход от вертикального к горизонтальному масштабированию
  - Пример TravelLine: Sheduler, SQL, хранилище
- Доступ к данным на удаленном вычислителе – очень длительная операция

Архитектура памяти параллельных вычислителей

3. Гибридная архитектура (Hybrid Distributed-Shared Memory)

Гибридная архитектура (1/2)

- Современные системы используют оба подхода одновременно
  - Shared memory – ОЗУ + GPU
  - Distributed memory - сеть



Гибридная архитектура Достоинства и недостатки (2/2)

Достоинства

- Те же что и в предыдущих случаях
- Высокая степень масштабируемости

Недостатки

- Те же что и в предыдущих случаях
- Высокая сложность

Модели параллельных вычислений

- Потоки
- Общая память
- Распределенная память / Обмен сообщениями
- Параллельная обработка данных
- Гибридные архитектуры
- Single Program/Instruction Multiple Data (SPMD, SIMD)
- Multiple Program/Instruction Multiple Data (MPMD , MIMD)

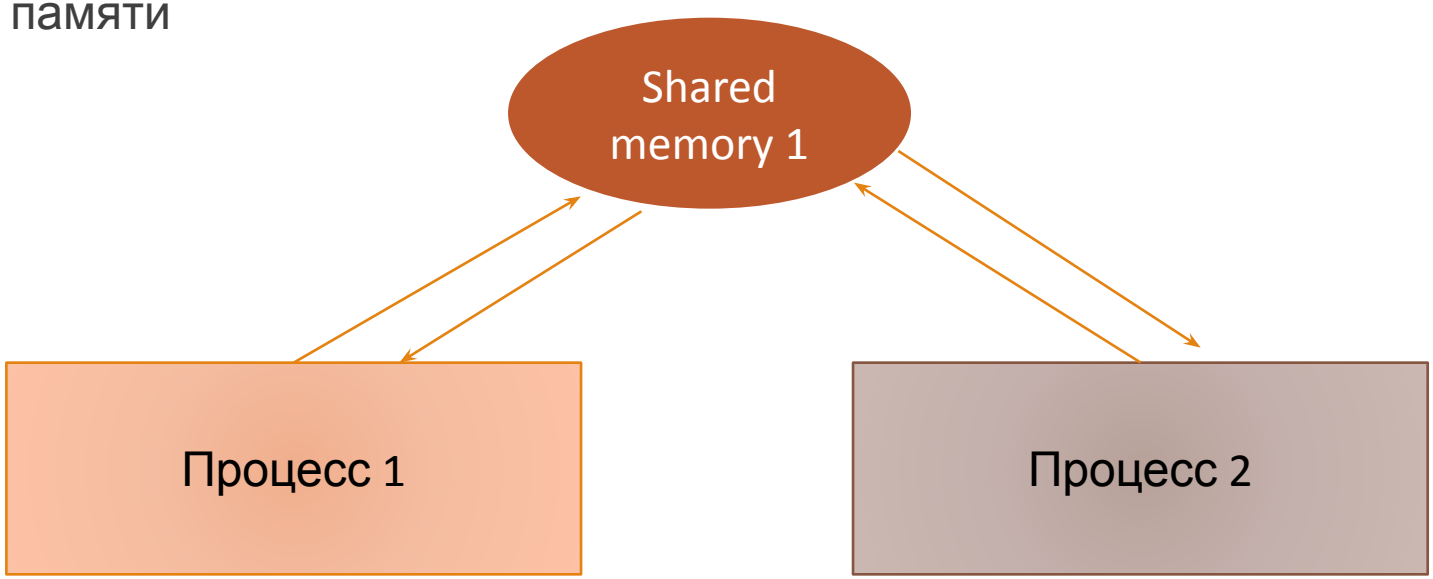
Модели параллельных вычислений

Общая память
(Shared Memory)

Shared memory

Именованная область памяти, которая разделяется между несколькими процессами на одном компьютере.

- Процессы имеют общую память, которую они читают и пишут асинхронно
- Должны использоваться механизмы синхронизации доступа к памяти



Преимущества

- Простое использование
- Быстрый обмен данными между вычислителями
- Нет необходимости реализации коммуникаций между вычислителями
- Наиболее распространенная модель вычислений в Web приложениях
  - TravelLine BackEnd
  - TravelLine ChannelManager

Недостатки

- Сложно не нарушить инкапсуляцию данных отдельных задач
  - Все что можно нарушить будет нарушено
- Требуются реализация коммуникаций между вычислителями

Реализация

Модель может быть реализована как на Shared Memory архитектуре так и на Distributed Memory с использованием дополнительных библиотек или аппаратных реализаций.

- Проект SHMEM (Symmetric Hierarchical Memory access)

Пример с созданием разделяемого блока памяти

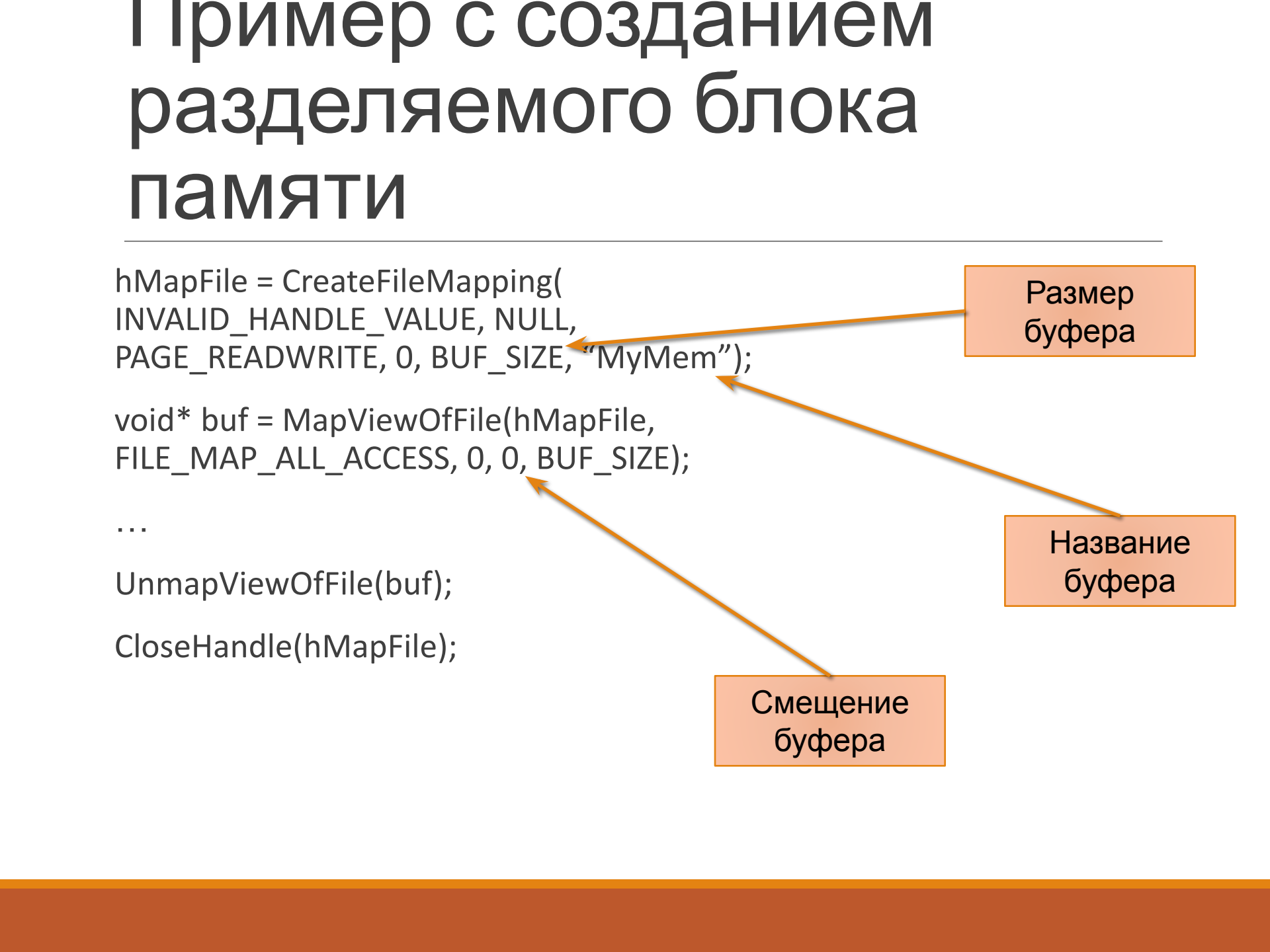
```
hMapFile = CreateFileMapping(
INVALID_HANDLE_VALUE, NULL,
PAGE_READWRITE, 0, BUF_SIZE, “MyMem”);

void* buf = MapViewOfFile(hMapFile,
FILE_MAP_ALL_ACCESS, 0, 0, BUF_SIZE);

…

UnmapViewOfFile(buf);

CloseHandle(hMapFile);
```

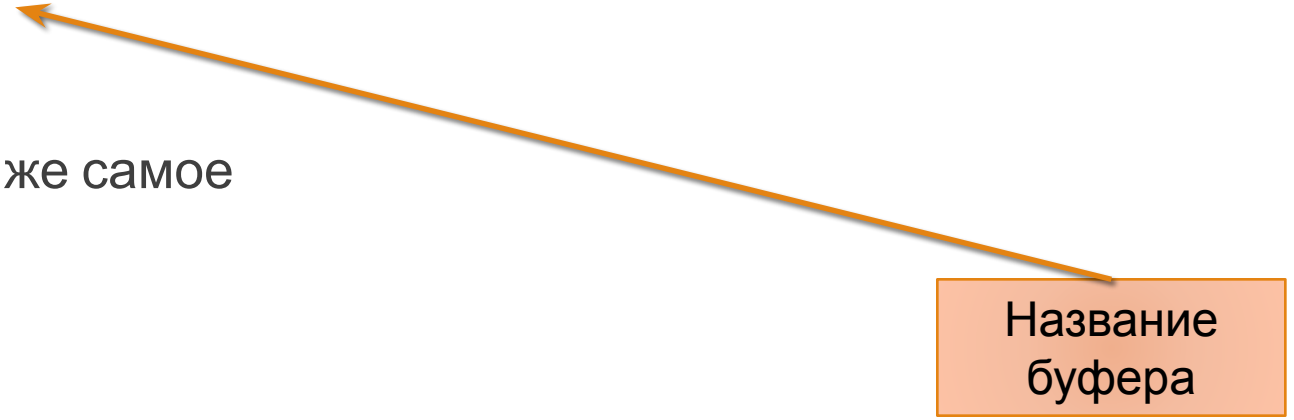


Пример с работой с готовым разделяемым блоком памяти

```
hMapFile = OpenFileMapping( FILE_MAP_ALL_ACCESS,
FALSE, “MyMem”);
```

А дальше все то же самое

…



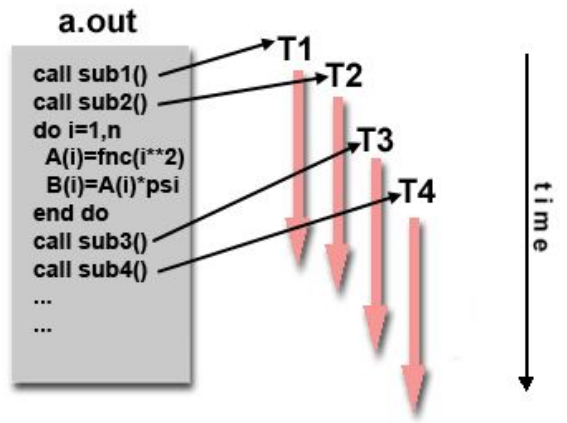
Модели параллельных вычислений

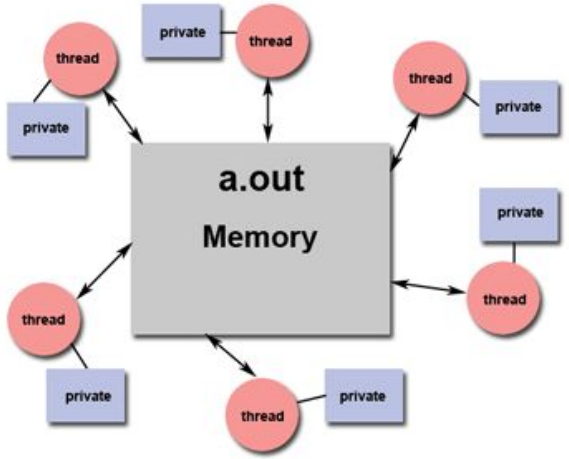
Потоки
(Threads)

Потоки

Разновидность реализации модели с общей памятью

Один тяжелый процесс может иметь несколько легковесных нитей





Реализация

Исторически разработчики аппаратного обеспечения разработали разные стандартны параллельной обработки.

Позже были разработаны два непохожих стандарта:

- POSIX
  - Определены стандартном IEEE POSIX 1003.1c в 1995 году только для языка C
  - Изначально были частью Unix систем
  - Реализованы в библиотеках
  - Явный паралеллизм
- OpenMP
  - Стандарт, разработанный группой крупных компаний
  - Поддержка на уровне компилятора
  - Изначально мультиплатформенный
  - Позволяет гибко менять степень паралеллизма
- Другие реализация
  - Microsoft Windows Threads

Достоинства и недостатки

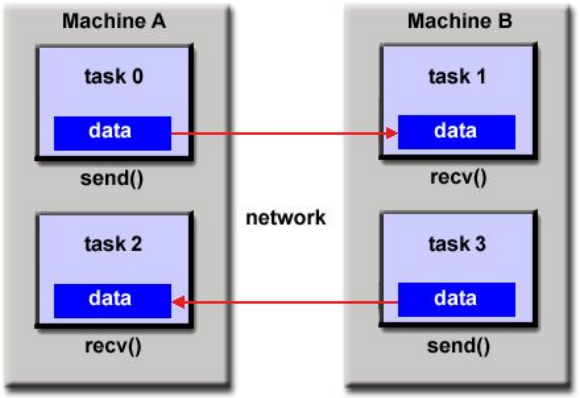
- Наследуют все от модели Shared Memory
- Более легковесны

Модели параллельных вычислений

Распределенная память. Обмен сообщениями.
(Distributed Memory. Messaging)

Распределенная память.

- Разные роли вычислителей
- Требуется канал коммуникации
- Обмен при помощи сообщений
- Как правило есть контроперации send/receive
- Клиент серверная архитектура
- К некоторых случаях размытая граница с системами распределенной памяти



Достоинства и недостатки

- Достоинства
  - Высокая степень инкапсуляции данных
    - Предоставляются через API
  - Высокая степень защищенности данных
    - Географическая удаленность
    - Шифрование канала
- Недостатки
  - Сложность реализации
  - Ненадежный канал
    - Дублирование сообщений
    - Идемпотентные протоколы
  - Медленный канал
    - Асинхронный UI

Обмен сообщениями

1. Оконные сообщения

Оконные сообщения

Серия сообщений WM_USER+I

- Можно передать два числа по 4 байт или 8 байт в сумме

Сообщения WM_COPYDATA

- Можно передать произвольный массив данных

Оконные сообщения

WM_USER+N

#define MY_CUSTOM_MESSAGE (**WM_USER+1**)

#define MY_NEXT_CUSTOM_MESSAGE (**WM_USER+2**)

SendMessage - синхронно

PostMessage – асинхронно

Два параметра

- lParam - контекст
- wParam - контекст

Оконные сообщения

Не WM_USER+N

Допустимо, но не рекомендуется

При желании можно управлять другим приложением, как будто им управляет пользователь

Оконные сообщения

SendMessage - синхронно

PostMessage – асинхронно

Два параметра

- lParam - контекст
- wParam - контекст

Оконные сообщения

WM_COPYDATA

```
If (strlen(cmdLine) != 0)
{
	COPYDATASTRUCT cds;
	cds.cbData = strlen(cmdLine) + 1;
	cds.lpData = cmdLine;
	SendMessage(hWnd, WM_COPYDATA, 0, (LPARAM)&cds);
}
```

Преимущества

- Просто
- Работает «прямо из коробки»

Недостатки

- Работает только под Windows
- Надо искать Handle второго процесса
- Все работает только на одной машине
- Отсылка сообщений может подвисать, если процессор загружен и второй процесс сейчас не выполняется

Обмен сообщениями

2. Named Pipes

Pipes

- “Быстрые файлы”, которые работают по локальной сети
- У пайпа есть имя
- Можно одновременно читать и писать
- Обычно с пайпами работают в потоках

Имя pipe-канала

- Имя удаленного pipe-канала:
  - \\<сервер>\Pipe\<имя_канала>
    <сервер> - IP, DNS, NetBIOS
    <имя_канала> – уникальное имя
- Имя pipe-канала внутри одного компьютера:
  - \\.\Pipe\<имя_канала>

Pipes API

- **CreateNamedPipe** – создать именованный канал
- **ConnectNamedPipe** – ждать подключения, подключиться (синхронно или асинхронно)
- **DisconnectNamedPipe** – отключиться

Pipes API: обмен сообщениями

- **CreateFile** – открыть канал (как файл);
- **ReadFile(Ex)** – читать из файла (файл – абстракция);
- **WriteFile(Ex)** – записать в файл;
- **TransactNamedPipe** – чтение и запись одновременно;
- **PeekNamedPipe** – просматривает данные из буфера пайпа без их извлечения;

Преимущества

- Быстрее файлов
- Работают по локальной сети
- Есть если не везде, то много где (в Linux/MacOS X/Windows)

Недостатки

- Не работают через интернет
- Нужны потоки (в отличие от оконных сообщений)

Обмен сообщениями

3. Mailslots

Mailslots

обеспечивают **«ненадежную»** связь в режиме широковещания;

серверы и **клиенты** ящиков: связь только в одну сторону

Имя ящика:
\\<сервер>\Mailslot\<имя_ящика>
<сервер> - IP, DNS, NetBIOS
< имя_ящика> – уникальное имя

net send

Схема взаимодействия через Mailslots

Если несколько серверных процессов внутри локальной сети создадут мэйлслоты с одинаковым именем, то сообщения, адресованные этому мэйлслоту и посылаемые в домен, будут приниматься всеми создавшими его процессами. Клиентом мэйлслота может быть любой процесс, знающий его имя. Клиент записывает в мэйлслот сообщения для передачи их посредством датаграмм серверу. Один и тот же процесс может быть одновременно клиентом и сервером мэйлслотов

Mailslots API: сервер

- **CreateMailslot** – создать ящик;
- **SetMailslotInfo** – установить настройки;
- **GetMailslotInfo** – получить настройки и статистику;
- **ReadFile** – читать данные (используется дескриптор ящика).

Mailslots API: клиент

- **CreateFile** – открыть ящик
  *\Mailslot\<имя_ящика>\...
  \\<сервер>\Mailslot\<имя_ящика>\...
- **WriteFile** – отправить сообщение; широковещательное – до 424 байт

Преимущества

- ??

Недостатки

- Односторонняя передача данных
- Нет гарантии доставки
- 424 БАЙТА МАКСИМУМ
  - случайности не случайны (424000)
  - To send messages that are larger than 424 bytes between computers, use named pipes or Windows Sockets instead.

Обмен сообщениями

4. Сокеты

Сокеты

Сокет – программный интерфейс сетевого обмена между процессами

- Прямое соединение
- IP
- Порт

Сокеты

Синхронные

Асинхронные

Блокирующие

Неблокирующие

TCP

UDP

Работа с сокетом

Сокет:

- создается;
- настраивается на заданный режим работы;
- применяется для организации обмена;
- ликвидируется.

Сокет характеризуется

- семейство протоколов
- локальный IP-адрес
- удаленный IP-адрес
- номер локального порта
- номер удаленного порта

Оператор формирования

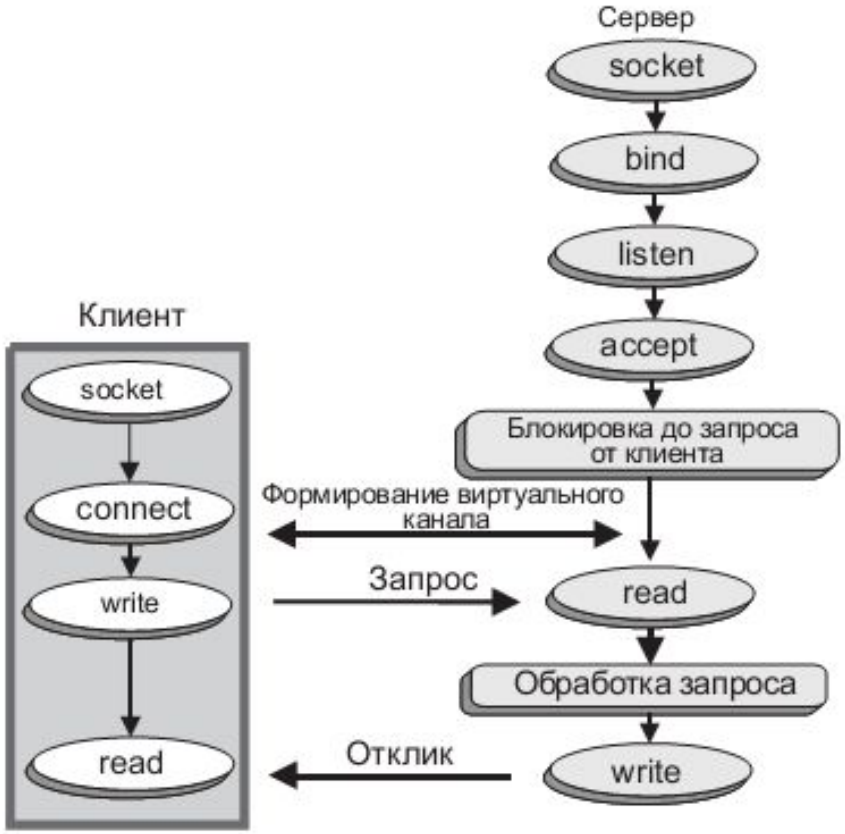
s = socket(INT AF, INT type, INT protocol), где:

AF (address_family) - набор протоколов (Internet, Unix, Appletalk и т. д.);

type - тип коммуникаций (SOCK_STREAM, SOCK_RAW, и SOCK_DGRAM);

protocol - код конкретного протокола из указанного набора (заданного AF) (например, IPPROTO_TCP или IPPROTO_UDP).

Схема взаимодействия



Обмен данными

write(s, buf, len);

read(s, buf, len);

send(s, buf, len, flags);

recv(s, buf, len, flags), где

- s — дескриптор сокета,
- buf — имя массива, подлежащего пересылке (или предназначенного для приема),
- len — длина этого массива
- flags - флаги диагностики и управления передачей данных

Преимущества и недостатки

- Преимущества
  - Работают везде
  - Высокий уровень масштабируемости
- Недостатки
  - Высокая сложность
  - Все проблемы, связанные с сетью

Вопросы?