МИНОБРНАУКИ РОССИИ
федеральное государственное бюджетное образовательное учреждение
высшего образования
«Московский государственный технологический университет «СТАНКИН»
(ФГБОУ ВО «МГТУ «СТАНКИН»)

Презентация к теории курса

Суперкомпьютерные вычисления

Выполнили студенты группы ИДМ-20-01: Абрамов Алексей, Беляков Егор, Крымов Артем, Сержантов Артем

Москва 2021 г.

Области применения многопроцессорных систем

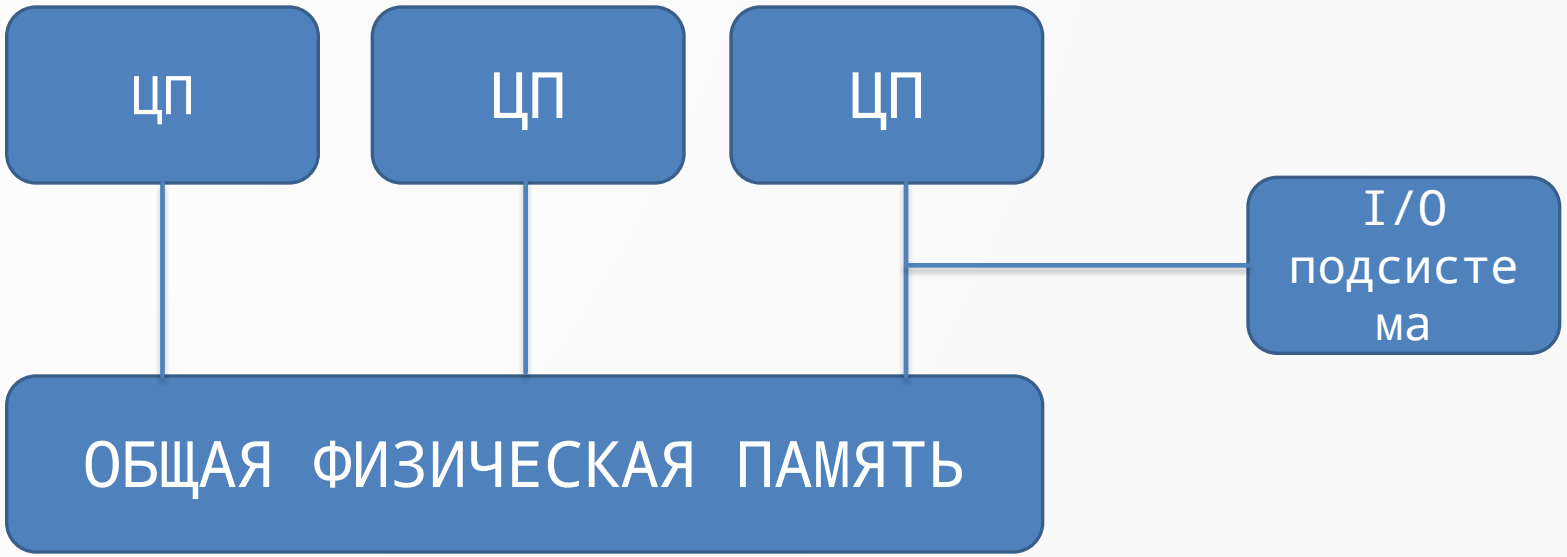
Обработка транзакций в режиме реального времени (OLTP, online transaction processing)

Создание хранилищ данных для организации систем поддержки принятия решений (DSS, Data Mining, Data Warehousing, Decision Support System)

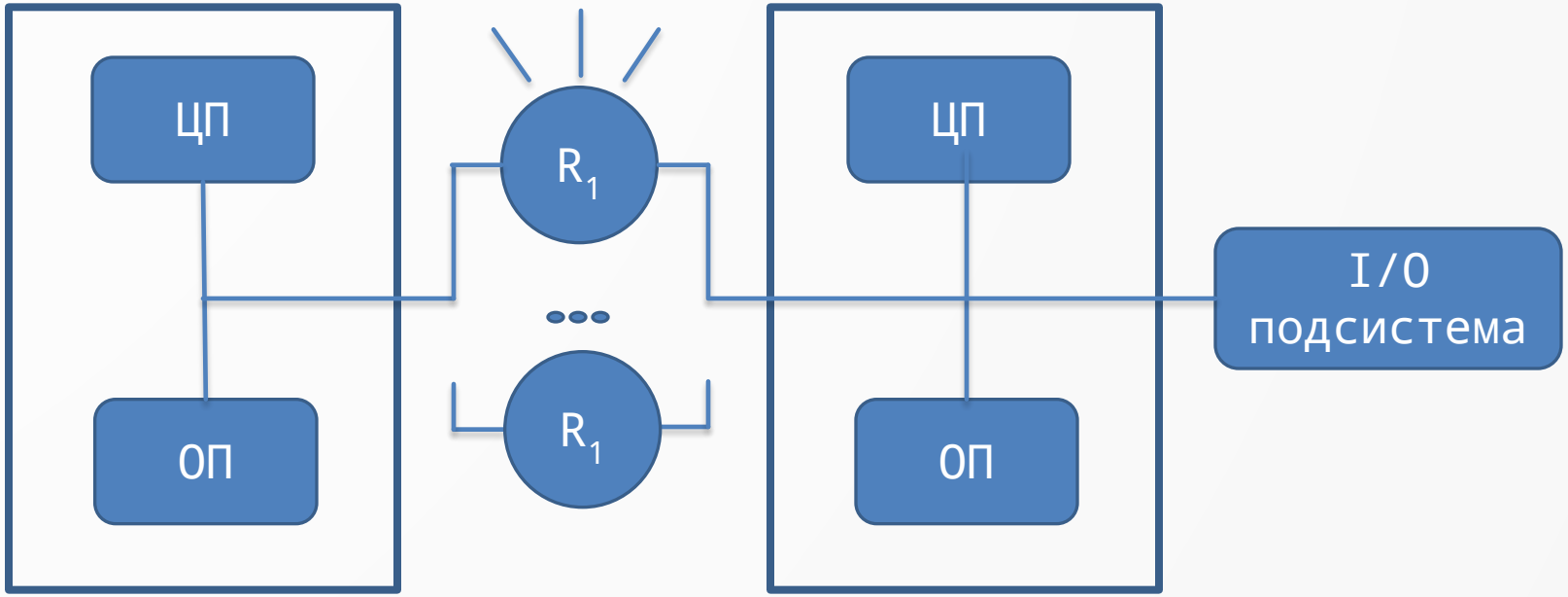
Архитектура многопроцессорных вычислительных систем

	Одиночный поток команд	Множество потоков команд
Одиночный поток данных	SISD	MISD
Множество потоков данных	SIMD	MIMD

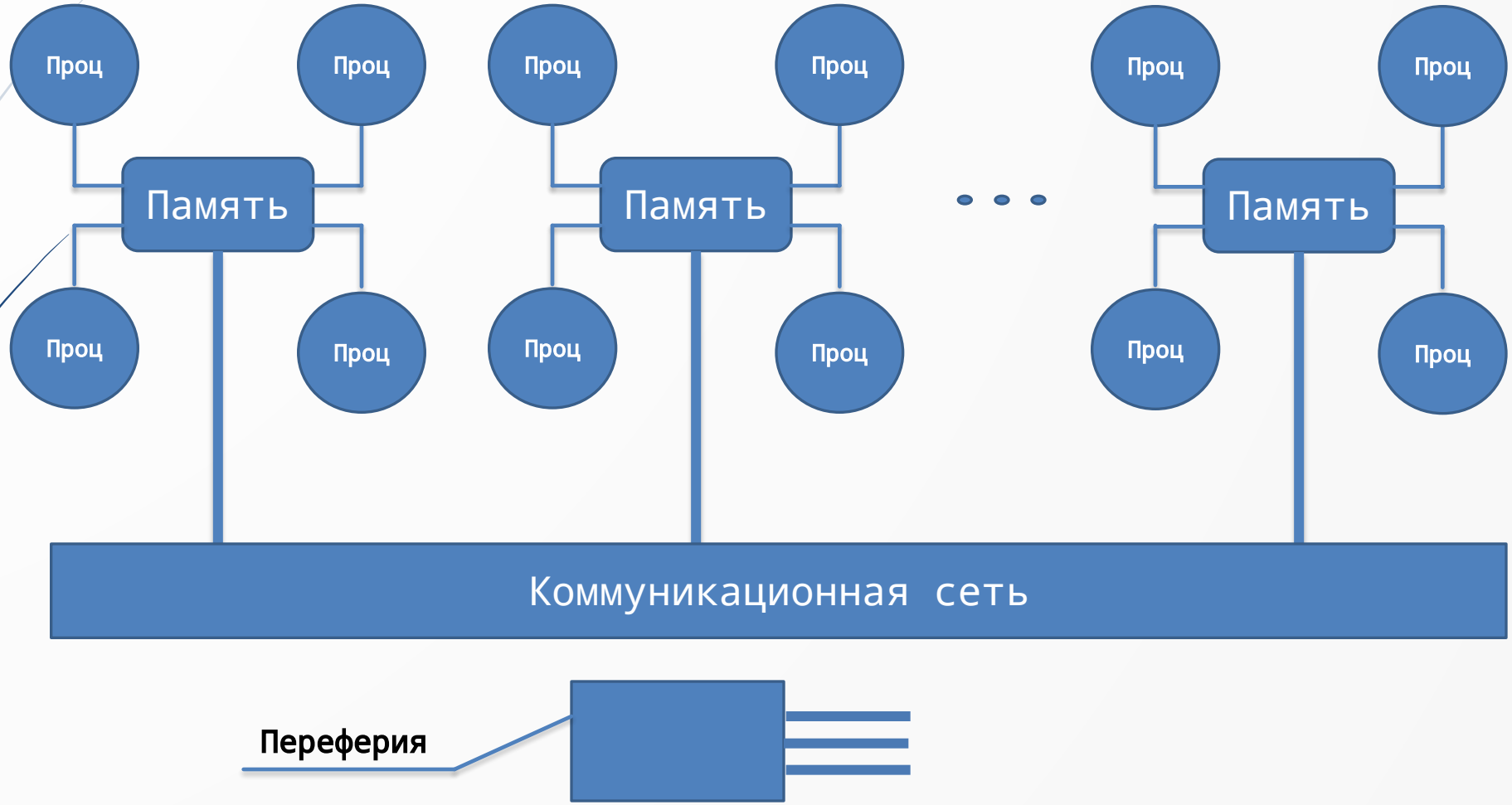
Многопроцессорные системы с общей памятью



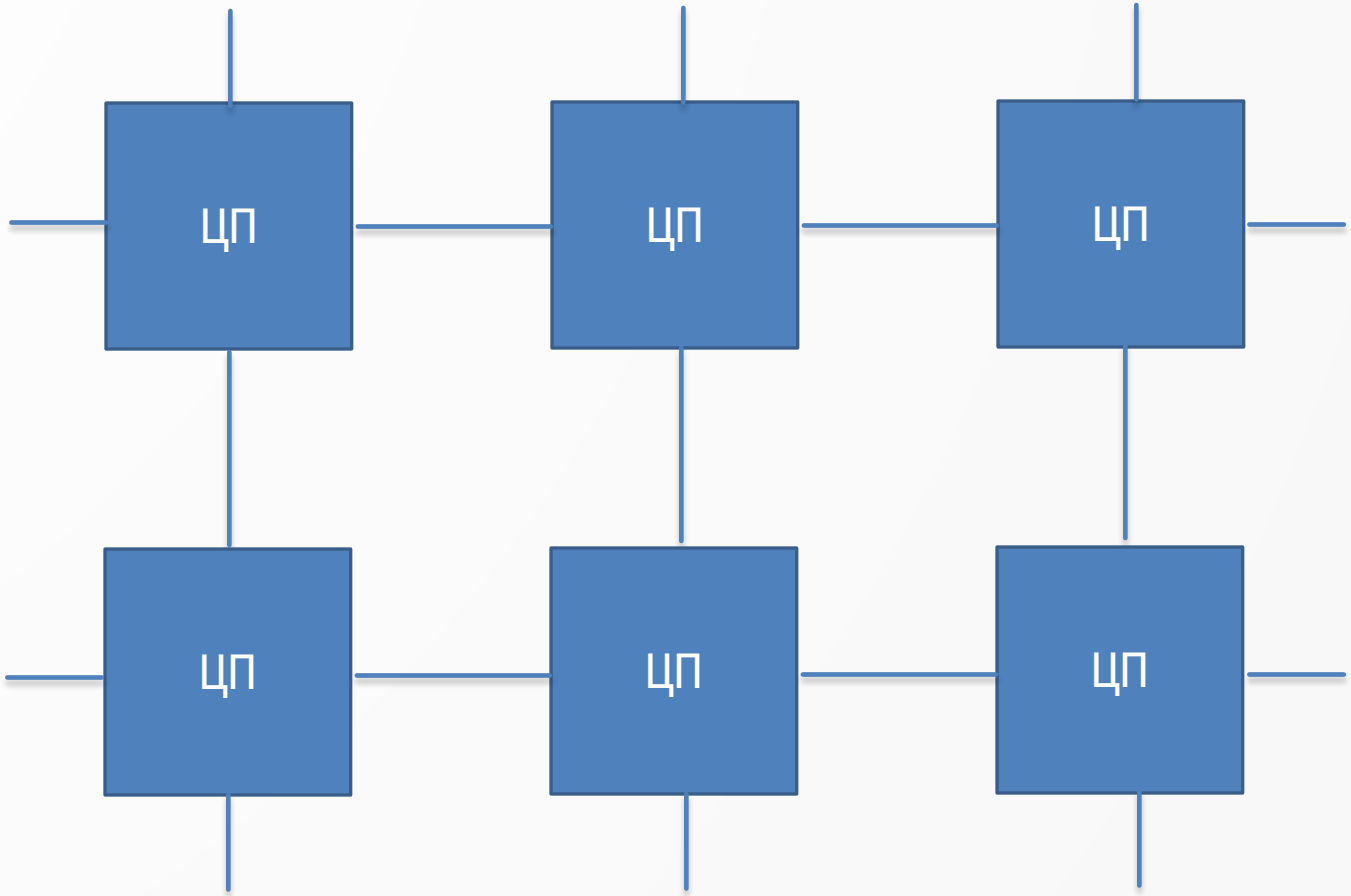
Многопроцессорные системы с раздельной памятью



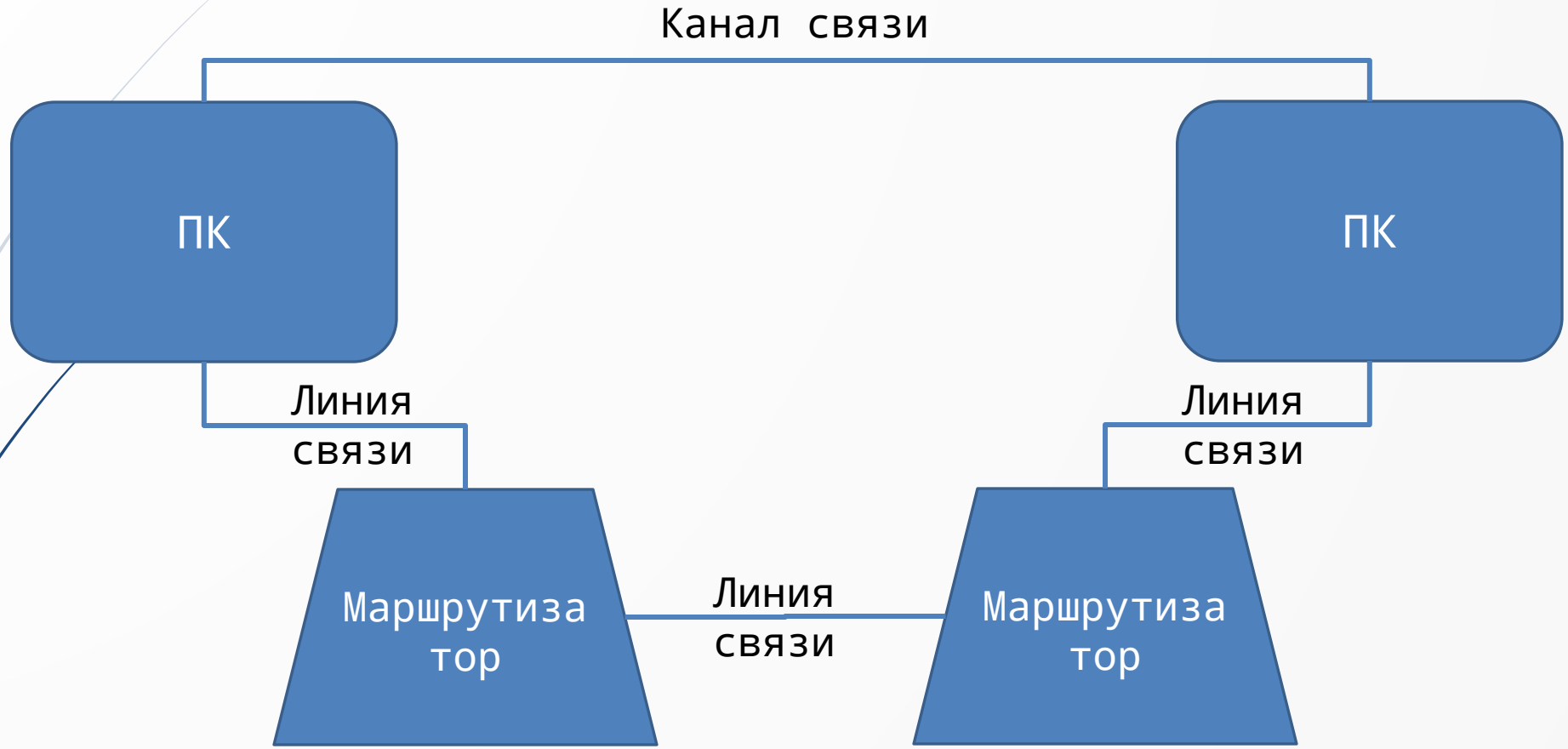
Многопроцессорные системы гибридного типа



Кластерные вычислительные системы



Компьютерные сети



Коммуникационные среды

В самом общем смысле архитектуру компьютера можно определить как способ соединения компьютеров между собой, с памятью и с внешними устройствами. Реализация этого соединения может идти различными путями. Конкретная реализация такого рода соединений называется коммуникационной средой компьютера.

Транспьютеры

Транспьютер - это сверхбольшая интегральная микросхема, заключающая в себе:

- центральный процессор
- блок операций с плавающей запятой
- статическое оперативное запоминающее устройство
- интерфейс с внешней памятью
- несколько каналов связи

Специализированные языки параллельного программирования

Язык Occam

- Язык программирования высокого уровня, альтернатива ассемблеру для транспьютеров
- Два типа каналов: ввод и вывод
- Пять видов конструкторов процессоров (SEQ, PAR, ALT, IF, WHILE)

Язык HFP

- Является расширением языка Fortran
- Эффективен для реализации SIMD и MIMD архитектур
- Новые операторы (например, FORALL)
- Возможность создавать PURE процедуры (без побочных эффектов)
- Дополнительные библиотечные процедуры, включающие в себя разброс данных и операции сортировки

Библиотеки и системы программирования для высокоуровневых языков

PARIX OS

- Многозадачная операционная система
- Расширение операционной системы UNIX
- Фиксированная регулярная топология
- Исполняемые программы реализуют SPMD модель

MPI

- Переносимый стандарт передачи сообщений
- Поддерживает языки C, C++ и Fortran
- Ориентирован на системы с распределенной памятью

OpenMP

- Открытый стандарт для реализации параллельных вычислений
- Поддерживает языки C, C++ и Fortran
- Ориентирован на системы с общей памятью

Принципы построения параллельных алгоритмов. Основные методы построения параллельных программ

Основные типы параллелизма, используемые для построения параллельных алгоритмов:

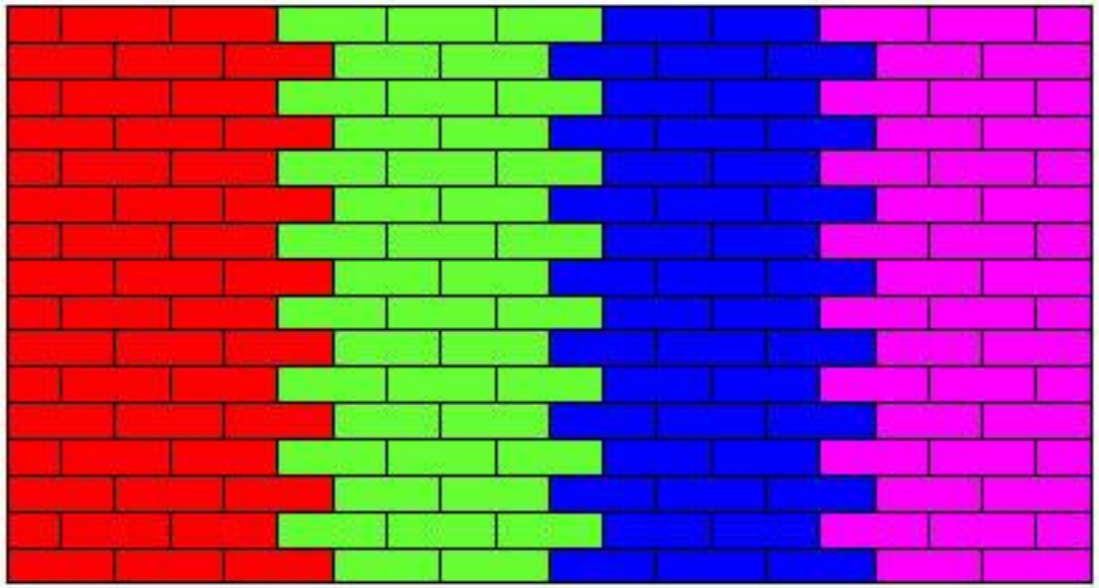
- Алгоритмический
- Геометрический
- Конвейерный
- «Коллективное решение»

Алгоритмический параллелизм

Алгоритмическим параллелизмом называется параллелизм, который выявляется путем выделения в данном алгоритме тех фрагментов, которые могут выполняться параллельно

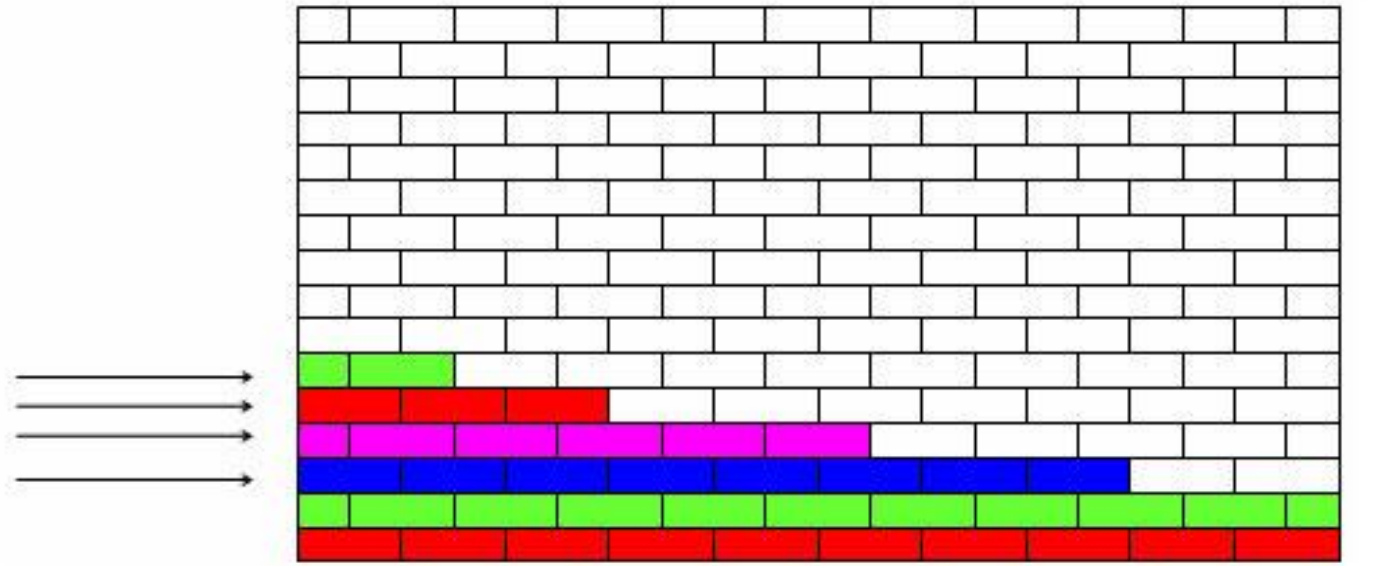
Геометрический параллелизм

Геометрический параллелизм допускает разбиение данных на части по числу процессоров, при котором обработку каждой части осуществляет соответствующий процессор.



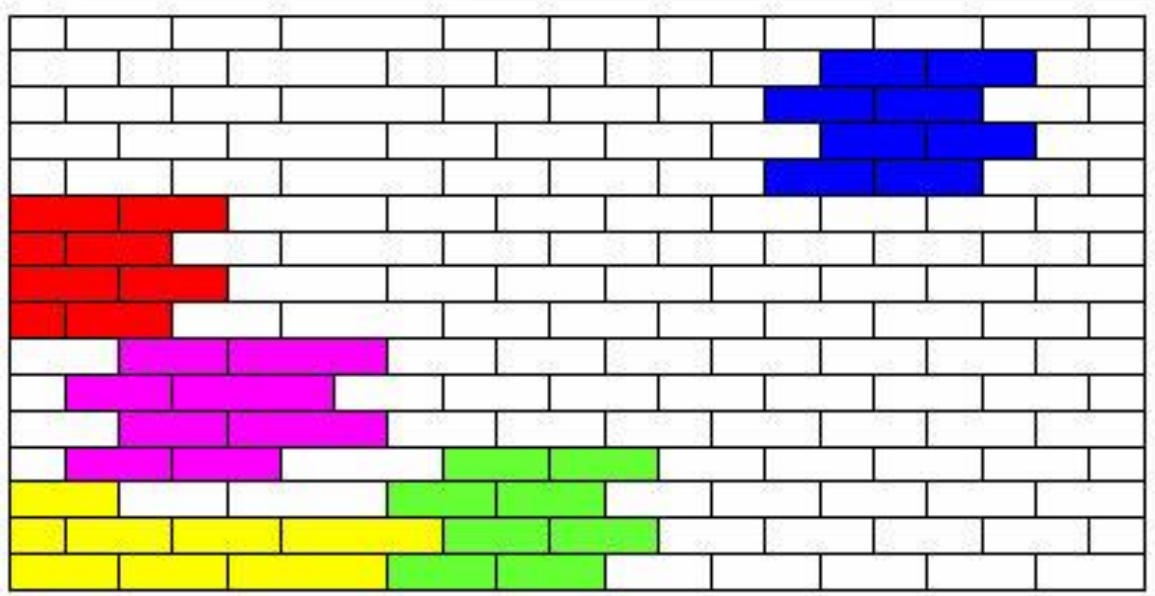
Конвейерный параллелизм

Первый каменщик (процессор) укладывает первый ряд кирпичей, второй каменщик - второй ряд, и т.д. Требуется время на передачу информации о свойствах кирпича и о том, что ряд ниже уложен. Неизбежны потери времени в начале и в конце выполнения задачи - например, второй каменщик не может начать работу прежде, чем первый каменщик уложит хотя бы один кирпич.



Параллелизм типа «коллективное решение»

Метод предполагает, что неизвестно, с какой скоростью какие процессоры работают. Существует один выделенный процессор, который выполняет управляющие функции, он раздает задачи и принимает результаты, а остальные процессоры выполняют обрабатывающие функции независимо друг от друга.



Статическая и динамическая балансировка загрузки процессоров

- ***Статическая балансировка*** выполняется до начала выполнения параллельного приложения.

- Алгоритм ***динамической балансировки*** определяет с какого вычислительного узла и на какой следует перенести задачу во время работы системы.

Проблема тупиков

Тупик - ситуация, в которой один или несколько процессов ожидают какого-либо события, которое никогда не произойдет.

Необходимые условия тупика:

- Условие взаимоисключения
- Условие ожидания ресурсов
- Условие неперераспределяемости
- Условие кругового ожидания

Недетерминированность параллельных программ

Недетерминированность процессов происходит от того, что они обмениваются данными в непредсказуемые моменты времени.

Параллельная программа представляется системой взаимодействующих процессов, *порядок* выполнения которых *жестко не фиксирован*, что является причиной недетерминизма вычислений

Эффективность, ускорение параллельных программ.

Ускорением параллельного алгоритма (Sp), называют отношение времени выполнения алгоритма на одном процессоре ко времени выполнения алгоритма на системе из p процессоров.

Эффективностью параллельного алгоритма (Ep) называют отношение его ускорения к числу процессоров, на котором это ускорение получено.

Степенью параллелизма алгоритма называют число действий алгоритма, которые могут выполняться одновременно, каждое на своем процессоре.

Оценка времени выполнения алгоритма

Степенью параллелизма алгоритма называют число действий алгоритма, которые могут выполняться одновременно, каждое на своем процессоре.

Каждое из сложений $c_i = a_i + b_i$, i=1, ..., n не зависит от других и может выполняться одновременно с ними, таким образом, степень параллелизма данного алгоритма равна n.

От числа процессоров зависит только время выполнения реальной программы при ее запуске на этих процессорах. Можно сделать вывод о том, что ускорить выполнение программы более чем в число раз, определяемое степенью параллелизма ее алгоритма, нельзя.

Внутренний параллелизм

- Внутренние свойства алгоритма, не зависящие от вида и особенностей вычислительных систем.
- На разных этапах выполнения алгоритма степень параллелизма может быть различной.
- Важно, чтобы высокой степенью внутреннего параллелизма обладали этапы, на выполнение которых тратится основное время.

Закон Амдаля

Закон Амдаля характеризует одну из самых серьезных проблем в области параллельного программирования - алгоритмов без определенной доли последовательных команд практически не существует.

$$S_p \leq \frac{1}{f + (1 - f)/p} \leq S^* = \frac{1}{f}$$

Параллельная программа как ансамбль взаимодействующих последовательных процессов

Поскольку параллельная программа – это множество взаимодействующих *последовательных процессов*, разработка параллельного алгоритма сводится к разработке набора *последовательных алгоритмов*, каждый из которых выполняется на своем процессоре.

Основные методы синхронизации

Синхронизация с помощью передачи сообщений

- Взаимодействие при помощи передачи сообщений. При этом обмен сообщениями может происходить асинхронно либо с использованием метода, при котором отправитель блокирован до тех пор, пока его сообщение не будет доставлено.

Синхронизация с помощью общих переменных

- Взаимодействие через разделяемую память - вид параллельного программирования требующий формы захвата управления для координации потоков между собой

Синхронизация с помощью передачи сообщений

Модель акторов

- Модель акторов – это математическая модель параллельных вычислений. Была предложена в 70х годах Карлом Хьюитом.
- Актор – примитивная единица, получающая сообщения и делающая примитивные вычисления, основанные на этих сообщениях

Действия акторов

- Создать новый актор
- Отправить сообщение другим акторам
- Поменять внутреннее состояние

Виды каналов передачи данных

Типовые топологии схем коммуникации процессоров

- Полный граф - система, в которой между любой парой процессоров существует прямая линия связи
- Линейка - система, в которой каждый процессор имеет линии связи толькос двумя соседними процессорами
- Кольцо - данная топология получается из линейки процессоров соединением первого ипоследнего процессоров линейки
- Звезда - система, в которой все процессоры имеют линии связи с некоторымуправляющим процессором
- Решетка - система, в которой граф линий связи образует прямоугольную сетку
- Гиперкуб - данная топология представляет частный случай структуры решетки, когда по каждой размерности сетки имеется только два процессора

Синхронный и асинхронный обмен сообщениями

Синхронный обмен сообщениями

- При синхронном обмене сообщениями отправитель и получатель дожидаются друг друга для передачи каждого сообщения, и операция отправки считается завершенной только после того, как получатель закончит прием сообщения.

Асинхронный обмен сообщениями

- При асинхронном обмене сообщениями не происходит никакой координации между отправителем и получателем сообщения.

Защита разделяемых данных

Для организации разделения ресурсов между несколькими потоками необходимо иметь возможность:

- определения доступности запрашиваемых ресурсов;
- выделения свободного ресурса одному из потоков, запросивших ресурс для использования;
- блокировки потоков, выдавших запросы на ресурсы, занятые другими потоками.

Главное требование к механизмам разделения ресурсов является гарантированное обеспечение использования каждого разделяемого ресурса только одним потоком от момента выделения ресурса этому потоку до момента освобождения ресурса.

Семафор. Реализация семафора с помощью традиционных операций.

Семафоры были введены в эксплуатацию Эдсжером Дейкстрой в 1965 году и с тех пор являются базовым средством управления доступом к критическим ресурсам. Алгоритм использования семафоров выглядит так:

Для реализации операций над семафором необходимо наличие примитивов, отвечающих за:

- Задержку исполнения процесса выполнившего операцию
- возобновление исполнения приостановленного процесса

Критическая секция. Монитор, Очереди заданий.

- Критическая секция – объект синхронизации потоков, позволяющий предотвратить одновременное выполнение некоторого набора операций несколькими потоками. При нахождении в критической секции двух или более потоков возникает состояние «гонки»
- Мониторинг системы – это ежедневное рутинное занятие, и этот документ предоставляет систематическую пошаговую процедуру для мониторинга сервера. Он дает обзор технических аспектов и концепций для упреждающего мониторинга системы.
- Очередь заданий содержит упорядоченный список заданий, ожидающих обработки подсистемой. Перед активизацией в подсистеме переданное на выполнение пакетное задание попадает в очередь заданий. Задание остается в очереди, пока не будет выполнено несколько условий.

Описание поля процессоров

Структура микрокоманды

	Структура микрокоманды	Таблица 1							
Поле	A	B	MA	MB	MEM	SRC	SH	N	ALU
Бит	4	4	2	2	3	3	4	4	4
Def	0	0	0	0	0	1	0	0	6
Поле	CCX	F	DST	WM	JFI	CC	CHA	CONST	
Бит	2	1	3	2	3	3	4	16	
Def	0	0	0	0	0	0	7	0	

Поле MEM

Поле MEM	Значение	Поле MEM	Значение
0	NOP	4	Чтение байта
1	NOP	5	Чтение слова
2	NOP	6	Запись байта
3	NOP	7	Запись слова

Соответствие между адресами РЗУ и регистрами микропроцессора

Поле A/B	Значение	Поле A/B	Значение
0	AX	8	CS
1	CX	9	SS
2	DX	A	DS
3	BX	B	ES
4	SP	C	IP
5	BP	D	PSW
6	SI	E	RGK
7	DI	F	RW

Значения полей MA и MB

Поле MA/MB	0	1	2	3
Источник адреса	Поле A/B МК	Reg1	Reg2	R/m

Описание поля процессоров

Поле SRC

Поле SRC	0	1	2	3	4	5	6	7
Операнд R	0000	RGA	RGA	RGA	RGA*2	CONST	CONST	CONST
Операнд S	0000	RGB	RGQ	RGR	RGB	RGB	RGR	RGQ

Поле ALU

N	Z	V	C	N	Z	V	C											
0	На всех выходах «0»	0	1	0	0	8	Умножение на 2 бита	+	+	+	+							
1	S	—	R	—	1	+	C0	+	+	+	+	9	R	&	S	+	+	0 0
2	R	—	S	—	1	+	C0	+	+	+	+	A	R	&	S	+	+	0 0
3	R	+	S	+ C0	+	+	+	+	B	R	&	S	+	+	0	0		
4	S	+ C0	+	+	+	+	C	R	V	S	+	+	0	0				
5	S	+ C0	+	+	+	+	D	R	V	S	+	+	0	0				
6	R	+ C0	+	+	+	+	E	R	Å	S	+	+	0	0				
7	R	+ C0	+	+	+	+	F	R	Å	S	+	+	0	0				

Поле SH

Поле SH	Операция
0	Без сдвига
1	АС АЛУ вправо
2	ЛС АЛУ вправо
3	АС АЛУ, RGQ вправо
4	ЛС АЛУ, RGQ вправо
5	ЛС RGQ вправо
6	RGQÜALU
8	ЛС АЛУ влево
A	ЛС АЛУ, RGQ влево
E	Расширение знака

Поле CCX

Поле CCX	0	1	2	3
C0	0	1	C	C

Описание поля процессоров

Поле DST

Поле DST	0	1	2	3	4
Источник	Без записи	RGR	RGRL	RGRH	SDA
Приемник	Без записи	РЗУ	РЗУН	РЗУL	РЗУ

Поле WM

Поле WM	0	1	2	3
Источник	Без записи	SDA	SDA	RGB
Приемник	Без записи	RGW	ARAM	ARAM

Поле CC

Поле CC	Вид перехода	Условие перехода
0	JP, JNP*	P=1
1	JZ, JNZ*	Z=1
2	JS, JNS*	N=1
3	JO, JNO*	V=1
4	JC, JNC*	C=1
5	JL, JNL*	N Â V=1
6	JLE, JNLE*	Z Ú (N Â V)=1
7	JBE, JNBE*	CÚZ=1

Поле CHA

Поле CHA	Мнемоника	X=0	X=1	RACT		
Y	Стек	Y	Стек			
0	JZ	0	Очистка	0	Очистка	Хранение
1	CJS	СМК	Хранение	CONST	Загрузка	Хранение
2	JMAP	PA	Хранение	PA	Хранение	Хранение
3	CJP	СМК	Хранение	CONST	Хранение	Хранение
4	RPCT	СМК	Хранение	CONST	Хранение	Декремент
5	CRTN	СМК	Хранение	Стек	Выгрузка	Хранение
6	LDCT	СМК	Хранение	СМК	Хранение	Загрузка
7	CONT	СМК	Хранение	СМК	Хранение	Хранение
8-F	Резерв	—	—	—	—	—

Виртуальные и реальные процессоры

Виртуальным процессором называется процесс сервера баз данных. Виртуальный процессор можно сравнить с операционной системой. Поток по отношению к нему выступает как процесс, подобно тому, как сам виртуальный процессор является процессом с точки зрения операционной системы.

Виртуальные процессоры (ВП) являются специализированными - они подразделяются на классы в соответствии с типом потоков, для выполнения которых они предназначены. Примеры классов ВП:

- CPU - Потоки обслуживания клиентов, реализуют оптимизацию и логику выполнения запросов. К этому классу относятся и некоторые системные потоки.
- AIO - Операции асинхронного обмена с диском.
- ADM - Административные функции, например, системный таймер.
- TLI - Контроль сетевого взаимодействия посредством интерфейса TLI (TransportLayerInterface).

Параллельное и конкурентное выполнение процессов

Конкурентность

- Конкурентность – это свойство систем , допускающее одновременное выполнение нескольких вычислительных процессов, которые могут взаимодействовать друг с другом

Параллельность

- Параллельные вычисления используют более одного вычислительного ядра, поскольку все управляющие потоки работают одновременно и занимают весь рабочий цикл ядра на время исполнения – именно поэтому параллельное вычисление невозможно на одноядерном компьютере.

Физические и виртуальные каналы

Виртуальные каналы – это устойчивые пути следования трафика, создаваемые в сети с коммутацией пакетов. Они представляют собой соединение между двумя конечными станциями сети и являются двунаправленными.

Существуют три вида виртуальных каналов:

- постоянные виртуальные каналы (PVC)
- коммутируемые виртуальные каналы (SVC)
- интеллектуальные постоянные виртуальные каналы (SPVC)

Физические и виртуальные топологии

В MPI топология представляет собой механизм сопоставления процессам, принадлежащим группе, альтернативных по отношению к обычной схем нумерации. Топологии обменов сообщениями в MPI являются виртуальными, они не связаны с физической топологией коммуникационной сети параллельной вычислительной системы. Использование коммуникаторов и топологий отличает MPI от большинства других систем передачи сообщений. Топологией в данном случае называют структуру соединений - линий и узлов сети без учета характеристик самих этих узлов. Узлами здесь являются процессы, соединениями - каналы обмена сообщениями, а сетью мы, фактически, называем все процессы, входящие в состав параллельной программы.

Режимы передачи данных

Режим передачи (transmissionmode) определяет возможные направления передачи сигналов между узлами сети. Существуют три режима передачи (режима использования канала):

- симплексный или односторонний (simplexmode),
- полудуплексный (half-duplexmode) и
- дуплексный (full-duplexmode).

Измерение времени

На время выполнения программы влияют следующие факторы:

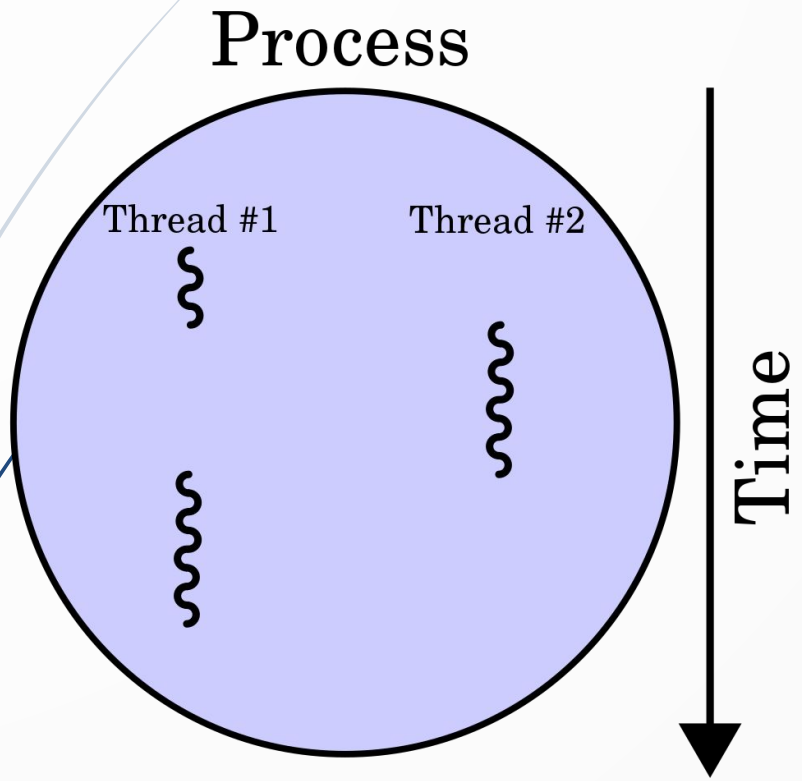
1. Ввод исходной информации в программу.
2. Качество скомпилированного кода исполняемой программы.
3. Машинные инструкции (естественные и ускоряющие), используемые для выполнения программы.
4. Временная сложность алгоритма соответствующей программы.

Отладка программы

Отладка – этап разработки компьютерной программы, на котором обнаруживают, локализуют и устраняют ошибки. Чтобы понять, где возникла ошибка, приходится:

- узнавать текущие значения переменных;
- выяснять, по какому пути выполнялась программа.

Выполнение процессов на общей памяти (треды)



Поток выполнения / тред – наименьшая единица обработки, исполнение которой может быть назначено ядром операционной системы. Реализация потоков выполнения и процессов в разных операционных системах отличается друг от друга, но в большинстве случаев поток выполнения находится внутри процесса.

Использование семафоров для синхронизации тредов

Семафоры традиционно использовались для синхронизации процессов, обращающихся к разделяемым данным. Каждый процесс должен исключать для всех других процессов возможность одновременно с ним обращаться к этим данным (взаимоисключение). Когда процесс обращается к разделяемым данным, говорят, что он находится в своем критическом участке.

Примеры математических моделей применение которых требуют суперкомпьютерных вычислений

Кратко разработку и программную реализацию моделирования задач механики сплошной среды можно сформулировать следующим образом. Сначала алгоритм разделяется на базовые операции. Затем, исходя из зависимости по данным между операциями, на первом уровне реализуется традиционное MPI распараллеливание. Далее, работая с операциями в отдельности, выполняется адаптация к потоковой обработке и различным вычислительным архитектурам, оптимизация структур данных и доступа к памяти.

Кинетически согласованные разностные схемы

Многие из проблем, связанных с использованием многопроцессорных вычислительных систем для решения задач газовой динамики, могут быть решены с помощью применения кинетически согласованных схем, опирающихся на представления статистической механики и механики сплошных сред.

Компиляция и выполнение параллельных программ под управлением PARIX

Для компиляции программных модулей используется команда:

```
px ancc <имя файла.c> -c -qcp1uscmt –O3
```

Для сборки программы используется команда:

```
px ancc <имя файла1>.o ... <имя файла n>.o [опции] –o
   <имя создаваемого выполняемого файла программы>
```

Уровни оптимизации выполняемого программного кода

Решение проблемы быстрой и недорогой разработки эффективного программного обеспечения для параллельных архитектур невозможно без оптимизирующих распараллеливающих компиляторов. Распараллеливающий компилятор — частный случай оптимизирующего, который условно можно разделить на блок синтаксического разбора текста, блок оптимизации и генератор кода.

Заголовочные файлы и библиотеки передачи данных, порождения процессов и доступа к семафорам

Send, Recv – функции синхронного обмена данными через каналы виртуальных топологий
AInit, ASend, ARecv, ASync, AInfo, AExit – функции асинхронного обмена данными через каналы виртуальных топологий
Select, CondSelect, SelectList, CondSelectList, ReceiveOption, ReceiveOption_B, TimeAfterOption, TimeAfterOption – Селективное ожидание ввода данных или таймаута
TimeNow, TimeWait, TimeAfter – доступ к процессорному таймеру
CreateSem, InitSem, DestroySem, Wait, TestWait, Signal - управление семафорами

Запуск и прекращение выполнения программы. Утилита nrm. Сервер epxd

Для запуска PARIX-приложения используется команда run.

Px nrm - утилита управления ресурсами многопроцессорной системы. Информацию о имеющихся в системе разделах можно получить с помощью команды px nrm –pp.

EPX - специализированный API-интерфейс используемый в качестве средства распараллеливания который образует «среду» Embedded PARIX (Parallel Extension to Unix - EPX). Благодаря этому API-интерфейсу можно создавать виртуальные каналы между пользовательскими потоками. Каждый узел ввода-вывода работает под управлением AIX и Parsytec EPX.

Анализ результатов выполнения параллельной программы. Распределение и освобождение разделов

Команда run занимает указанный раздел, так что он (вместе с перекрывающимися разделами) становится недоступен для запуска других приложений PARIX. По окончании PARIX-приложения, run автоматически освобождает занятый раздел, так что он становится доступен для других PARIX- приложений.

Заголовочные файлы и библиотеке MPI. Версии MPI: lam, mpich. Версии Mpi для Unix. Версия Mpi для Windows

LAM - Реализация MPI и среда разработки MPI-программ для гетерогенных кластеров из UNIX-машин, разработана в Ohio Supercomputer Center, теперь эта реализация перешла в ведение Laboratory for Scientific Computing (университет Notre-Damme). Последняя версия - 6.1.

MPICH - это высокопроизводительная и широко переносимая реализация стандарта интерфейса передачи сообщений (MPI) (MPI-1, MPI-2 и MPI-3). Работает почти на всех UNIX-системах и Windows, разработанная в Argonne National Laboratory. Поддерживаются кластеры на базе SMP-узлов.

Мониторинг выполнения параллельной программы

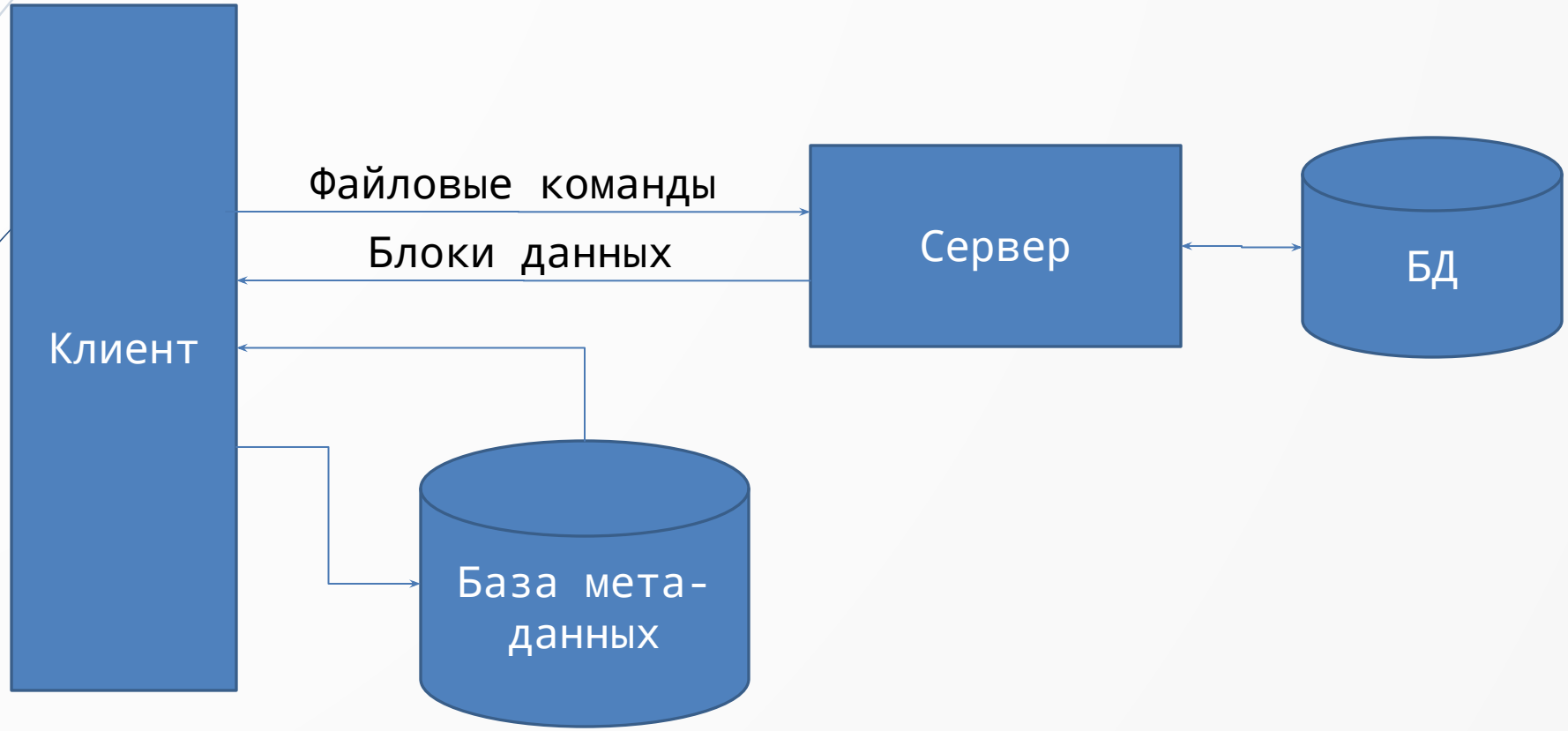
Обычно для целей трассировки в исследуемую программу встраиваются "профилировочные" вызовы, которые фиксируют наступление определенных событий или продолжительность интервалов, и фиксируют эту информацию в журнале трассировки, передают ее онлайн-анализатору или просто модифицируют собираемую статистику.

Некоторые средства Unix (rsh, ssh, bash, ping, scp) их использование для подготовки, выполнения и отладки параллельных программ

1. rsh – удалённый командный интерпретатор.
2. SSH - (Secure Shell) - это протокол удаленного управления компьютером с операционной системой Linux.
3. Bash - это самый популярный интерпретатор команд, который используется в большинстве дистрибутивов Linux.
4. Утилита ping - это очень простой инструмент для диагностики сети.
5. Команда scp - это утилита, которая работает по протоколу SSH, а значит, все что вам нужно для передачи файла на компьютер

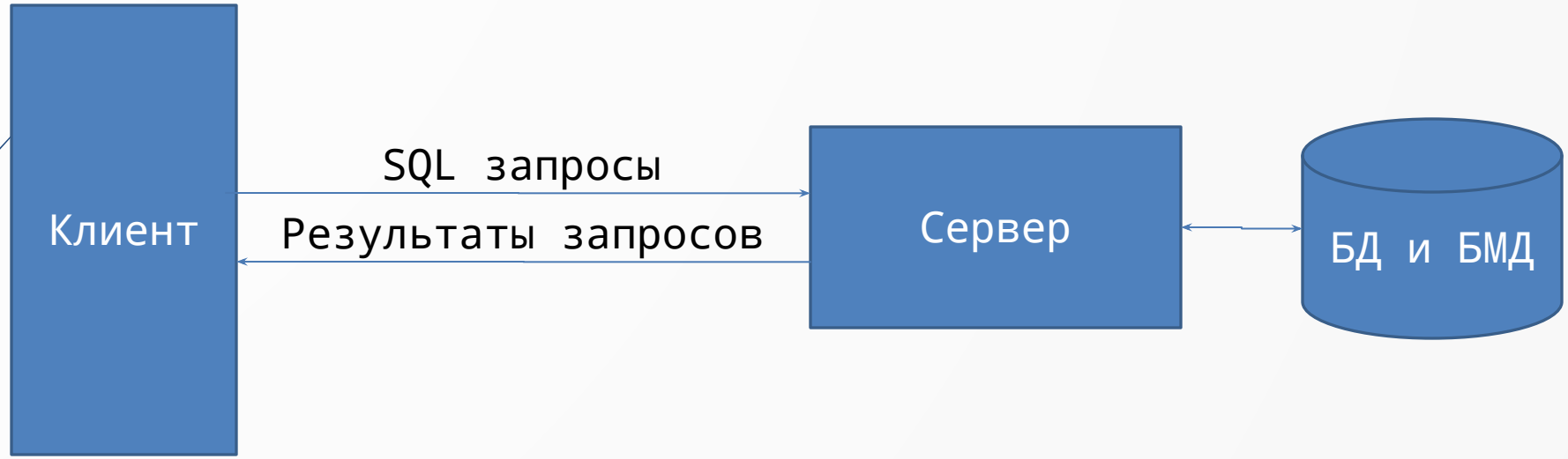
Работа в режиме удаленного доступа

Модель файлового сервера



Работа в режиме удаленного доступа

Модель удаленного доступа



Статическая и динамическая балансировка нагрузки

Статическая балансировка

- Выполняется до начала выполнения распределенного приложения

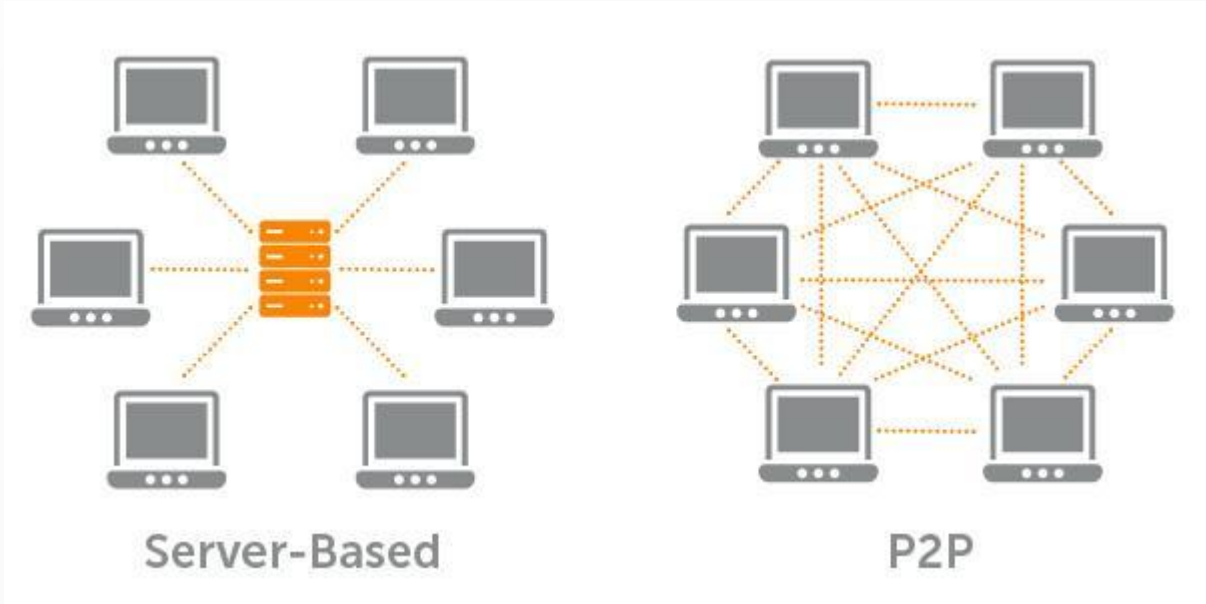
Динамическая балансировка

- Предусматривает распределение вычислительной нагрузки на узлы во время выполнения приложения

Диффузная балансировка нагрузки

Диффузная балансировка нагрузки является методом динамической балансировки загрузки процессоров, где перераспределение вычислительной нагрузки выполняется между логически соседними процессорами.

Централизованное и децентрализованное управление вычислениями



Построение алгоритмов динамической балансировки загрузки на основе технологии «клиент-сервер»

Нагрузка на объединенные между собой серверы распределяется с учетом закрепленных за ними ролей.

1) Автономный сервер
2) Рабочий сервер
3) Мастер-сервер
4) Выделенный мастер-сервер

Особенности обеспечения балансировки загрузки процессоров при использовании неструктурированных сеток

Проблема балансировки загрузки при решении задач на нерегулярных сетках часто сводится к задаче разбиения графа на заданное число частей.

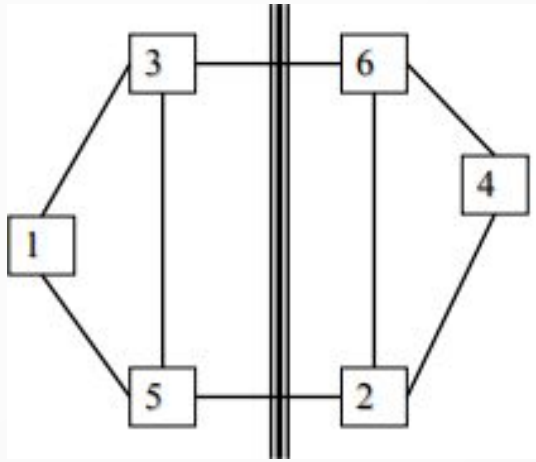
Для графов небольшого объема - спектральный алгоритм.

Для графов большого размера - иерархические алгоритмы:

1. рекурсивное огрубление графа (уменьшение его размера)
2. рекурсивная бисекция (разбиение графа на две части) вершин графа на заданное число групп
3. рекурсивное разукрупнение графа и локальное уточнение разбиения промежуточных графов

Спектральная матрица графа. Вектора Фидлера

$$a_{ij} = \begin{cases} w(v_i, v_j), & i \neq j \\ -\sum_{k=1,N}^{N} w(v_i, v_k), & i = j \end{cases}$$



$$A = \begin{pmatrix} -2 & 0 & 1 & 0 & 1 & 0 \\ 0 & -3 & 0 & 1 & 1 & 1 \\ 1 & 0 & -3 & 0 & 1 & 1 \\ 0 & 1 & 0 & -2 & 0 & 1 \\ 1 & 1 & 1 & 0 & -3 & 0 \\ 0 & 1 & 1 & 1 & 0 & -3 \end{pmatrix}$$

Иерархические алгоритмы разбиения больших графов

Алгоритм Кернигана-Лина:

- Формирование множества пар вершин для перестановки.
- Построение новых вариантов разбиения графа.
- Выбор лучшего варианта разбиения графа.
- Проверка использования всех вершин.
- Выбор наилучшего варианта разбиения графа.

Методы визуализации двух и трехмерных сеточных данных

- Под двухмерной визуализацией понимается отображение данных на плоскости в виде изображения. Для формирования изображений обычно используется либо растровый, либо векторный способы.
- Трёхмерная визуализация данных осуществляется на основе модели данных, полученных при помощи обработки снимков. Под моделью в данном случае понимается описание исходных данных на строго определённом языке или в виде структуры.

Сжатие сеточных данных, заданных на структурированных и неструктурированных сетках.

Работоспособность системы распределенной визуализации трехмерных скалярных полей определяется, главным образом, качеством алгоритмов сжатия неструктурированных сеток. Появление триангуляции вызвано наличием разбиения каждого из кубов сетки на набор пирамид, обеспечивающих в большинстве случаев однозначность построения изоповерхности.

Построение систем удаленной обработки данных и визуализации на основе технологии «клиент-сервер».

Организация ЛВС на предприятии дает возможность распределить ресурсы ПК по отдельным функциональным сферам деятельности и изменить технологию обработки данных в направлении децентрализации.

Распределенная обработка данных имеет следующие преимущества:

· возможность увеличения числа удаленных взаимодействующих пользователей, выполняющих функции сбора, обработки, хранения и передачи информации;

· снятие пиковых нагрузок с централизованной базы путем распределения обработки и хранения локальных баз на разных персональных компьютерах;

· обеспечение доступа пользователей к вычислительным ресурсам ЛВС;

· обеспечение обмена данными между удаленными пользователями.