Кластерные вычисления

д.т.н. Елена Янакова

Компетенции

Способностью осуществлять поиск, хранение, обработку и анализ информации из различных источников и баз данных, представлять ее в требуемом формате с использованием информационных, компьютерных и сетевых технологий.

Знание основ построения и функционирования вычислительных систем, в частности, многопроцессорных и распределенных.

Быстрее, выше, сильнее (Citius, Altius, Fortius)

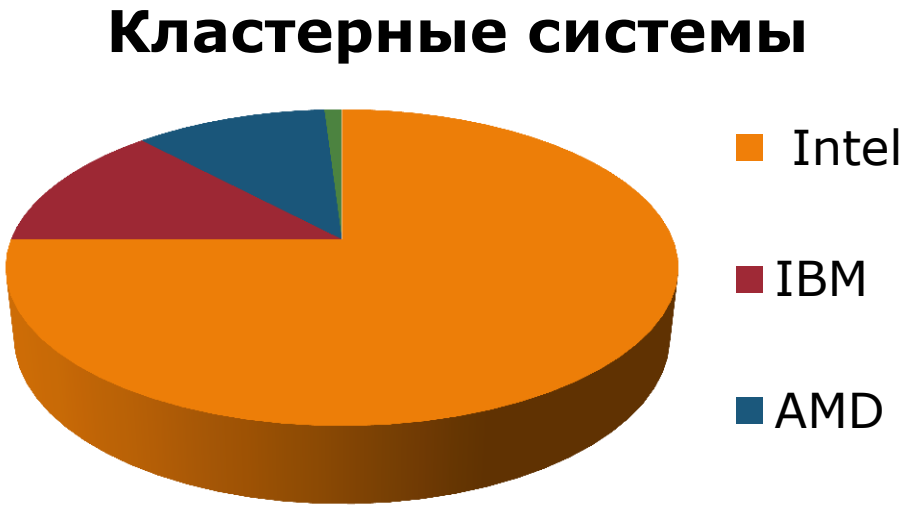
«Производительность вычислительных систем удваивается каждый 18 месяцев»

Гордон Мур

Суперкомпьютер «Roadrunner» 2008 г, IBM - 1,026 петафлопс

Суперкомпьютер «ASCI Red», 1997, Intel - один терафлопс

Быстрее, выше, сильнее (Citius, Altius, Fortius)



81% систем используют два типа интерконнекта: Gigabit Ethernet или Infiniband.

85% систем работают под управлением операционной системы из семейства Linux.

Мини-кластеры

1994 год кластеры на основе обычных рабочих станций.

2006 год кластеры промышленного исполнения.

Мини-кластер
T-Forge Mini

Технические характеристики:
-до 4-х двухпроцессорных узлов AMD Opteron™;
- Gigabit Ethernet.
- память до 64 Гб.
- до 4-х устройств HDD SATA до 2 Тб.
- 4 блока питания мощностью 350 Вт.
-до 9-ти портов Gigabit Ethernet.
- видео-карта ATI Rage XL 8Mb.
- Операционная система: ОС SUSE Linux Enterprise Server 9, RedHat Enterprise Linux 4 или Microsoft Windows Compute Cluster Server 2003.

Основные определения и задачи

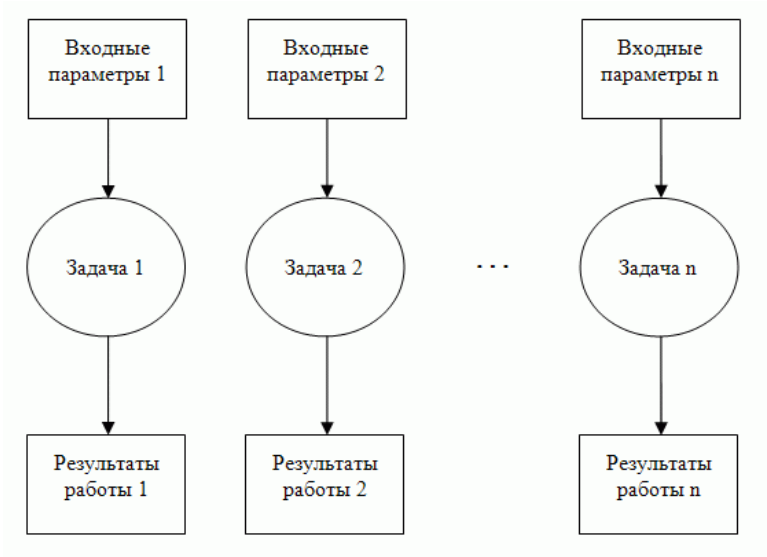
Кластер - это локальная вычислительная система, состоящая из множества работающих совместно вычислительных узлов, объединенных сетью, составляющих единый вычислительный ресурс.

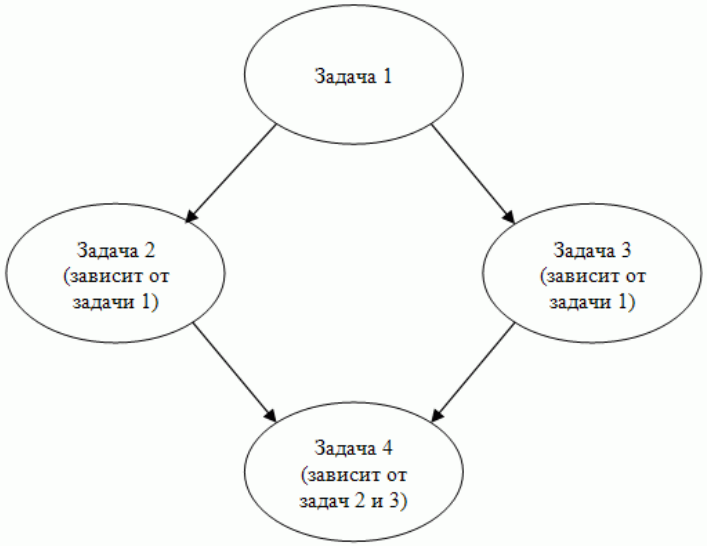
Параллельное приложение для кластерной системы представляет собой несколько процессов, которые общаются друг с другом по сети.

Три крупных этапа развития архитектуры кластеных систем

- 1-й этап – увеличение тактовой частоты работы процессоров (до 2000 г.);
- 2-й этап – появление многоядерных процессоров (после 2000г.);
- 3-й этап – появление распределенных вычислений на промышленном уровне (после 2006 г).

Работа с заданиями в кластерных системах





Архитектура кластеров

Вычислительные узлы кластера - узлы, на которых происходит запуск счетных заданий. Вычислительный узел может содержать либо один микропроцессор, либо несколько, образуя, симметричную конфигурацию (SMP).

Головной узел - выделенный узел, управляющий вычислениями на кластере (принимает задания, поддерживает очередь заданий, планирует запуски, собирает статистику выполнения и осуществляет другие служебные операции).

Особенности архитектуры кластерных систем

- в качестве вычислительных узлов в основном используют не специализированные устройства, а серийно выпускаемые вычислительные машины и системы;
- в единую систему объединяются вычислительные узлы разной архитектуры и конфигурации. Кластерные системы с одинаковыми узлами называют гомогенными кластерами, а с разнотипными узлами - гетерогенными кластерами;
- вычислительные узлы кластера могут функционировать самостоятельно и отдельно от кластера;
- каждый узел кластера работает под управлением своей операционной системы, поэтому отсутствует проблема когерентности кешей. Чаще всего используют операционные системы (ОС) Linux, FreeBSD, Solaris и версии Windows;
- легкая масштабируемость позволяет достичь высочайших показателей производительности.

Взаимодействие вычислительных узлов в кластерных системах

Технологии:
1) стандартные сетевые технологии, такие как Fast/Gigabit Ethernet, Myrinet (≈120 Мбит/с).
2) специальные технологии, обеспечивающих сверхбыструю передачу данных между узлами кластера, на базе шинной архитектуры или коммутатора.

Протоколы:
1) транспортные протоколы: TCP (Transmission Control Protocol) или UDP (User Datagram Protocol).
2) специальные протоколы, например, Virtual Interface Architecture (VIA).

Основные технические характеристиками сетей:
1) задержка (latency) передачи данных;
2) широта полосы пропускания (bandwidth).

Преимущества и недостатки кластерных систем

Преимущества:

- **легкая масштабируемость**: кластерные технологии предоставляют возможность создавать большие кластеры, превосходящие по вычислительной мощности высокопроизводительные одиночные вычислительные машины, и постепенно наращивать производительность, добавляя новые узлы;
- **высокий коэффициент готовности**: каждый узел кластера является самостоятельной вычислительной машиной или вычислительной системой, поэтому отказ одного из узлов не приводит к потере работоспособности кластера;
- **превосходное соотношение цена/производительность**: кластер может состоять из нескольких вычислительных машин или вычислительной системы, стоимость которых будет ниже, чем одиночной ЭВМ с эквивалентной вычислительной мощностью.

Основной недостаток: время взаимодействия между узлами кластера значительно выше, чем в вычислительных системах других типов.

Программное обеспечение (ПО) кластерных систем

ПО необходимое для построения и функционирования кластеров включает:

- ПО для разработки (компиляторы для языков, библиотеки различного назначения, средства измерения производительности, отладчики);

- специализированное ПО, система управления (СУ) кластером, организующее бесперебойную работу кластера (при отказе одного или нескольких узлов) и управление ресурсами, к которым относятся средства инсталляции, администрирования и планирования потоков работ.

Две модели программирования:

- модель на основе «передачи сообщений» (message passing), реализованная в виде стандарта MPI (Message Passing Interface);
- модель на основе «глобального распределенного адресного пространства» (GPAS – global partitioned address space), реализованная с помощью языков HPF (High Performance Fortran) и UPC (Unified Parallel C).

Функции СУ кластерных систем

- перераспределение вычислительной нагрузки (статическая и динамическая балансировка) в случае отказа одного или нескольких узлов кластера;

- восстанавливает вычисления при сбое в узле (последовательное и неблокирующее создание контрольных точек);

- постоянно контролирует работоспособность всех остальных узлов, путем рассылки каждым узлом сигнала keepalive («пока жив») или heartbeat («сердцебиение»);

- управление над выполняемыми заданиями, пользователями, ресурсами;
- поддержка web-интерфейсов, задания должны формулироваться с использованием скриптовых языков.

СУ кластерами под UNIX-подобные операционные системы

- Beowulf;
- EnFuzion;
- MOSIX;
- Condor;
- LoadLeveler;
- LSF;
- PBS;
- Cleo.

СУ кластерами под UNIX-подобные операционные системы

Таблица – Характеристики систем управления кластерами

	Beowulf	EnFuzion	MOSIX	Condor	Load Leveler	LSF	Cleo	PBS Pro
Пакетные задания	Да	Да	Да	Да	Да	Да	Да	Да
Файл задания	Да	Да	Нет	Да	Да	Да	Нет	Да
Зависимость задач по результатам	Нет	Нет	Нет	Нет	Нет	Да	Нет	Нет
Статус задачи	Да	Да	Да	Да	Да	Да	Да	Да
Балансировка	Нет	Нет	Да	Нет	Да	Да	Да	Да
Задание ресурсов	Да	Нет	Нет	Да	Да	Да	Да	Да
Задание приоритетов	Да	Да	Нет	Да	Да	Да	Да	Да
Интерактивные задачи	Нет	Нет	Да	Нет	Да	Да	Нет	Да
Ограничение ресурсов	Да	Нет	Нет	Нет	Да	Да	Нет	Да
Контрольные точки	Нет	Нет	Нет	Да	Да	Да	Нет	Нет
Миграция процессов	Нет	Нет	Да	Да	Да	Да	Нет	Нет
Перекомпиляция	Да	Нет	Нет	Да	Нет	Нет	Нет	Нет
Защита от сбоев	Нет	Да	Да	Да	Да	Да	Нет	Да
Параллельные программы	Да	Нет	Нет	Да	Да	Да	Да	Да
Монопольный режим	Да	Да	Да	Да	Да	Да	Да	Да
Интерфейс	Да	Нет	Да	Нет	Да	Да	Нет	Да

Condor, PBS, Platform LSF - UNIX системы управления

2005-2006 г. - появляются кластеры промышленного исполнения.

Microsoft High Performance Computing Server 2008

2006 г - Microsoft объявила о выходе собственной системы управления Microsoft Compute Cluster Server 2003 (CCS).

2008 г - Microsoft High Performance Computing Server (HPC Server 2008 или HPC 2008) - новая версия системы управления кластером от Microsoft.

Microsoft High Performance Computing Server 2008

Эффективное распределение задач:

- планирование на различных уровнях (ядра, сокеты, узлы);
- управление политикой планирования (адаптивное выделение ресурсов, приоритетное прерывание обслуживания);

Удобные инструменты администрирования систем:

- создание групп из вычислительных узлов;
- возможность использования нескольких головных узлов (повышение надежности);
- упрощение развертывания больших кластеров.

Оптимизация интерфейса передачи сообщений (MPI):

- поддержка сетевого интерфейса NetworkDirect (повышение производительности MPI приложений);
-оптимизация MPI операций, работающих через общую память;
- улучшенная поддержка высокопроизводительных систем хранения данных.

Microsoft High Performance Computing Server 2008. Основные понятия

Задание (job) - запрос на выделение вычислительного ресурса кластера для выполнения одной или нескольких задач.
Задача (task) - команда, которая должна быть выполнена на кластере.
Планировщик заданий (job scheduler) - сервис, отвечающий за поддержание очереди заданий, выделение системных ресурсов, постанову задач на выполнение, отслеживание состояния запущенных задач.
Узел (node) - вычислительный компьютер, включенный в кластер.
Сокет (socket) - один или нескольких вычислительных устройств (процессоров) узла.
Ядро (core) - минимальный счетный элемент вычислительного узла.
Очередь (queue) - список заданий, отправленных планировщику для выполнения на кластере.
Список задач (task list) - эквивалент очереди заданий для задач каждого конкретного задания.

Microsoft MPI

Message Passing Interface (MPI, интерфейс передачи сообщений) - открытый стандарт, описывающий интерфейс обмена сообщениями между процессами параллельной программы.

MS MPI - реализация стандарта MPI (версии 2) от Microsoft. MS MPI включает всю функциональность, описанную в стандарте, за исключением динамического создания процессов.

МИЭТ2008


СУПЕРКОМПЬЮТЕРЫ
Top50
Система МИЭТ2008 на базе
процессоров Intel Xeon 5335 2 ГГц и сети Gigabit Ethernet
Московского государственного института электронной техники
занимает позицию
№ 43
в рейтинге Top50 самых мощных компьютеров СНГ с результатом
по производительности на тесте Linpack
871.5 Gflop/s
9-ая редакция рейтинга Top50 опубликована на на Всероссийской научной конференции
«Научный сервис в сети Интернет: решение больших задач»
23 сентября 2008 года
Примите искренние поздравления от составителей Top50!
Воеводин Вл. В.
Научно-исследовательский
вычислительный центр МГУ имени М.В.Ломоносова
Шабанов Б. М.
Межведомственный
суперкомпьютерный центр РАН

Microsoft High Performance Computing Server 2008. Сетевые топологии

Сетевые интерфейсы: - Gigabit Ethernet;
- 10 Gigabit Ethernet;
- Infiniband;
- Myrinet.

Понятия, используемые при описании сетевых топологий:

- открытая сеть (public network) - корпоративная сеть организации, соединенная с головным и (возможно) вычислительными узлами кластера;

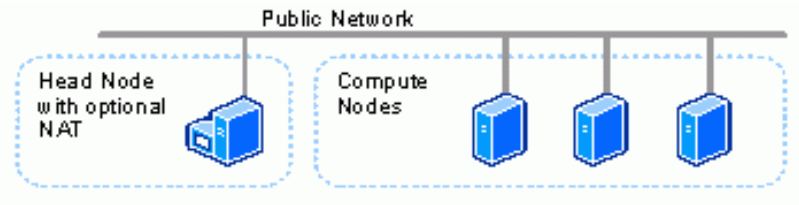
- закрытая сеть (private network) - выделенная сеть, предназначенная для коммуникации между узлами вычислительного кластера;

- MPI-сеть (MPI network) - выделенная сеть, предположительно, наиболее быстрая, через которую идет трафик MPI – программ.

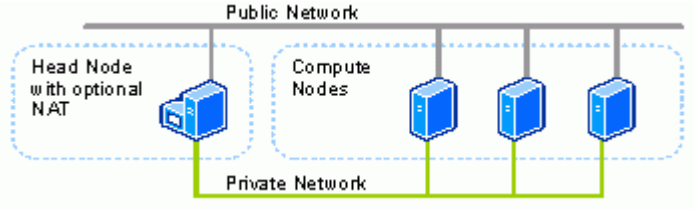
Microsoft High Performance Computing Server 2008. Сетевые топологии

Пять вариантов сетевых топологий.

1. **Все узлы в открытой сети**

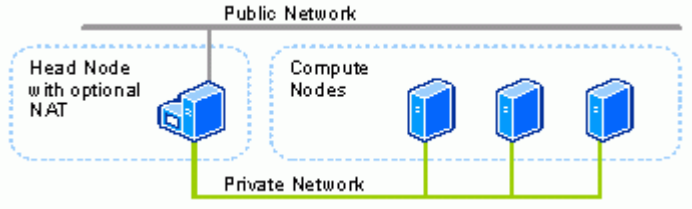


2. **Все узлы в открытой и закрытой сетях**

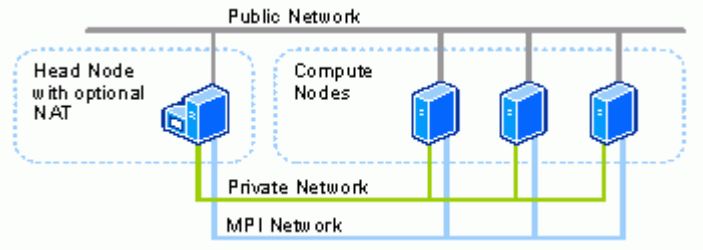


Microsoft High Performance Computing Server 2008. Сетевые топологии

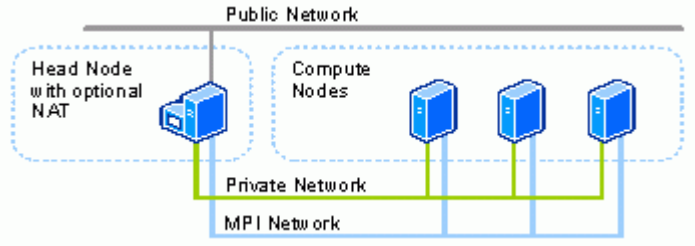
3. **Вычислительные узлы соединены только закрытой сетью**



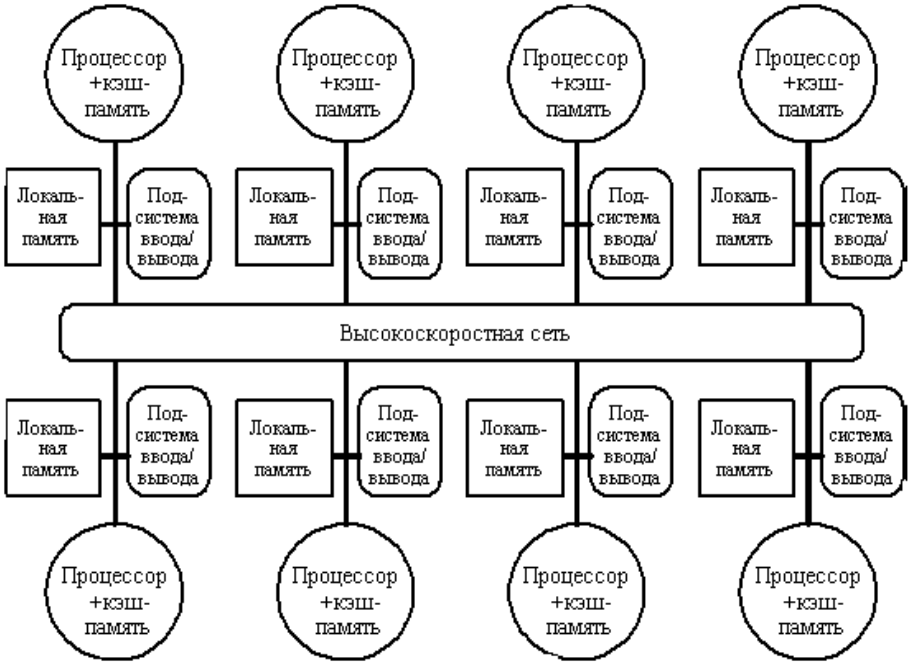
4. **Все узлы в открытой, закрытой и MPI – сетях**



5. **Вычислительные узлы соединены закрытой и MPI - сетями**



Архитектура машины с распределенной памятью



- машины с разделяемой (общей) памятью;
- машины с множеством адресных пространств.

Преимущества обмена с помощью передачи сообщений

1. Аппаратура может быть более простой, особенно по сравнению с моделью разделяемой памяти, которая поддерживает масштабируемую когерентность кэш-памяти.
2. Модели обмена понятны, принуждают программистов (или компиляторы) уделять внимание обмену, который обычно имеет высокую, связанную с ним стоимость.

Характеристики производительности механизма обмена

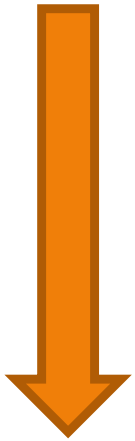
Полоса пропускания: в идеале полоса пропускания механизма обмена будет ограничена полосами пропускания процессора, памяти и системы межсоединений, а не какими-либо аспектами механизма обмена.

Задержка: в идеале задержка должна быть настолько мала, насколько это возможно (минимальные задержки с инициированием и завершением обмена).

Упрятывание задержки: насколько хорошо механизм скрывает задержку путем перекрытия обмена с вычислениями или с другими обменами.

Жизненный цикл кластерных систем

1. Проектирование.
2. Отладка и ввод в эксплуатацию.
3. Эксплуатация/модификация.



тестирование

Жизненный цикл кластерных систем. Тестирование

Классификация тестовых программ:
а) "*Игрушечные " тесты* (*toy benchmarks*);
б) микротесты (*microbenchmarks*);
- производительность центрального процессора;
- производительность и пропускную способности локальной оперативной памяти;
- скорость базовых операции ввода/вывода;
- производительность и пропускная способность коммуникационной среды.

Микротесты объединяются в пакеты тестов:
- ядра (*kernels*) ;
- синтетические тесты (*synthetic benchmarks*);
- приложения (*application benchmarks*);
- пакеты тестов (*benchmarks suites*).

Стандартные тесты производительности. Linpack benchmark

Решении системы линейных арифметических уравнений вида:

$$Ax=b.$$

Метод LU-разложения — представление матрицы A в виде произведения двух матриц, A=LU , где L и U — нижняя и верхняя треугольные матрицы.

$$LUx=b.$$

Решение в два этапа:

1) Ly=b решается методом прямой подстановки;
2) Ux=y решается методом обратной подстановки.

Linpack. Решаемая задача

1) A разделяется на блоки NB x NB (NB - параметр алгоритма, часто от 32 - 256).
2) Блоки разбиваются сеткой P x Q (P x Q < число узлов в кластере).
3) Факторизация матрицы.
4) Рассылка результатов всем узлам.
5) последовательно решается две системы уравнений (см. предыдущий слайд)

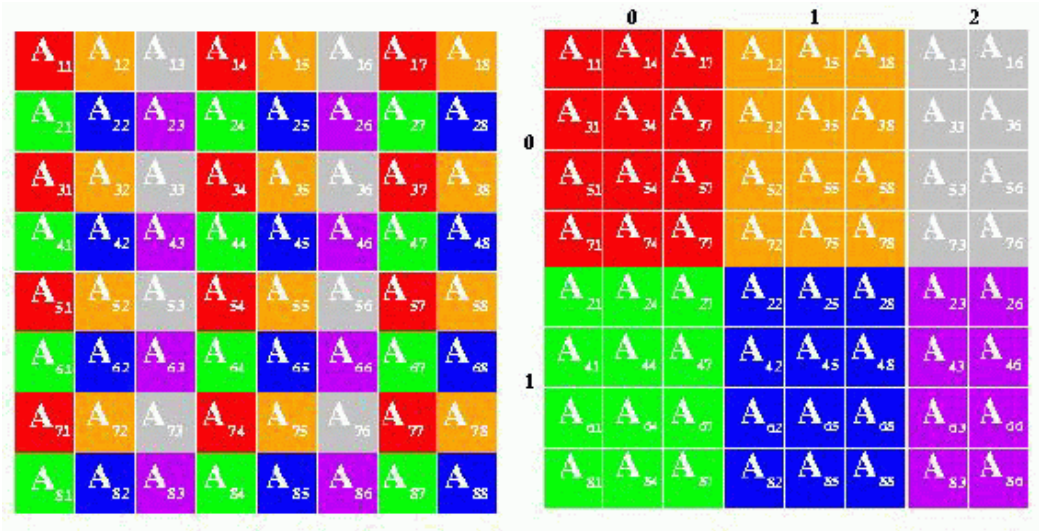


Рисунок - Схема распределения данных при решении системы линейных уравнений в тесте Linpack

Linpack. Решаемая задача

Задача считается успешно решенной, а тест считается выполненным, если выполнены следующие условия:

$$\|Ax-b\|\|Ax-b\|_\infty /(\varepsilon * \|A\|_1 * N) < \alpha \|Ax-b\|_\infty /(\varepsilon * \|A\|_1 * \|x\|_1) <$$

$$< \alpha \|Ax-b\|_\infty /(\varepsilon * \|A\|_\infty * \|x\|_\infty) < \alpha$$

- ε - точность представления чисел с плавающей точкой,
- α - константа, задаваемая в конфигурационном файле,

$$||A||_1 = max_j \sum_i a_i j$$

$$||A||_\infty = max_i \sum_j a_i j$$

$$||x||_\infty = max_i x_i$$

$$||x||_1 = \sum_i x_i$$

Отладка параллельной программы. Intel® *Thread Checker*

!Одним из основных этапов разработки программ является - *отладка*

Шаги процесса отладки приложения:
1) определение факта наличия ошибки;
2) поиск (локализация) ошибки;
3) выяснение причин ошибки;
4) определение способа устранения ошибки;
5) устранение ошибки.

Возможности Intel® *Thread Checker*

– **гонки данных (**несколько потоков работают с разделяемыми данными и конечный результат зависит от соотношения скоростей потоков**);**
– **тупики** (взаимная блокировка потоков);
– **потоки в состоянии ожидания;**
– **потерянные сигналы**;
– **заброшенные замки (**возникают в ситуации, когда поток захватил некоторый ресурс (критическую секцию, мьютекс) и был снят с выполнения по той или иной причине**).**

Принцип сбора информации ITC

В процессе анализа контролируется:
–доступ к памяти;
–операции синхронизации;
–операции создания потоков.

!Оптимизация должна быть выключена.
! Компиляция потоко-безопасного кода: -MT[d], -MD[d].
! Использование debug опций: -Z[i,I,7], -Od.
! Связывание с ключом /fixed:no.

Оптимизация параллельной программы

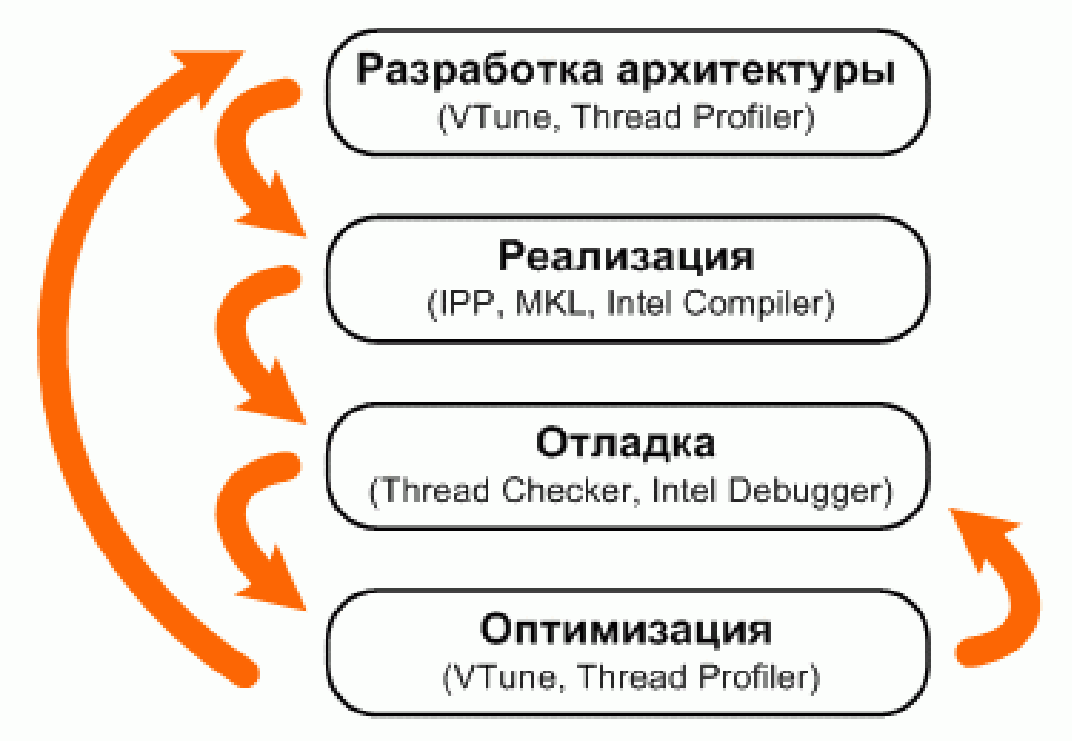
Рассмотрели:

–аппаратную основу кластера;
–системное обеспечение, необходимого для работы кластера;
–оценка эффективности построенного кластера;
–среды разработки *параллельных программ*;
–оптимизация параллельной программы.

Семейство инструментов для поддержки многопоточного программирования компании Intel:

Intel® C/C++ Compiler и Intel® Fortran Compiler - компиляторы;
Intel® Performance Libraries - библиотеки математических функций;
Intel® VTune Performance *Analyzer* - *профилировщик*;
Intel® Thread *Checker* - отладчик многопоточных приложений;
Intel® Thread Profiler - *профилировщик* многопоточных приложений;
Intel® Threading Building Blocks - C++ библиотека времени выполнения, представляющая набор примитивов для разработки многопоточных приложений.

Цикл разработки многопоточного приложения



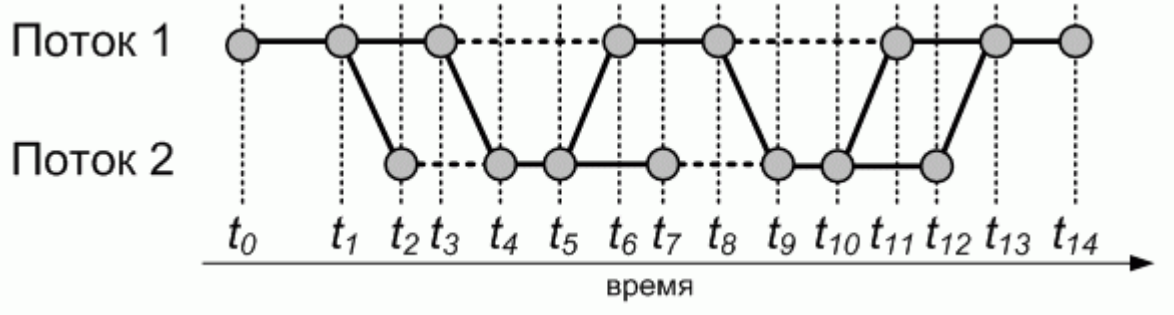
Цели профилирования

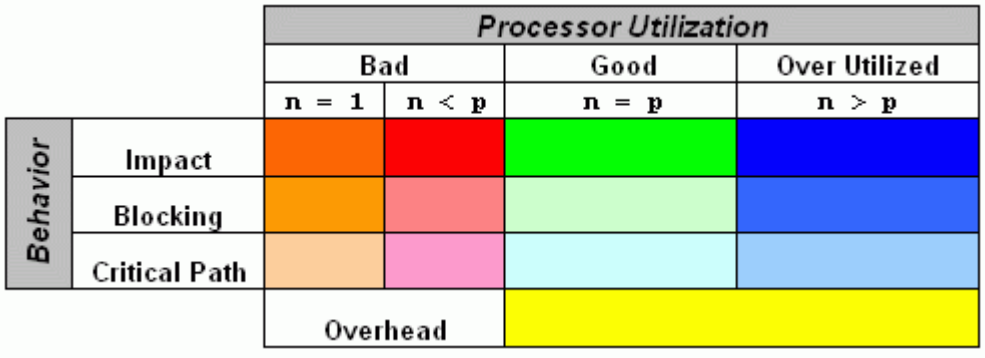
Выявить участки кода, в которых *приложение* проводит основную часть времени и оптимизировать их.

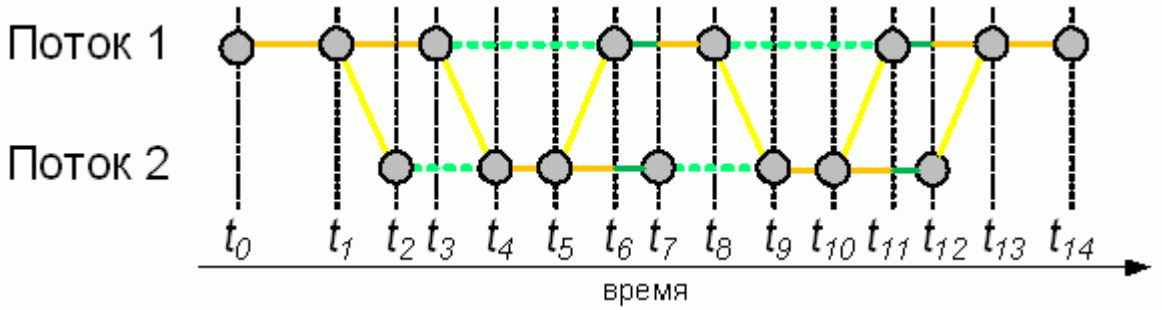
Эвристический закон "**20/80**", утверждающий, что *приложение* проводит 80% времени работы в 20% кода.

Назначение Intel® Thread Profiler :
– анализ эффективности распределения вычислительной нагрузки между потоками;
– анализ эффективности обращения к разделяемым ресурсам;
– определение наиболее медленных потоков для оптимизации;
– выбор оптимальной архитектуры многопоточного приложения;
– анализ эффективности работы с потоками и примитивами синхронизации;
– анализ масштабируемости приложения на вычислительных узлах с различным числом процессоров.

Понятие критического пути







Выводы

Рассмотрели:

– аппаратную основу кластера;
– системное обеспечение, необходимого для работы кластера (Microsoft High Performance Computing Server);
– оценка эффективности построенного кластера (Linpack);
– среды разработки параллельных программ;
– оптимизацию параллельной программы.