

Тема: Классификации архитектур вычислительных систем

Организация схем коммутации в МВС

ВВЕДЕНИЕ.....	1
КЛАССИФИКАЦИЯ ФЛИННА	4
Дополнения ВАНГА и БРИГТСА К КЛАССИФИКАЦИИ ФЛИННА.....	6
ОСНОВНЫЕ КЛАССЫ СОВРЕМЕННЫХ ПАРАЛЛЕЛЬНЫХ КОМПЬЮТЕРОВ.....	8
БАЗОВЫЕ ХАРАКТЕРИСТИКИ ОСНОВНЫХ КЛАССОВ СОВРЕМЕННЫХ КОМПЬЮТЕРОВ.....	9
МАССИВНО-ПАРАЛЛЕЛЬНЫЕ СИСТЕМЫ (MPP).....	9
СИММЕТРИЧНЫЕ МУЛЬТИПРОЦЕССОРНЫЕ СИСТЕМЫ (SMP)	11
СИСТЕМЫ С НЕОДНОРОДНЫМ ДОСТУПОМ К ПАМЯТИ (NUMA)	12
ПАРАЛЛЕЛЬНЫЕ ВЕКТОРНЫЕ СИСТЕМЫ (PVP).....	14
КЛАСТЕРНЫЕ СИСТЕМЫ	14
ОРГАНИЗАЦИЯ СХЕМ КОММУТАЦИИ В МВС С ОБЩЕЙ ПАМЯТЬЮ.....	15
ОРГАНИЗАЦИЯ СХЕМ КОММУТАЦИИ В МВС С РАСПРЕДЕЛЕННОЙ ПАМЯТЬЮ.....	15
АРХИТЕКТУРА СИСТЕМ СО СМЕШЕННОЙ ОРГАНИЗАЦИЕЙ ПАМЯТИ.....	15

Введение

Стремительное развитие науки и проникновение человеческой мысли во все новые области вместе с решением поставленных прежде проблем постоянно порождает поток вопросов и ставит новые, как правило более сложные, задачи. Во времена первых компьютеров казалось, что увеличение их быстродействия в 100 раз позволит решить большинство проблем, однако гигафлопная производительность современных суперЭВМ сегодня является явно недостаточной для многих ученых.

- Электро и гидродинамика,
- сейсморазведка и
- прогноз погоды,
- моделирование химических соединений,
- исследование виртуальной реальности - вот далеко не полный список областей науки, исследователи которых используют каждую возможность ускорить выполнение своих программ.

Наиболее перспективным и динамичным направлением увеличения скорости решения прикладных задач *является широкое внедрение идей параллелизма в работу вычислительных систем*. К настоящему времени спроектированы и опробованы сотни различных компьютеров, использующих в своей архитектуре тот или иной вид параллельной обработки данных. В научной литературе и технической документации

можно найти более десятка различных названий, характеризующих лишь общие принципы функционирования параллельных машин: векторно-конвейерные, массивно-параллельные, компьютеры с широким командным словом, систолические массивы, гиперкубы, спецпроцессоры и мультипроцессоры, иерархические и кластерные компьютеры, dataflow, матричные ЭВМ и многие другие. Если же к подобным названиям для полноты описания добавить еще и данные о таких важных параметрах, как, например, организация памяти, топология связи между процессорами, синхронность работы отдельных устройств или способ исполнения арифметических операций, то *число различных архитектур станет и вовсе необозримым.*

Попытки систематизировать все множество архитектур начались после опубликования М.Флинном первого варианта классификации вычислительных систем в конце 60-х годов и непрерывно продолжаются по сей день. Ясно, что навести порядок в хаосе очень важно для лучшего понимания исследуемой предметной области, однако нахождение удачной классификации может иметь целый ряд существенных следствий.

В самом деле, вспомним открытый в 1869 году Д.И.Менделеевым периодический закон. Выписав на карточках названия химических элементов и указав их важнейшие свойства, он сумел найти такое расположение, при котором четко прослеживалась закономерность в изменении свойств элементов, расположенных в каждом столбце и в каждой строке. Теперь, зная положение какого-либо элемента в таблице, он мог с большой степенью точности описывать его свойства, не проводя с ним никаких непосредственных экспериментов. Другим, поистине фантастическим следствием, явилось то, что данный закон сразу указал на несколько "белых пятен" в таблице и позволил предсказать существование (!) и свойства (!!)

неизвестных до тех пор элементов. В 1875 году французский ученый Буабодран, изучая спектры минералов, открыл предсказанный Менделеевым галлий и впервые подробно описал его свойства. В свою очередь Менделеев, никогда прежде не видевший данного химического элемента, не только смог указать на ошибку в определении плотности, но и вычислил ее правильное значение.

Существующая классификация растительного и животного мира, в отличие от периодического закона, носит скорее описательный характер. С ее помощью намного сложнее предсказывать существование нового вида, однако знание того, что исследуемый экземпляр принадлежит такому-то роду/семейству/отряду/классу позволяет оправданно предположить наличие у него вполне определенных свойств.

Подобную классификацию хотелось бы найти и для архитектур параллельных вычислительных систем. Основной вопрос - что заложить в

основу классификации, может решаться по-разному, в зависимости от того, для кого данная классификация создается и на решение какой задачи направлена. Так, часто используемое деление компьютеров на

- персональные ЭВМ,
- рабочие станции, мини--ЭВМ,
- большие универсальные ЭВМ,
- минисупер--ЭВМ и
- супер--ЭВМ,

позволяет, быть может, примерно прикинуть стоимость компьютера.

Однако она не приближает пользователя к пониманию того, что от него потребуется для написания программы, работающий на пределе производительности параллельного компьютера, т.е. того, ради чего он и решился его использовать. Как это ни странно, но от обилия разных параллельных компьютеров страдает, прежде всего, конечный пользователь, для которого, вроде бы, они и создавались: он вынужден каждый раз подбирать наиболее эффективный алгоритм, он испытывает на себе "прелести" параллельного программирования и отладки, решает проблемы переносимости и затем все повторяется заново.

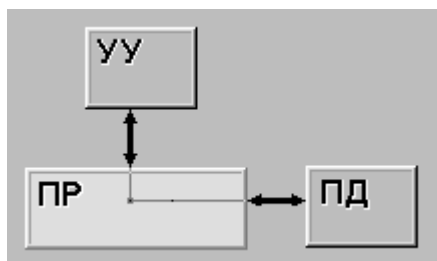
Хотелось бы, чтобы такая классификация помогла ему разобраться с тем, что представляет собой каждая архитектура, как они взаимосвязаны между собой, что он должен учитывать для написания действительно эффективных программ или же на какой класс архитектур ему следует ориентироваться для решения требуемого класса задач.

Одновременно удачная классификация могла бы подсказать возможные пути совершенствования компьютеров и в этом смысле она должна быть достаточно содержательной. Трудно рассчитывать на нахождение нетривиальных "белых пятен", например, в классификации по стоимости, однако размышления о возможной систематике с точки зрения простоты и технологичности программирования могут оказаться чрезвычайно полезными для определения направлений поиска новых архитектур.

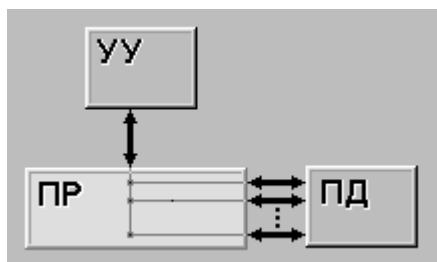
Рассмотрим наиболее известные на сегодня классификации.

Классификация Флинна

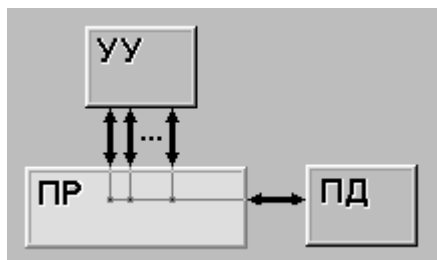
По-видимому, самой ранней и наиболее известной является классификация архитектур вычислительных систем, предложенная в 1966 году М.Флинном. Классификация базируется на понятии **потока**, под которым понимается **последовательность элементов, команд или данных, обрабатываемая процессором**. На основе числа потоков команд и потоков данных Флинн выделяет четыре класса архитектур: **SISD, MISD, SIMD, MIMD**.



SISD (single instruction stream / single data stream) - одиночный поток команд и одиночный поток данных. К этому классу относятся, прежде всего, классические последовательные машины, или иначе, машины фон-неймановского типа, например, PDP-11 или VAX 11/780. В таких машинах есть только один поток команд, все команды обрабатываются последовательно друг за другом и каждая команда инициирует одну операцию с одним потоком данных. Не имеет значения тот факт, что для увеличения скорости обработки команд и скорости выполнения арифметических операций может применяться конвейерная обработка - как машина CDC 6600 со скалярными функциональными устройствами, так и CDC 7600 с конвейерными попадают в этот класс.



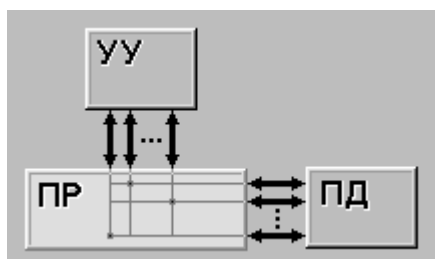
SIMD (single instruction stream / multiple data stream) - одиночный поток команд и множественный поток данных. В архитектурах подобного рода сохраняется один поток команд, включающий, в отличие от предыдущего класса, векторные команды. Это позволяет выполнять одну арифметическую операцию сразу над многими данными - элементами вектора. Способ выполнения векторных операций не оговаривается, поэтому обработка элементов вектора может производиться либо процессорной матрицей, как в ILLIAC IV, либо с помощью конвейера, как, например, в машине CRAY-1.



MISD (multiple instruction stream / single data stream) - множественный поток команд и одиночный поток данных. Определение подразумевает наличие в архитектуре многих процессоров, обрабатывающих один и тот же поток данных. Однако ни Флинн, ни другие специалисты в области архитектуры компьютеров до некоторого времени не могли представить убедительный пример реально существующей вычислительной системы, построенной на данном принципе. Ряд

исследователей относят конвейерные машины к данному классу, однако это не нашло окончательного признания в научном сообществе.

Появившиеся многоядерные компьютеры можно отнести к данному классу.



MIMD (multiple instruction stream / multiple data stream) - множественный поток команд и множественный поток данных. Этот класс предполагает, что в вычислительной системе есть несколько устройств обработки команд, объединенных в единый комплекс и работающих каждое со своим потоком команд и данных.

Итак, что же собой представляет каждый класс? В **SISD**, как уже говорилось, входят однопроцессорные последовательные компьютеры типа VAX 11/780. Однако многими критиками замечено, что в этот класс можно включить и векторно-конвейерные машины, если рассматривать вектор как одно неделимое данное для соответствующей команды. В таком случае в этот класс попадут и такие системы, как CRAY-1, CYBER 205, машины семейства FACOM VP и многие другие.

Бесспорными представителями класса **SIMD** считаются матрицы процессоров: ILLIAC IV, ICL DAP, Goodyear Aerospace MPP, Connection Machine 1 и т.п. В таких системах единое управляющее устройство контролирует множество процессорных элементов. Каждый процессорный элемент получает от устройства управления в каждый фиксированный момент времени одинаковую команду и выполняет ее над своими локальными данными. Для классических процессорных матриц никаких вопросов не возникает, однако в этот же класс можно включить и векторно-конвейерные машины, например, CRAY-1. В этом случае каждый элемент вектора надо рассматривать как *отдельный элемент потока данных*.

Класс **MIMD** чрезвычайно широк, поскольку включает в себя всевозможные мультипроцессорные системы: Cm*, C.mmp, CRAY Y-MP, Denelcor HEP, BBN Butterfly, Intel Paragon, CRAY T3D и многие другие. Интересно то, что если конвейерную обработку рассматривать как выполнение множества команд (операций ступеней конвейера) не над одиночным векторным потоком данных, а над множественным скалярным потоком, то все рассмотренные выше векторно-конвейерные компьютеры можно расположить и в данном классе.

Предложенная схема классификации вплоть до настоящего времени является самой **применяемой при начальной характеристике того или**

инного компьютера. Если говорится, что компьютер принадлежит классу SIMD или MIMD, то сразу становится понятным базовый принцип его работы, и в некоторых случаях этого бывает достаточно.

Однако видны и явные недостатки. В частности, некоторые заслуживающие внимания архитектуры, например **dataflow и векторно-конвейерные машины**, четко не вписываются в данную классификацию. Другой недостаток - это чрезмерная заполненность класса MIMD. Необходимо средство, более избирательно систематизирующее архитектуры, которые по Флинну попадают в один класс, но совершенно различны по числу процессоров, природе и топологии связи между ними, по способу организации памяти и, конечно же, по технологии программирования.

Наличие “пустого” класса (MISD) на момент формирования классификации оказалось чрезвычайно полезным для разработки принципиально новых концепций в теории и практике построения вычислительных систем, а именно многоядерных процессоров, которые появились уже в начале 21 века.

Дополнения Ванга и Бриггса к классификации Флинна

Оставляя четыре ранее введенных базовых класса (SISD, SIMD, MISD, MIMD), авторы внесли следующие изменения.

Класс **SISD** разбивается на два подкласса:

- архитектуры с единственным функциональным устройством, например, PDP-11;
- архитектуры, имеющие в своем составе несколько функциональных устройств - CDC 6600, CRAY-1, FPS AP-120B, CDC Cyber 205, FACOM VP-200.

В класс **SIMD** также вводится два подкласса:

- архитектуры с пословно-последовательной обработкой информации - ILLIAC IV, PEPE, BSP;
- архитектуры с разрядно-последовательной обработкой - STARAN, ICL DAP.

В классе **MIMD** авторы различают

- вычислительные системы со слабой связью между процессорами, к которым они относят все системы с распределенной памятью, например, Cosmic Cube,

- и вычислительные системы с сильной связью (системы с общей памятью), куда попадают такие компьютеры, как C.mmp, BBN Butterfly, CRAY Y-MP, Denelcor HEP.

Основные классы современных параллельных компьютеров

Классифицируя современные компьютеры, которые практически все относятся к классу MIMD будем основываться на анализе используемых в системах способах организации оперативной памяти. На рис.1 приведена классификация класса MIMD, которая фактически повторяет классификацию Р.Хокни, но в более современном изложении.

Данный подход позволяет различать два важных типа многопроцессорных систем - *multiprocessors* (*мультипроцессоры* или системы с общей разделяемой памятью) и *multicomputers* (*мультикомпьютеры* или системы с распределенной памятью).

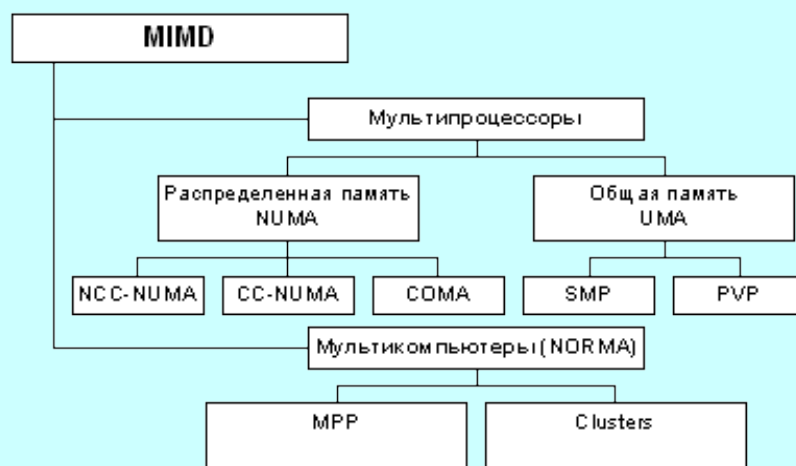


Рис. 1. Структура класса современных вычислительных систем

Для **мультипроцессоров** учитывается способ построения общей памяти. Возможный подход - использование единой (централизованной) общей памяти. Такой подход обеспечивает *однородный доступ к памяти (uniform memory access or UMA)* и служит основой для построения *векторных суперкомпьютеров (parallel vector processor, PVP)* и симметричных *мультипроцессоров (symmetric multiprocessor or SMP)*. Среди примеров первой группы суперкомпьютер Cray T90, ко второй группе относятся IBM eServer p690, Sun Fire E15K, HP Superdome, SGI Origin 300 и др.

Общий доступ к данным может быть обеспечен и при физически распределенной памяти (при этом, естественно, длительность доступа уже не будет одинаковой для всех элементов памяти). Такой подход именуется как *неоднородный доступ к памяти (non-uniform memory access or NUMA)*. Среди систем с таким типом памяти выделяют:

- Системы, в которых для представления данных используется только локальная кэш память имеющихся процессоров (*cache-only memory architecture or COMA*); примерами таких систем являются, например, KSR-1 и DDM;

- Системы, в которых обеспечивается однозначность (*когерентность*) локальных кэш памяти разных процессоров (*cache-coherent NUMA or CC-NUMA*); среди систем данного типа SGI Origin2000, Sun HPC 10000, IBM/Sequent NUMA-Q 2000;
- Системы, в которых обеспечивается общий доступ к локальной памяти разных процессоров без поддержки на аппаратном уровне когерентности кэша (*non-cache coherent NUMA or NCC-NUMA*); к данному типу относится, например, система Cray T3E.

Мультикомпьютеры (системы с распределенной памятью) уже не обеспечивают общий доступ ко всей имеющейся в системах памяти (*no-remote memory access or NORMA*). Данный подход используется при построении двух важных типов многопроцессорных вычислительных систем - *массивно-параллельных систем* (*massively parallel processor or MPP*) и *кластеров* (*clusters*). Среди представителей первого типа систем - IBM RS/6000 SP2, Intel PARAGON/ASCI Red, транспьютерные системы Parsytec и др.; примерами кластеров являются, например, системы AC3 Velocity и NCSA/NT Supercluster.

Следует отметить чрезвычайно быстрое развитие **кластерного типа** многопроцессорных вычислительных систем

Как уже отмечалось, основным параметром классификации параллельных компьютеров является наличие общей (SMP) или распределенной памяти (MPP). Нечто среднее между SMP и MPP представляют собой NUMA-архитектуры, где память физически распределена, но логически общедоступна. Кластерные системы являются более дешевым вариантом MPP. При поддержке команд обработки векторных данных говорят о векторно-конвейерных процессорах, которые, в свою очередь могут объединяться в PVP-системы с использованием общей или распределенной памяти.

Все большую популярность приобретают идеи комбинирования различных архитектур в одной системе и построения неоднородных систем.

При организациях распределенных вычислений в глобальных сетях (Интернет) говорят о [мета-компьютерах](#), которые, строго говоря, не представляют собой параллельных архитектур.

Кратко рассмотрим базовые характеристики определенных выше классов современных архитектур. (*Более подробно см. презентации докладов по соответствующим темам*)

Базовые характеристики основных классов современных компьютеров

Массивно-параллельные системы (MPP)

Архитектура: Система состоит из однородных *вычислительных узлов*, включающих:

- один или несколько центральных процессоров (обычно RISC),
- локальную память (прямой доступ к памяти других узлов невозможен),
- коммуникационный процессор или сетевой адаптер
- иногда - жесткие диски (как в SP) и/или другие устройства В/В

К системе могут быть добавлены специальные узлы ввода-вывода и управляющие узлы. Узлы связаны через некоторую коммуникационную среду (высокоскоростная сеть, коммутатор и т.п.)

Примеры: IBM RS/6000 SP2, Intel PARAGON/ASCI Red, SGI/CRAY T3E, Hitachi SR8000, транспьютерные системы Parsytec и др.

Масштабируемость: Общее число процессоров в реальных системах достигает нескольких тысяч (ASCI Red, Blue Mountain).

Операционная система: Существуют два основных варианта:

1. Полноценная ОС работает только на управляющей машине (front-end), на каждом узле работает сильно урезанный вариант ОС, обеспечивающие только работу расположенной в нем ветви параллельного приложения. Пример: Cray T3E.
2. На каждом узле работает полноценная UNIX-подобная ОС (вариант, близкий к кластерному подходу). Пример: IBM RS/6000 SP + ОС AIX, устанавливаемая отдельно на каждом узле.

Модель программирования: Программирование в рамках модели **передачи сообщений** (MPI, PVM, BSPlib)

Распределенность памяти означает, что каждый процессор имеет непосредственный доступ только к своей локальной памяти, а доступ к данным, расположенным в памяти других процессоров, выполняется другими способами.

Чтобы переслать информацию от процессора к процессору, необходим механизм передачи сообщений по сети, связывающей вычислительные узлы. Для абстрагирования от подробностей функционирования коммуникационной аппаратуры и программирования на высоком уровне, используются библиотеки передачи сообщений. Несмотря на существенные различия средств межпроцессорного взаимодействия в разных системах по скоростным параметрам и по способу аппаратной реализации, библиотеки обмена сообщениями выполняют приблизительно одни и те же функции.

Выбор топологии машины часто определяет способ решения прикладной задачи. Надо заметить, что оптимизация алгоритмов для параллельных архитектур существенно отличается от той же работы для последовательных систем. Если переход с одного скалярного процессора на другой практически никогда не требует пересмотра алгоритма, то алгоритм, идеально приспособленный для одной параллельной архитектуры, на другой машине (с тем же числом процессоров того же типа) может работать неприемлемо медленно. Для оценки производительности распределенной

системы, кроме топологии связей, необходимо знать скорость выполнения арифметических операций, время инициализации канала связи и время передачи единицы объема информации. Если топология системы не тривиальна, то в состав операционной системы или пакета передачи сообщений приходится включать процедуры маршрутизации сообщений, работающие на каждом узле и обеспечивающие пересылку транзитных сообщений. Они также вызывают задержку при передаче информации между узлами, не имеющими прямого канала связи

Таким образом, применение дешевых процессоров позволяет сделать относительно недорогой суперкомпьютер. Широкому распространению подобных архитектур препятствует в основном отсутствие эффективных параллельных программ, полностью использующих их возможности.

Симметричные мультипроцессорные системы (SMP)

Архитектура: Система состоит из нескольких однородных процессоров и массива общей памяти (обычно из нескольких независимых блоков). Все процессоры имеют доступ к любой точке памяти с одинаковой скоростью. Процессоры подключены к памяти либо с помощью общей шины (базовые 2-4 процессорные SMP-сервера), либо с помощью crossbar-коммутатора (HP 9000). Аппаратно поддерживается когерентность кэшей.

Примеры: HP 9000 V-class, N-class; SMP-сервера и рабочие станции на базе процессоров Intel (IBM, HP, Compaq, Dell, ALR, Unisys, DG, Fujitsu и др.).

Масштабируемость: Наличие общей памяти сильно упрощает взаимодействие процессоров между собой, однако накладывает сильные ограничения на их число - не более 32 в реальных системах. Для построения масштабируемых систем на базе SMP используются кластерные или NUMA-архитектуры.

Операционная система: Вся система работает под управлением единой ОС (обычно UNIX-подобной, но для Intel-платформ поддерживается Windows NT). ОС автоматически (в процессе работы) распределяет процессы/нити по процессорам (scheduling), но иногда возможна и явная привязка.

Модель программирования: Программирование в модели **общей памяти**. (POSIX threads, OpenMP). Для SMP-систем существуют сравнительно эффективные средства автоматического распараллеливания.

SMP - это один компьютер с несколькими равноправными процессорами, но с одной памятью, подсистемой ввода/вывода и одной ОС. Каждый процессор имеет доступ ко всей памяти, может выполнять любую операцию ввода/вывода, прерывать другие процессоры и т.д., но это представление справедливо только на уровне программного обеспечения. На самом же деле в SMP имеется несколько устройств памяти.

Каждый процессор имеет по крайней мере одну собственную кэш-память, что необходимо для достижения хорошей производительности, поскольку основная память работает слишком медленно по сравнению со скоростью процессоров (и это соотношение все больше ухудшается), а кэш работает со скоростью процессора, но дорог, и поэтому устройства кэш-памяти обладают относительно небольшой емкостью. Из-за этого в кэш помещается лишь оперативная информация, остальное же хранится в основной памяти. Отсюда возникает проблема когерентности кэшей - получение процессором значения, находящегося в кэш-памяти другого процессора. Это решается при помощи отправки широковещательного запроса всем устройствам кэш-памяти, основной памяти и даже подсистеме ввода/вывода, если она работает с основной памятью напрямую, с целью получения актуальной информации.

Имеется еще одно следствие, связанное с параллелизмом. Неявно производимая аппаратурой SMP пересылка данных между кэшами является наиболее быстрым и самым дешевым средством коммуникации в любой параллельной архитектуре общего назначения. Поэтому при наличии большого числа коротких транзакций (свойственных, например, банковским приложениям), когда приходится часто синхронизовать доступ к общим данным, архитектура SMP является наилучшим выбором; любая другая архитектура работает хуже.

Тем не менее, архитектуры с разделяемой общей памятью не считаются перспективными. Основная причина довольно проста. Рост производительности в параллельных системах обеспечивается наращиванием числа процессоров, что приводит к тому, что узким местом становится доступ к памяти. Увеличение локальной кэш-памяти не способно полностью решить проблему: задача поддержания согласованного состояния нескольких банков кэш-памяти столь же трудна. Как правило, на основе общей памяти не создают систем с числом процессоров более 32, при необходимости объединяя их в кластерные или NUMA-архитектуры.

Системы с неоднородным доступом к памяти (NUMA)

Архитектура: Система состоит из однородных базовых модулей (плат), состоящих из небольшого числа процессоров и блока памяти. Модули объединены с помощью высокоскоростного коммутатора. Поддерживается единое адресное пространство, аппаратно поддерживается доступ к удаленной памяти, т.е. к памяти других модулей. При этом доступ к локальной памяти в несколько раз быстрее, чем к удаленной.

В случае, если аппаратно поддерживается когерентность кэшей во всей системе (обычно это так), говорят об архитектуре **cc-NUMA** (cache-coherent NUMA)

Примеры: HP 9000 V-class в SCA-конфигурациях, SGI Origin2000, Sun HPC 10000, IBM/Sequent NUMA-Q 2000, SNI RM600 и др.

Масштабируемость: Масштабируемость NUMA-систем ограничивается объемом адресного пространства, возможностями аппаратуры поддержки когерентности кэшей и возможностями операционной системы по управлению большим числом процессоров. На настоящий момент, максимальное число процессоров в NUMA-системах составляет 256 (Origin2000).

Операционная система: Обычно вся система работает под управлением единой ОС, как в SMP. Но возможны также варианты динамического "подразделения" системы, когда отдельные "разделы" системы работают под управлением разных ОС (например, Windows NT и UNIX в NUMA-Q 2000).

Модель программирования: Аналогично SMP.

По сути своей NUMA представляет собой большую SMP, разбитую на набор более мелких и простых SMP. Аппаратура позволяет работать со всеми отдельными устройствами основной памяти составных частей системы (называемых обычно узлами) как с единой гигантской памятью. Этот подход порождает ряд следствий. Во-первых, в системе имеется одно адресное пространство, распространяемое на все узлы. Реальный (не виртуальный) адрес 0 для каждого процессора в любом узле соответствует адресу 0 в частной памяти узла 0; реальный адрес 1 для всей машины - это адрес 1 в узле 0 и т.д., пока не будет использована вся память узла 0. Затем происходит переход к памяти узла 1, затем узла 2 и т.д. Для реализации этого единого адресного пространства каждый узел NUMA включает специальную аппаратуру (Dir), которая решает проблему когерентности кэшей, обеспечивая получение актуальной информации от других узлов.

Понятно, что этот процесс длится несколько дольше, чем если бы требуемое значение находилось в частной памяти того же узла. Отсюда и происходит словосочетание "неоднородный доступ к памяти". В отличие от SMP, время выборки значения зависит от адреса и от того, от какого процессора исходит запрос (если, конечно, требуемое значение не содержится в кэше).

Поэтому ключевым вопросом является степень "неоднородности" NUMA. Например, если для взятия значения из другого узла требуется только на 10% большее время, то это никого не задевает. В этом случае все будут относиться к системе как к SMP, и разработанные для SMP программы будут выполняться достаточно хорошо.

Однако в текущем поколении NUMA-систем для соединения узлов используется сеть. Это позволяет включать в систему большее число узлов, до 64 узлов с общим числом процессоров 128 в некоторых системах. В результате, современные NUMA-системы не выдерживают правила 10% - лучшие образцы замедление 200-300% и даже более. При такой разнице в скорости доступа к памяти для обеспечения должной эффективности следует позаботиться о правильном расположении требуемых данных. Чтобы этого добиться, можно соответствующим образом модифицировать операционную систему (и это сделали поставщики систем в архитектуре NUMA). Например,

такая операционная система при запросе из программы блока памяти выделяет память в узле, в котором выполняется эта программа, так что когда процессор ищет соответствующие данные, то находит их в своем собственном узле. Аналогичным образом должны быть изменены подсистемы (включая СУБД), осуществляющие собственное планирование и распределение памяти (что и сделали Oracle и Informix). Как утверждает компания Silicon Graphics, такие изменения позволяют эффективно выполнять в системах с архитектурой NUMA приложения, разработанные для SMP, без потребности изменения кода.

Параллельные векторные системы (PVP)

Архитектура: Основным признаком PVP-систем является наличие специальных векторно-конвейерных процессоров, в которых предусмотрены команды однотипной обработки векторов независимых данных, эффективно выполняющиеся на конвейерных функциональных устройствах.

Как правило, несколько таких процессоров (1-16) работают одновременно над общей памятью (аналогично SMP) в рамках многопроцессорных конфигураций. Несколько таких узлов могут быть объединены с помощью коммутатора (аналогично MPP).

Примеры: NEC SX-4/SX-5, линия векторно-конвейерных компьютеров CRAY: от CRAY-1, CRAY J90/T90, CRAY SV1, серия Fujitsu VPP и др.

Модель программирования: Эффективное программирование подразумевает векторизацию циклов (для достижения разумной производительности одного процессора) и их распараллеливание (для одновременной загрузки нескольких процессоров одним приложением).

Кластерные системы

Архитектура: Набор рабочих станций (или даже ПК) общего назначения, используется в качестве дешевого варианта массивно-параллельного компьютера. Для связи узлов используется одна из стандартных сетевых технологий (Fast/Gigabit Ethernet, Myrinet и др.) на базе шинной архитектуры или коммутатора.

При объединении в кластер компьютеров разной мощности или разной архитектуры, говорят о **гетерогенных** (неоднородных) кластерах.

Узлы кластера могут одновременно использоваться в качестве пользовательских рабочих станций. В случае, когда это не нужно, узлы могут быть существенно облегчены и/или установлены в стойку.

Примеры: NT-кластер в NCSA, Beowulf-кластеры, кластеры МГУ и СПбГУ и др.

Операционная система: Используются стандартные для рабочих станций ОС, чаще всего, свободно распространяемые - Linux/FreeBSD,

вместе со специальными средствами поддержки параллельного программирования и распределения нагрузки.

Модель программирования: Программирование, как правило, в рамках модели передачи сообщений (чаще всего - MPI).

Излагаемый ниже материал см. в аудиторных лекциях.

Организация схем коммутации в МВС с общей памятью

- Типовая архитектура МВС с общей памятью
- Особенности организации средств коммутации. Примеры
- Организация средств коммутации в архитектуре “Butterfly”

Организация схем коммутации в МВС с распределенной памятью

- Типовая архитектура
- Архитектура со связями через общую шину
- Архитектура со связями через несколько шин
- Архитектура со связями через многоступенчатый переключатель
- Структура типа гиперкуб

Архитектура систем со смешенной организацией памяти

Пример типовой архитектуры