Общие требования, предъявляемые к современным компьютерам

•Общие требования, предъявляемые к современным компьютерам
- •Отношение стоимость/производительность
- •Надежность и отказоустойчивость
- •Масштабируемость
- •Совместимость и мобильность программного обеспечения

Отношение стоимость/производительность

Появление любого нового направления в вычислительной технике определяется требованиями компьютерного рынка. Поэтому у разработчиков компьютеров нет одной единственной цели. Большая универсальная вычислительная машина (мейнфрейм) или суперкомпьютер стоят дорого. Для достижения поставленных целей при проектировании высокопроизводительных конструкций приходится игнорировать стоимостные характеристики. Суперкомпьютеры фирмы Cray Research и высокопроизводительные мейнфреймы компании IBM относятся именно к этой категории компьютеров. Другим крайним примером может служить низкостоимостная конструкция, где производительность принесена в жертву для достижения низкой стоимости. К этому направлению относятся персональные компьютеры различных клонов IBM PC. Между этими двумя крайними направлениями находятся конструкции, основанные на отношении стоимость/ производительность, в которых разработчики находят баланс между стоимостными параметрами и производительностью. Типичными примерами такого рода компьютеров являются миникомпьютеры и рабочие станции.

Для сравнения различных компьютеров между собой обычно используются стандартные методики измерения производительности. Эти методики позволяют разработчикам и пользователям использовать полученные в результате испытаний количественные показатели для оценки тех или иных технических решений, и в конце концов именно производительность и стоимость дают пользователю рациональную основу для решения вопроса, какой компьютер выбрать.

Надежность и отказоустойчивость

Важнейшей характеристикой вычислительных систем является надежность. Повышение надежности основано на принципе предотвращения неисправностей путем снижения интенсивности отказов и сбоев за счет применения электронных схем и компонентов с высокой и сверхвысокой степенью интеграции, снижения уровня помех, облегченных режимов работы схем, обеспечение тепловых режимов их работы, а также за счет совершенствования методов сборки аппаратуры.

Отказоустойчивость - это такое свойство вычислительной системы, которое обеспечивает ей, как логической машине, возможность продолжения действий, заданных программой, после возникновения неисправностей. Введение отказоустойчивости требует избыточного аппаратного и программного обеспечения. Направления, связанные с предотвращением неисправностей и с отказоустойчивостью, - основные в проблеме надежности. Концепции параллельности и отказоустойчивости вычислительных систем естественным образом связаны между собой, поскольку в обоих случаях требуются дополнительные функциональные компоненты. Поэтому, собственно, на параллельных вычислительных системах достигается как наиболее высокая производительность, так и, во многих случаях, очень высокая надежность. Имеющиеся ресурсы избыточности в параллельных системах могут гибко использоваться как для повышения производительности, так и для повышения надежности. Структура многопроцессорных и многомашинных систем приспособлена к автоматической реконфигурации и обеспечивает возможность продолжения работы системы после возникновения неисправностей.

Следует помнить, что понятие надежности включает не только аппаратные средства, но и программное обеспечение. Главной целью повышения надежности систем является целостность хранимых в них данных.

Масштабируемость

Масштабируемость представляет собой возможность наращивания числа и мощности процессоров, объемов оперативной и внешней памяти и других ресурсов вычислительной системы. Масштабируемость должна обеспечиваться архитектурой и конструкцией компьютера, а также соответствующими средствами программного обеспечения.

Добавление каждого нового процессора в действительно масштабируемой системе должно давать прогнозируемое увеличение производительности и пропускной способности при приемлемых затратах. Одной из основных задач при построении масштабируемых систем является минимизация стоимости расширения компьютера и упрощение планирования. В идеале добавление процессоров к системе должно приводить к линейному росту ее производительности. Однако это не всегда так. Потери производительности могут возникать, например, при недостаточной пропускной способности шин из-за возрастания трафика между процессорами и основной памятью, а также между памятью и устройствами ввода/вывода. В действительности реальное увеличение производительности трудно оценить заранее, поскольку оно в значительной степени зависит от динамики поведения прикладных задач.

Возможность масштабирования системы определяется не только архитектурой аппаратных средств, но зависит от заложенных свойств программного обеспечения. Масштабируемость программного обеспечения затрагивает все его уровни от простых механизмов передачи сообщений до работы с такими сложными объектами как мониторы транзакций и вся среда прикладной системы. В частности, программное обеспечение должно минимизировать трафик межпроцессорного обмена, который может препятствовать линейному росту производительности системы. Аппаратные средства (процессоры, шины и устройства ввода/вывода) являются только частью масштабируемой архитектуры, на которой программное обеспечение может обеспечить предсказуемый рост производительности. Важно понимать, что простой переход, например, на более мощный процессор может привести к перегрузке других компонентов системы. Это означает, что действительно масштабируемая система должна быть сбалансирована по всем параметрам.

Совместимость и мобильность программного обеспечения

Концепция программной совместимости впервые в широких масштабах была применена разработчиками системы IBM/360. Основная задача при проектировании всего ряда моделей этой системы заключалась в создании такой архитектуры, которая была бы одинаковой с точки зрения пользователя для всех моделей системы независимо от цены и производительности каждой из них. Огромные преимущества такого подхода, позволяющего сохранять существующий задел программного обеспечения при переходе на новые (как правило, более производительные) модели были быстро оценены как производителями компьютеров, так и пользователями и начиная с этого времени практически все фирмы-поставщики компьютерного оборудования взяли на вооружение эти принципы, поставляя серии совместимых компьютеров. Следует заметить однако, что со временем даже самая передовая архитектура неизбежно устаревает и возникает потребность внесения радикальных изменений архитектуру и способы организации вычислительных систем.

В настоящее время одним из наиболее важных факторов, определяющих современные тенденции в развитии информационных технологий, является ориентация компаний-поставщиков компьютерного оборудования на рынок прикладных программных средств. Это объясняется прежде всего тем, что для конечного пользователя в конце концов важно программное обеспечение, позволяющее решить его задачи, а не выбор той или иной аппаратной платформы. Переход от однородных сетей программно совместимых компьютеров к построению неоднородных сетей, включающих компьютеры разных фирм-производителей, в корне изменил и точку зрения на саму сеть: из сравнительно простого средства обмена

информацией она превратилась в средство интеграции отдельных ресурсов - мощную распределенную вычислительную систему, каждый элемент которой (сервер или рабочая станция) лучше всего соответствует требованиям конкретной прикладной задачи.

Этот переход выдвинул ряд новых требований. Прежде всего такая вычислительная среда должна позволять гибко менять количество и состав аппаратных средств и программного обеспечения в соответствии с меняющимися требованиями решаемых задач. Во-вторых, она должна обеспечивать возможность запуска одних и тех же программных систем на различных аппаратных платформах, т.е. обеспечивать мобильность программного обеспечения. В третьих, эта среда должна гарантировать возможность применения одних и тех же человеко-машинных интерфейсов на всех компьютерах, входящих в неоднородную сеть. В условиях жесткой конкуренции производителей аппаратных платформ и программного обеспечения сформировалась концепция открытых систем, представляющая собой совокупность стандартов на различные компоненты вычислительной среды, предназначенных для обеспечения мобильности программных средств в рамках неоднородной, распределенной вычислительной системы.

Одним из вариантов моделей открытой среды является модель OSE (Open System Environment), предложенная комитетом IEEE POSIX. На основе этой модели национальный институт стандартов и технологии США выпустил документ "Application Portability Profile (APP). The U.S. Government's Open System Environment Profile OSE/1 Version 2.0", который определяет рекомендуемые для федеральных учреждений США спецификации в области информационных технологий, обеспечивающие мобильность системного и прикладного программного обеспечения. Все ведущие производители компьютеров и программного обеспечения в США в настоящее время придерживаются требований этого документа.

Многопроцессорные системы

- Многопроцессорные системы
  - Классификация систем параллельной обработки данных
  - Многопроцессорные системы с общей памятью
  - Многопроцессорные системы с локальной памятью и многомашинные системы

Классификация систем параллельной обработки данных

На протяжении всей истории развития вычислительной техники делались попытки найти какую-то общую классификацию, под которую подпадали бы все возможные направления развития компьютерных архитектур. Ни одна из таких классификаций не могла охватить все разнообразие разрабатываемых архитектурных решений и не выдерживала испытания временем. Тем не менее в научный оборот попали и широко используются ряд терминов, которые полезно знать не только разработчикам, но и пользователям компьютеров.

Любая вычислительная система (будь то супер-ЭВМ или персональный компьютер) достигает своей наивысшей производительности благодаря использованию высокоскоростных элементов и параллельному выполнению большого числа операций. Именно возможность параллельной работы различных устройств системы (работы с перекрытием) является основой ускорения основных операций.

Параллельные ЭВМ часто подразделяются по классификации Флинна на машины типа SIMD (Single Instruction Multiple Data - с одним потоком команд при множественном потоке данных) и MIMD (Multiple Instruction Multiple Data - с множественным потоком команд при множественном потоке данных). Как и любая другая, приведенная выше классификация несовершенна: существуют машины прямо в нее не попадающие, имеются также важные признаки, которые в этой классификации не учтены. В частности, к машинам типа SIMD часто относят векторные процессоры, хотя их высокая производительность зависит от другой формы параллелизма - конвейерной организации машины. Многопроцессорные векторные системы, типа Cray Y-MP, состоят из нескольких векторных процессоров и поэтому могут быть названы MSIMD (Multiple SIMD).

Классификация Флинна не делает различия по другим важным для вычислительных моделей характеристикам, например, по уровню "зернистости" параллельных вычислений и методам синхронизации.

Можно выделить четыре основных типа архитектуры систем параллельной обработки:

1) Конвейерная и векторная обработка.

Основу конвейерной обработки составляет раздельное выполнение некоторой операции в несколько этапов (за несколько ступеней) с передачей данных одного этапа следующему. Производительность при этом возрастает благодаря тому, что одновременно на различных ступенях конвейера выполняются несколько операций. Конвейеризация эффективна только тогда, когда загрузка конвейера близка к полной, а скорость подачи новых операндов соответствует максимальной производительности конвейера. Если происходит задержка, то параллельно будет выполняться меньше операций и суммарная производительность снизится. Векторные операции обеспечивают идеальную возможность полной загрузки вычислительного конвейера.

При выполнении векторной команды одна и та же операция применяется ко всем элементам вектора (или чаще всего к соответствующим элементам пары векторов). Для настройки конвейера на выполнение конкретной операции может потребоваться некоторое установочное время, однако затем операнды могут поступать в конвейер с максимальной скоростью, допускаемой возможностями памяти. При этом не возникает пауз ни в связи с выборкой новой команды, ни в связи с определением ветви вычислений при условном переходе. Таким образом, главный принцип вычислений на векторной машине состоит в выполнении некоторой элементарной операции или комбинации из нескольких элементарных операций, которые должны повторно применяться к некоторому блоку данных. Таким операциям в исходной программе соответствуют небольшие компактные циклы.

2) Машины типа SIMD. Машины типа SIMD состоят из большого числа идентичных процессорных элементов, имеющих собственную память. Все процессорные элементы в такой машине выполняют одну и ту же программу. Очевидно, что такая машина, составленная из большого числа процессоров, может обеспечить очень высокую производительность только на тех задачах, при решении которых все процессоры могут делать одну и ту же работу. Модель вычислений для машины SIMD очень похожа на модель вычислений для векторного процессора: одиночная операция выполняется над большим блоком данных.
В отличие от ограниченного конвейерного функционирования векторного процессора, матричный процессор (синоним для большинства SIMD-машин) может быть значительно более гибким. Обрабатывающие элементы таких процессоров - это универсальные программируемые ЭВМ, так что задача, решаемая параллельно, может быть достаточно сложной и содержать ветвления. Обычное проявление этой вычислительной модели в исходной программе примерно такое же, как и в случае векторных операций: циклы на элементах массива, в которых значения, вырабатываемые на одной итерации цикла, не используются на другой итерации цикла.

Модели вычислений на векторных и матричных ЭВМ настолько схожи, что эти ЭВМ часто обсуждаются как эквивалентные.

3) Машины типа MIMD. Термин "мультипроцессор" покрывает большинство машин типа MIMD и (подобно тому, как термин "матричный процессор" применяется к машинам типа SIMD) часто используется в качестве синонима для машин типа MIMD. В мультипроцессорной системе каждый процессорный элемент (ПЭ) выполняет свою программу достаточно независимо от других процессорных элементов. Процессорные элементы, конечно, должны как-то связываться друг с другом, что делает необходимым более подробную классификацию машин типа MIMD. В мультипроцессорах с общей памятью (сильносвязанных мультипроцессорах) имеется память данных и команд, доступная всем ПЭ. С общей памятью ПЭ связываются с помощью общей шины или сети обмена. В противоположность этому варианту в слабосвязанных многопроцессорных системах (машинах с локальной памятью) вся память делится между процессорными элементами и каждый блок памяти доступен только связанному с ним процессору. Сеть обмена связывает процессорные элементы друг с другом.

Базовой моделью вычислений на MIMD-мультипроцессоре является совокупность независимых процессов, эпизодически обращающихся к разделяемым данным. Существует большое количество вариантов этой модели. На одном конце спектра - модель распределенных вычислений, в которой программа делится на довольно большое число параллельных задач, состоящих из множества подпрограмм. На другом конце спектра - модель потоковых вычислений, в которых каждая операция в программе может рассматриваться как отдельный процесс. Такая операция ждет своих входных данных (операндов), которые должны быть переданы ей другими процессами. По их получении операция выполняется, и полученное значение передается тем процессам, которые в нем нуждаются. В потоковых моделях вычислений с большим и средним уровнем гранулярности, процессы содержат большое число операций и выполняются в потоковой манере.

4) Многопроцессорные машины с SIMD-процессорами.

Многие современные супер-ЭВМ представляют собой многопроцессорные системы, в которых в качестве процессоров используются векторные процессоры или процессоры типа SIMD. Такие машины относятся к машинам класса MSIMD.

Языки программирования и соответствующие компиляторы для машин типа MSIMD обычно обеспечивают языковые конструкции, которые позволяют программисту описывать "крупнозернистый" параллелизм. В пределах каждой задачи компилятор автоматически векторизует подходящие циклы. Машины типа MSIMD, как можно себе представить, дают возможность использовать лучший из этих двух принципов декомпозиции: векторные операции ("мелкозернистый" параллелизм) для тех частей программы, которые подходят для этого, и гибкие возможности MIMD-архитектуры для других частей программы.

Многопроцессорные системы за годы развития вычислительной техники претерпели ряд этапов своего развития. Исторически первой стала осваиваться технология SIMD. Однако в

настоящее время наметился устойчивый интерес к архитектурам MIMD. Этот интерес главным образом определяется двумя факторами:

1. Архитектура MIMD дает большую гибкость: при наличии адекватной поддержки со стороны аппаратных средств и программного обеспечения MIMD может работать как однопользовательская система, обеспечивая высокопроизводительную обработку данных для одной прикладной задачи, как многопрограммная машина, выполняющая множество задач параллельно, и как некоторая комбинация этих возможностей.
2. Архитектура MIMD может использовать все преимущества современной микропроцессорной технологии на основе строгого учета соотношения стоимость/производительность. В действительности практически все современные многопроцессорные системы строятся на тех же микропроцессорах, которые можно найти в персональных компьютерах, рабочих станциях и небольших однопроцессорных серверах.

Одной из отличительных особенностей многопроцессорной вычислительной системы является сеть обмена, с помощью которой процессоры соединяются друг с другом или с памятью. Модель обмена настолько важна для многопроцессорной системы, что многие характеристики производительности и другие оценки выражаются отношением времени обработки к времени обмена, соответствующим решаемым задачам. Существуют две основные модели межпроцессорного обмена: одна основана на передаче сообщений, другая - на использовании общей памяти. В многопроцессорной системе с общей памятью один процессор осуществляет запись в конкретную ячейку, а другой процессор производит считывание из этой ячейки памяти. Чтобы обеспечить согласованность данных и синхронизацию процессов, обмен часто реализуется по принципу взаимно исключающего доступа к общей памяти методом "почтового ящика".

В архитектурах с локальной памятью непосредственное разделение памяти невозможно. Вместо этого процессоры получают доступ к совместно используемым данным посредством передачи сообщений по сети обмена. Эффективность схемы коммуникаций зависит от протоколов обмена, основных сетей обмена и пропускной способности памяти и каналов обмена.

Часто, и притом необоснованно, в машинах с общей памятью и векторных машинах затраты на обмен не учитываются, так как проблемы обмена в значительной степени скрыты от программиста. Однако накладные расходы на обмен в этих машинах имеются и определяются конфликтами шин, памяти и процессоров. Чем больше процессоров добавляется в систему, тем больше процессов соперничают при использовании одних и тех же данных и шины, что приводит к состоянию насыщения. Модель системы с общей памятью очень удобна для программирования и иногда рассматривается как высокоуровневое средство оценки влияния обмена на работу системы, даже если основная система в действительности реализована с применением локальной памяти и принципа передачи сообщений.

В сетях с коммутацией каналов и в сетях с коммутацией пакетов по мере возрастания требований к обмену следует учитывать возможность перегрузки сети. Здесь межпроцессорный обмен связывает сетевые ресурсы: каналы, процессоры, буферы сообщений. Объем передаваемой информации может быть сокращен за счет тщательной функциональной декомпозиции задачи и тщательного диспетчирования выполняемых функций.

Таким образом, существующие MIMD-машины распадаются на два основных класса в зависимости от количества объединяемых процессоров, которое определяет и способ организации памяти и методику их межсоединений.

К первой группе относятся машины с общей (разделяемой) основной памятью, объединяющие до нескольких десятков (обычно менее 32) процессоров. Сравнительно небольшое количество процессоров в таких машинах позволяет иметь одну централизованную общую память и объединить процессоры и память с помощью одной шины. При наличии у процессоров кэш-памяти достаточного объема высокопроизводительная шина и общая память могут удовлетворить обращения к памяти, поступающие от нескольких процессоров. Поскольку имеется единственная память с одним и тем же временем доступа, эти машины иногда называются UMA (Uniform Memory Access). Такой способ организации со сравнительно

небольшой разделяемой памятью в настоящее время является наиболее популярным. Структура подобной системы представлена на рис. 10.1.

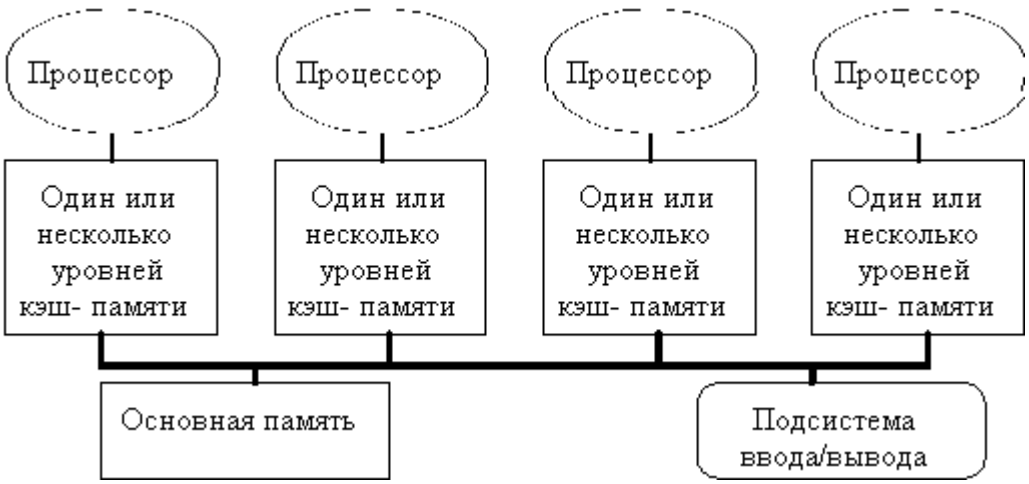


Рис. 10.1. Типовая архитектура мультипроцессорной системы с общей памятью.

Вторую группу машин составляют крупномасштабные системы с распределенной памятью. Для того чтобы поддерживать большое количество процессоров приходится распределять основную память между ними, в противном случае полосы пропускания памяти просто может не хватить для удовлетворения запросов, поступающих от очень большого числа процессоров. Естественно при таком подходе также требуется реализовать связь процессоров между собой. На рис. 10.2 показана структура такой системы.

С ростом числа процессоров просто невозможно обойти необходимость реализации модели распределенной памяти с высокоскоростной сетью для связи процессоров. С быстрым ростом производительности процессоров и связанным с этим ужесточением требования увеличения полосы пропускания памяти, масштаб систем (т.е. число процессоров в системе), для которых требуется организация распределенной памяти, уменьшается, также как и уменьшается число процессоров, которые удается поддерживать на одной разделяемой шине и общей памяти.

Распределение памяти между отдельными узлами системы имеет два главных преимущества. Во-первых, это эффективный с точки зрения стоимости способ увеличения полосы пропускания памяти, поскольку большинство обращений могут выполняться параллельно к локальной памяти в каждом узле. Во-вторых, это уменьшает задержку обращения (время доступа) к локальной памяти. Эти два преимущества еще больше сокращают количество процессоров, для которых архитектура с распределенной памятью имеет смысл.

Обычно устройства ввода/вывода, также как и память, распределяются по узлам и в действительности узлы могут состоять из небольшого числа (2-8) процессоров, соединенных между собой другим способом. Хотя такая кластеризация нескольких процессоров с памятью и сетевой интерфейс могут быть достаточно полезными с точки зрения эффективности в стоимостном выражении, это не очень существенно для понимания того, как такая машина работает, поэтому мы пока остановимся на системах с одним процессором на узел. Основная разница в архитектуре, которую следует выделить в машинах с распределенной памятью заключается в том, как осуществляется связь и какова логическая модель памяти.

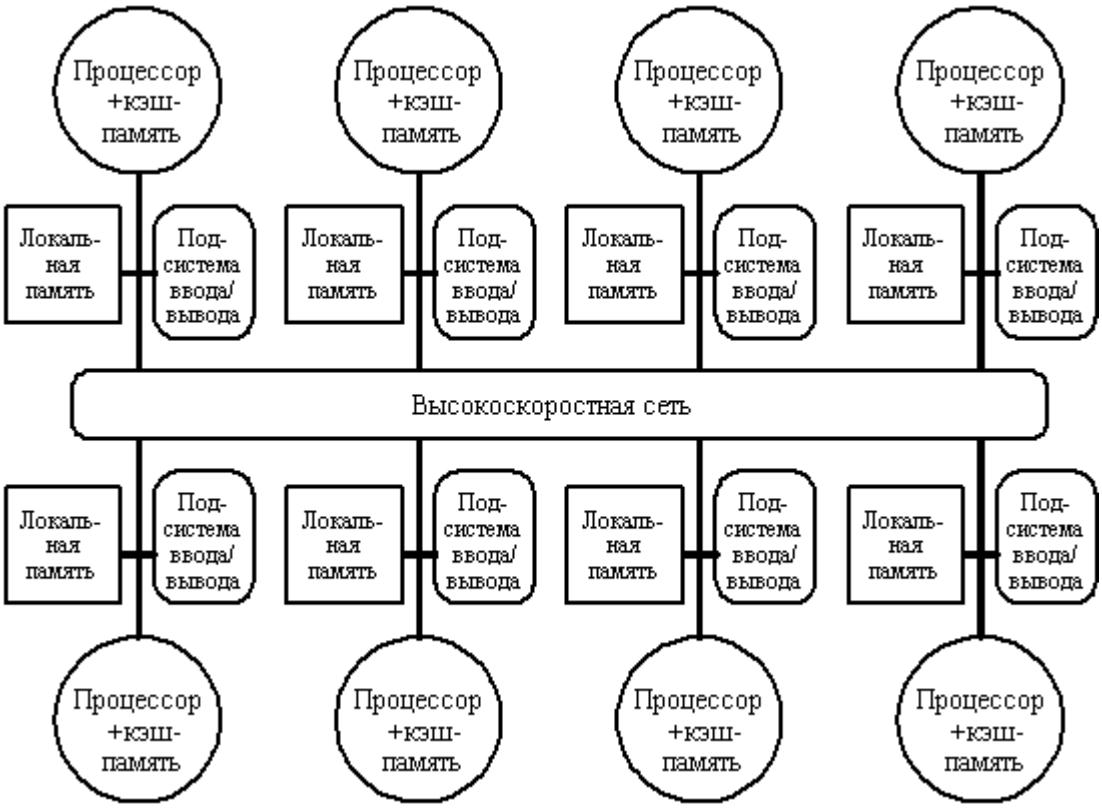


Рис. 10.2. Типовая архитектура машины с распределенной памятью.

Модели связи и архитектуры памяти

Как уже было отмечено, любая крупномасштабная многопроцессорная система должна использовать множество устройств памяти, которые физически распределяются вместе с процессорами. Имеется две альтернативных организации адресации этих устройств памяти и связанных с этим два альтернативных метода для передачи данных между процессорами. Физически отдельные устройства памяти могут адресоваться как логически единое адресное пространство, что означает, что любой процессор может выполнять обращения к любым ячейкам памяти, предполагая, что он имеет соответствующие права доступа. Такие машины называются машинами с распределенной разделяемой (общей) памятью (DSM - distributed shared memory), масштабируемые архитектуры с разделяемой памятью, а иногда NUMA's - Non-Uniform Memory Access, поскольку время доступа зависит от расположения ячейки в памяти.

В альтернативном случае, адресное пространство состоит из отдельных адресных пространств, которые логически не связаны и доступ к которым не может быть осуществлен аппаратно другим процессором. В таком примере каждый модуль процессор-память представляет собой отдельный компьютер, поэтому такие системы называются многомашинными (multicomputers).

С каждой из этих организаций адресного пространства связан свой механизм обмена. Для машины с единым адресным пространством это адресное пространство может быть использовано для обмена данными посредством операций загрузки и записи. Поэтому эти машины и получили название машин с разделяемой (общей) памятью. Для машин с множеством адресных пространств обмен данными должен использовать другой механизм: передачу сообщений между процессорами; поэтому эти машины часто называют машинами с передачей сообщений.

Каждый из этих механизмов обмена имеет свои преимущества. Для обмена в общей памяти это включает:

- Совместимость с хорошо понятными используемыми как в однопроцессорных, так и маломасштабных многопроцессорных системах, механизмами, которые используют для обмена общую память.
- Простота программирования, когда модели обмена между процессорами сложные или динамически меняются во время выполнения. Подобные преимущества упрощают конструирование компилятора.

- Более низкая задержка обмена и лучшее использование полосы пропускания при обмене малыми порциями данных.
- Возможность использования аппаратно управляемого кэширования для снижения частоты удаленного обмена, допускающая кэширование всех данных как разделяемых, так и неразделяемых.

Основные преимущества обмена с помощью передачи сообщений являются:

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

Конечно, требуемая модель обмена может быть надстроена над аппаратной моделью, которая использует любой из этих механизмов. Поддержка передачи сообщений над разделяемой памятью, естественно, намного проще, если предположить, что машины имеют адекватные полосы пропускания. Основные трудности возникают при работе с сообщениями, которые могут быть неправильно выровнены и сообщениями произвольной длины в системе памяти, которая обычно ориентирована на передачу выровненных блоков данных, организованных как блоки кэш-памяти. Эти трудности можно преодолеть либо с небольшими потерями производительности программным способом, либо существенно без потерь при использовании небольшой аппаратной поддержки.

Построение механизмов реализации разделяемой памяти над механизмом передачи сообщений намного сложнее. Без предполагаемой поддержки со стороны аппаратуры все обращения к разделяемой памяти потребуют привлечения операционной системы как для обеспечения преобразования адресов и защиты памяти, так и для преобразования обращений к памяти в посылку и прием сообщений. Поскольку операции загрузки и записи обычно работают с небольшим объемом данных, то большие накладные расходы по поддержанию такого обмена делают невозможной чисто программную реализацию.

При оценке любого механизма обмена критичными являются три характеристики производительности:

1. *Полоса пропускания:* в идеале полоса пропускания механизма обмена будет ограничена полосами пропускания процессора, памяти и системы межсоединений, а не какими-либо аспектами механизма обмена. Связанные с механизмом обмена накладные расходы (например, длина межпроцессорной связи) прямо воздействуют на полосу пропускания.
2. *Задержка:* в идеале задержка должна быть настолько мала, насколько это возможно. Для ее определения критичны накладные расходы аппаратуры и программного обеспечения, связанные с инициированием и завершением обмена.
3. *Упрятывание задержки:* насколько хорошо механизм скрывает задержку путем перекрытия обмена с вычислениями или с другими обменами.

Каждый из этих параметров производительности воздействует на характеристики обмена. В частности, задержка и полоса пропускания могут меняться в зависимости от размера элемента данных. В общем случае, механизм, который одинаково хорошо работает как с небольшими, так и с большими объемами данных будет более гибким и эффективным.

Таким образом, отличия разных машин с распределенной памятью определяются моделью памяти и механизмом обмена. Исторически машины с распределенной памятью первоначально были построены с использованием механизма передачи сообщений, поскольку это было очевидно проще и многие разработчики и исследователи не верили, что единое адресное пространство можно построить и в машинах с распределенной памятью. С недавнего времени модели обмена с общей памятью действительно начали поддерживаться практически в каждой разработанной машине (характерным примером могут служить системы с симметричной мультипроцессорной обработкой). Хотя машины с централизованной общей памятью, построенные на базе общей шины все еще доминируют в терминах размера компьютерного рынка, долговременные технические тенденции направлены на использование преимуществ распределенной памяти даже в машинах умеренного размера. Как мы увидим, возможно наиболее важным вопросом, который встает при создании машин с распределенной памятью, является вопрос о кэшировании и когерентности кэш-памяти.

Многопроцессорные системы с общей памятью

Требования, предъявляемые современными процессорами к полосе пропускания памяти можно существенно сократить путем применения больших многоуровневых кэшей. Тогда, если эти требования снижаются, то несколько процессоров смогут разделять доступ к одной и той же памяти. Начиная с 1980 года эта идея, подкрепленная широким распространением микропроцессоров, стимулировала многих разработчиков на создание небольших мультипроцессоров, в которых несколько процессоров разделяют одну физическую память, соединенную с ними с помощью разделяемой шины. Из-за малого размера процессоров и заметного сокращения требуемой полосы пропускания шины, достигнутого за счет возможности реализации достаточно большой кэш-памяти, такие машины стали исключительно эффективными по стоимости. В первых разработках подобного рода машин удавалось разместить весь процессор и кэш на одной плате, которая затем вставлялась в заднюю панель, с помощью которой реализовывалась шинная архитектура. Современные конструкции позволяют разместить до четырех процессоров на одной плате. На рис. 10.1 показана схема именно такой машины.

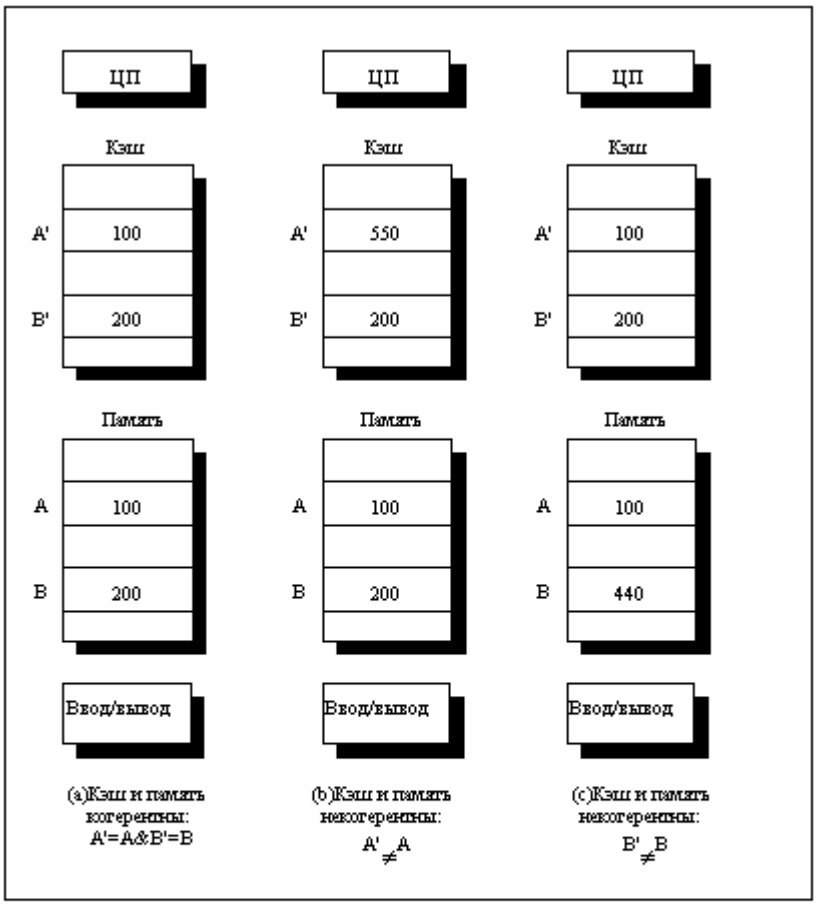
В такой машине кэши могут содержать как разделяемые, так и частные данные. Частные данные - это данные, которые используются одним процессором, в то время как разделяемые данные используются многими процессорами, по существу обеспечивая обмен между ними. Когда кэшируется элемент частных данных, их значение переносится в кэш для сокращения среднего времени доступа, а также требуемой полосы пропускания. Поскольку никакой другой процессор не использует эти данные, этот процесс идентичен процессу для однопроцессорной машины с кэш-памятью. Если кэшируются разделяемые данные, то разделяемое значение реплицируется и может содержаться в нескольких кэшах. Кроме сокращения задержки доступа и требуемой полосы пропускания такая репликация данных способствует также общему сокращению количества обменов. Однако кэширование разделяемых данных вызывает новую проблему: когерентность кэш-памяти.

Мультипроцессорная когерентность кэш-памяти

Проблема, о которой идет речь, возникает из-за того, что значение элемента данных в памяти, хранящееся в двух разных процессорах, доступно этим процессорам только через их индивидуальные кэши. На рис. 10.3 показан простой пример, иллюстрирующий эту проблему.

Проблема когерентности памяти для мультипроцессоров и устройств ввода/вывода имеет много аспектов. Обычно в малых мультипроцессорах используется аппаратный механизм, называемый протоколом, позволяющий решить эту проблему. Такие протоколы называются протоколами когерентности кэш-памяти. Существуют два класса таких протоколов:

1. Протоколы на основе справочника (directory based). Информация о состоянии блока физической памяти содержится только в одном месте, называемом справочником (физически справочник может быть распределен по узлам системы). Этот подход будет рассмотрен в разд. 10.3.
2. Протоколы наблюдения (snooping). Каждый кэш, который содержит копию данных некоторого блока физической памяти, имеет также соответствующую копию служебной информации о его состоянии. Централизованная система записей отсутствует. Обычно кэши расположены на общей (разделяемой) шине и контроллеры всех кэшей наблюдают за шиной (просматривают ее) для определения того, не содержат ли они копию соответствующего блока.



A' и B' - кэшированные копии элементов A и B в основной памяти

a) Когерентное состояние кэша и основной памяти.

b) Предполагается использование кэш-памяти с отложенным обратным копированием, когда ЦП записывает значение 550 в ячейку A. В результате A' содержит новое значение, а в основной памяти осталось старое значение 100. При попытке вывода A из памяти будет получено старое значение.

c) Подсистема ввода/вывода вводит в ячейку памяти B новое значение 440, а в кэш-памяти осталось старое значение B.

Рис. 10.3. Иллюстрация проблемы когерентности кэш-памяти

В мультипроцессорных системах, использующих микропроцессоры с кэш-памятью, подсоединенные к централизованной общей памяти, протоколы наблюдения приобрели популярность, поскольку для опроса состояния кэшей они могут использовать заранее существующее физическое соединение - шину памяти.

Неформально, проблема когерентности памяти состоит в необходимости гарантировать, что любое считывание элемента данных возвращает последнее по времени записанное в него значение. Это определение не совсем корректно, поскольку невозможно требовать, чтобы операция считывания мгновенно видела значение, записанное в этот элемент данных некоторым другим процессором. Если, например, операция записи на одном процессоре предшествует операции чтения той же ячейки на другом процессоре в пределах очень короткого интервала времени, то невозможно гарантировать, что чтение вернет записанное значение данных, поскольку в этот момент времени записываемые данные могут даже не покинуть процессор. Вопрос о том, когда точно записываемое значение должно быть доступно процессору, выполняющему чтение, определяется выбранной моделью согласованного (непротиворечивого) состояния памяти и связан с реализацией синхронизации параллельных вычислений. Поэтому с целью упрощения предположим, что мы требуем только, чтобы записанное операцией записи значение было доступно операции чтения, возникшей немного позже записи и что операции записи данного процессора всегда видны в порядке их выполнения.

С этим простым определением согласованного состояния памяти мы можем гарантировать когерентность путем обеспечения двух свойств:

1. Операция чтения ячейки памяти одним процессором, которая следует за операцией записи в ту же ячейку памяти другим процессором получит записанное значение, если операции чтения и записи достаточно отделены друг от друга по времени.
2. Операции записи в одну и ту же ячейку памяти выполняются строго последовательно (иногда говорят, что они сериализованы): это означает, что две подряд идущие операции записи в одну и ту же ячейку памяти будут наблюдаться другими процессорами именно в том порядке, в котором они появляются в программе процессора, выполняющего эти операции записи.

Первое свойство очевидно связано с определением когерентного (согласованного) состояния памяти: если бы процессор всегда бы считывал только старое значение данных, мы сказали бы, что память некогерентна.

Необходимость строго последовательного выполнения операций записи является более тонким, но также очень важным свойством. Представим себе, что строго последовательное выполнение операций записи не соблюдается. Тогда процессор P1 может записать данные в ячейку, а затем в эту ячейку выполнит запись процессор P2. Строго последовательное выполнение операций записи гарантирует два важных следствия для этой последовательности операций записи. Во-первых, оно гарантирует, что каждый процессор в машине в некоторый момент времени будет наблюдать запись, выполняемую процессором P2. Если последовательность операций записи не соблюдается, то может возникнуть ситуация, когда какой-нибудь процессор будет наблюдать сначала операцию записи процессора P2, а затем операцию записи процессора P1, и будет хранить это записанное P1 значение неограниченно долго. Более тонкая проблема возникает с поддержанием разумной модели порядка выполнения программ и когерентности памяти для пользователя: представьте, что третий процессор постоянно читает ту же самую ячейку памяти, в которую записывают процессоры P1 и P2; он должен наблюдать сначала значение, записанное P1, а затем значение, записанное P2. Возможно он никогда не сможет увидеть значения, записанного P1, поскольку запись от P2 возникла раньше чтения. Если он даже видит значение, записанное P1, он должен видеть значение, записанное P2, при последующем чтении. Подобным образом любой другой процессор, который может наблюдать за значениями, записываемыми как P1, так и P2, должен наблюдать идентичное поведение. Простейший способ добиться таких свойств заключается в строгом соблюдении порядка операций записи, чтобы все записи в одну и ту же ячейку могли наблюдаться в том же самом порядке. Это свойство называется последовательным выполнением (сериализацией) операций записи (write serialization). Вопрос о том, когда процессор должен увидеть значение, записанное другим процессором достаточно сложен и имеет заметное воздействие на производительность, особенно в больших машинах.

Альтернативные протоколы

Имеются две методики поддержания описанной выше когерентности. Один из методов заключается в том, чтобы гарантировать, что процессор должен получить исключительные права доступа к элементу данных перед выполнением записи в этот элемент данных. Этот тип протоколов называется протоколом *записи с аннулированием (write ivalidate protocol),* поскольку при выполнении записи он аннулирует другие копии. Это наиболее часто используемый протокол как в схемах на основе справочников, так и в схемах наблюдения. Исключительное право доступа гарантирует, что во время выполнения записи не существует никаких других копий элемента данных, в которые можно писать или из которых можно читать: все другие кэшированные копии элемента данных аннулированы. Чтобы увидеть, как такой протокол обеспечивает когерентность, рассмотрим операцию записи, вслед за которой следует операция чтения другим процессором. Поскольку запись требует исключительного права доступа, любая копия, поддерживаемая читающим процессором должна быть аннулирована (в соответствии с названием протокола). Таким образом, когда возникает операция чтения, произойдет промах кэш-памяти, который вынуждает выполнить выборку новой копии данных. Для выполнения операции записи мы можем потребовать, чтобы процессор имел *достоверную (valid)* копию данных в своей кэш-памяти прежде, чем выполнять в нее запись. Таким образом, если оба процессора попытаются записать в один и тот же элемент

данных одновременно, один из них выиграет состязание у второго (мы вскоре увидим, как принять решение, кто из них выиграет) и вызывает аннулирование его копии. Другой процессор для завершения своей операции записи должен сначала получить новую копию данных, которая теперь уже должна содержать обновленное значение.

Альтернативой протоколу записи с аннулированием является обновление всех копий элемента данных в случае записи в этот элемент данных. Этот тип протокола называется протоколом *записи с обновлением (write update protocol)* или протоколом *записи с трансляцией (write broadcast protocol).* Обычно в этом протоколе для снижения требований к полосе пропускания полезно отслеживать, является ли слово в кэш-памяти разделяемым объектом, или нет, а именно, содержится ли оно в других кэшах. Если нет, то нет никакой необходимости обновлять другой кэш или транслировать в него обновленные данные.

Разница в производительности между протоколами записи с обновлением и с аннулированием определяется тремя характеристиками:

1. Несколько последовательных операций записи в одно и то же слово, не перемежающихся операциями чтения, требуют нескольких операций трансляции при использовании протокола записи с обновлением, но только одной начальной операции аннулирования при использовании протокола записи с аннулированием.
2. При наличии многословных блоков в кэш-памяти каждое слово, записываемое в блок кэша, требует трансляции при использовании протокола записи с обновлением, в то время как только первая запись в любое слово блока нуждается в генерации операции аннулирования при использовании протокола записи с аннулированием. Протокол записи с аннулированием работает на уровне блоков кэш-памяти, в то время как протокол записи с обновлением должен работать на уровне отдельных слов (или байтов, если выполняется запись байта).
3. Задержка между записью слова в одном процессоре и чтением записанного значения другим процессором обычно меньше при использовании схемы записи с обновлением, поскольку записанные данные немедленно транслируются в процессор, выполняющий чтение (предполагается, что этот процессор имеет копию данных). Для сравнения, при использовании протокола записи с аннулированием в процессоре, выполняющим чтение, сначала произойдет аннулирование его копии, затем будет производиться чтение данных и его приостановка до тех пор, пока обновленная копия блока не станет доступной и не вернется в процессор.

Эти две схемы во многом похожи на схемы работы кэш-памяти со сквозной записью и с записью с обратным копированием. Также как и схема задержанной записи с обратным копированием требует меньшей полосы пропускания памяти, так как она использует преимущества операций над целым блоком, протокол записи с аннулированием обычно требует менее тяжелого трафика, чем протокол записи с обновлением, поскольку несколько записей в один и тот же блок кэш-памяти не требуют трансляции каждой записи. При сквозной записи память обновляется почти мгновенно после записи (возможно с некоторой задержкой в буфере записи). Подобным образом при использовании протокола записи с обновлением другие копии обновляются так быстро, насколько это возможно. Наиболее важное отличие в производительности протоколов записи с аннулированием и с обновлением связано с характеристиками прикладных программ и с выбором размера блока.

Основы реализации

Ключевым моментом реализации в многопроцессорных системах с небольшим числом процессоров как схемы записи с аннулированием, так и схемы записи с обновлением данных, является использование для выполнения этих операций механизма шины. Для выполнения операции обновления или аннулирования процессор просто захватывает шину и транслирует по ней адрес, по которому должно производиться обновление или аннулирование данных. Все процессоры непрерывно наблюдают за шиной, контролируя появляющиеся на ней адреса. Процессоры проверяют не находится ли в их кэш-памяти адрес, появившийся на шине. Если это так, то соответствующие данные в кэше либо аннулируются, либо обновляются в зависимости от используемого протокола. Последовательный порядок обращений, присущий

шине, обеспечивает также строго последовательное выполнение операций записи, поскольку когда два процессора конкурируют за выполнение записи в одну и ту же ячейку, один из них должен получить доступ к шине раньше другого. Один процессор, получив доступ к шине, вызовет необходимость обновления или аннулирования копий в других процессорах. В любом случае, все записи будут выполняться строго последовательно. Один из выводов, который следует сделать из анализа этой схемы заключается в том, что запись в разделяемый элемент данных не может закончиться до тех пор, пока она не захватит доступ к шине.

В дополнение к аннулированию или обновлению соответствующих копий блока кэш-памяти, в который производилась запись, мы должны также разместить элемент данных, если при записи происходит промах кэш-памяти. В кэш-памяти со сквозной записью последнее значение элемента данных найти легко, поскольку все записываемые данные всегда посылаются также и в память, из которой последнее записанное значение элемента данных может быть выбрано (наличие буферов записи может привести к некоторому усложнению).

Однако для кэш-памяти с обратным копированием задача нахождения последнего значения элемента данных сложнее, поскольку это значение скорее всего находится в кэше, а не в памяти. В этом случае используется та же самая схема наблюдения, что и при записи: каждый процессор наблюдает и контролирует адреса, помещаемые на шину. Если процессор обнаруживает, что он имеет модифицированную ("грязную") копию блока кэш-памяти, то именно он должен обеспечить пересылку этого блока в ответ на запрос чтения и вызвать отмену обращения к основной памяти. Поскольку кэши с обратным копированием предъявляют меньшие требования к полосе пропускания памяти, они намного предпочтительнее в мультипроцессорах, несмотря на некоторое увеличение сложности. Поэтому далее мы рассмотрим вопросы реализации кэш-памяти с обратным копированием.

Для реализации процесса наблюдения могут быть использованы обычные теги кэша. Более того, упоминавшийся ранее *бит достоверности (valid bit),* позволяет легко реализовать аннулирование. Промахи операций чтения, вызванные либо аннулированием, либо каким-нибудь другим событием, также не сложны для понимания, поскольку они просто основаны на возможности наблюдения. Для операций записи мы хотели бы также знать, имеются ли другие кэшированные копии блока, поскольку в случае отсутствия таких копий, запись можно не посылать на шину, что сокращает время на выполнение записи, а также требуемую полосу пропускания.

Чтобы отследить, является ли блок разделяемым, мы можем ввести дополнительный бит состояния (shared), связанный с каждым блоком, точно также как это делалось для битов достоверности (valid) и модификации (modified или dirty) блока. Добавив бит состояния, определяющий является ли блок разделяемым, мы можем решить вопрос о том, должна ли запись генерировать операцию аннулирования в протоколе с аннулированием, или операцию трансляции при использовании протокола с обновлением. Если происходит запись в блок, находящийся в состоянии "разделяемый" при использовании протокола записи с аннулированием, кэш формирует на шине операцию аннулирования и помечает блок как *частный* (private). Никаких последующих операций аннулирования этого блока данный процессор посылать больше не будет. Процессор с *исключительной* (exclusive) копией блока кэш-памяти обычно называется "владельцем" (owner) блока кэш-памяти.

При использовании протокола записи с обновлением, если блок находится в состоянии "разделяемый", то каждая запись в этот блок должна транслироваться. В случае протокола с аннулированием, когда посылается операция аннулирования, состояние блока меняется с "разделяемый" на "неразделяемый" (или "частный"). Позже, если другой процессор запросит этот блок, состояние снова должно измениться на "разделяемый". Поскольку наш наблюдающий кэш видит также все промахи, он знает, когда этот блок кэша запрашивается другим процессором, и его состояние должно стать "разделяемый".

Поскольку любая транзакция на шине контролирует адресные теги кэша, потенциально это может приводить к конфликтам с обращениями к кэшу со стороны процессора. Число таких потенциальных конфликтов можно снизить применением одного из двух методов: дублированием тегов, или использованием многоуровневых кэшей с "охватом" (inclusion), в которых уровни, находящиеся ближе к процессору являются поднабором уровней, находящихся дальше от него. Если теги дублируются, то обращения процессора и наблюдение за шиной

могут выполняться параллельно. Конечно, если при обращении процессора происходит промах, он должен будет выполнять арбитраж с механизмом наблюдения для обновления обоих наборов тегов. Точно также, если механизм наблюдения за шиной находит совпадающий тег, ему будет нужно проводить арбитраж и обращаться к обоим наборам тегов кэша (для выполнения аннулирования или обновления бита "разделяемый"), возможно также и к массиву данных в кэше, для нахождения копии блока. Таким образом, при использовании схемы дублирования тегов процессор должен приостановиться только в том случае, если он выполняет обращение к кэшу в тот же самый момент времени, когда механизм наблюдения обнаружил копию в кэше. Более того, активность механизма наблюдения задерживается только когда кэш имеет дело с промахом.

Наимено-вание	Тип протокола	Стратегия записи в память	Уникальные свойства	Применение
Одиночная запись	Запись с аннулированием	Обратное копирование при первой записи	Первый описанный в литературе протокол наблюдения	-
Synapse N+1	Запись с аннулированием	Обратное копирование	Точное состояние, где "владельцем является память"	Машины Synapse Первые машины с когерентной кэш-памятью
Berkely	Запись с аннулированием	Обратное копирование	Состояние "разделяемый"	Машина SPUR университета Berkely
Illinois	Запись с аннулированием	Обратное копирование	Состояние "приватный";	Серии Power и Challenge
			может	компании
			передавать	Silicon
			данные из	Graphics
			любого кэша	
"Firefly"	Запись с транс-ляцией	Обратное копирование для "приватных" блоков и сквозная запись для "разделяемых"	Обновление памяти во время трансляции	SPARCcenter 2000

Рис. 10.4. Примеры протоколов наблюдения

Если процессор использует многоуровневый кэш со свойствами охвата, тогда каждая строка в основном кэше имеется и во вторичном кэше. Таким образом, активность по наблюдению может быть связана с кэшем второго уровня, в то время как большинство активностей процессора могут быть связаны с первичным кэшем. Если механизм наблюдения получает попадание во вторичный кэш, тогда он должен выполнять арбитраж за первичный кэш, чтобы обновить состояние и возможно найти данные, что обычно будет приводить к приостановке процессора. Такое решение было принято во многих современных системах, поскольку многоуровневый кэш позволяет существенно снизить требований к полосе пропускания. Иногда может быть даже полезно дублировать теги во вторичном кэше, чтобы

еще больше сократить количество конфликтов между активностями процессора и механизма наблюдения.

В реальных системах существует много вариаций схем когерентности кэша, в зависимости от того используется ли схема на основе аннулирования или обновления, построена ли кэш-память на принципах сквозной или обратной записи, когда происходит обновление, а также имеет ли место состояние "владения" и как оно реализуется. На рис. 10.4 представлены несколько протоколов с наблюдением и некоторые машины, которые используют эти протоколы.

Многопроцессорные системы с локальной памятью и многомашинные системы

Существуют два различных способа построения крупномасштабных систем с распределенной памятью. Простейший способ заключается в том, чтобы исключить аппаратные механизмы, обеспечивающие когерентность кэш-памяти, и сосредоточить внимание на создании масштабируемой системы памяти. Несколько компаний разработали такого типа машины. Наиболее известным примером такой системы является компьютер T3D компании Cray Research. В этих машинах память распределяется между узлами (процессорными элементами) и все узлы соединяются между собой посредством того или иного типа сети. Доступ к памяти может быть локальным или удаленным. Специальные контроллеры, размещаемые в узлах сети, могут на основе анализа адреса обращения принять решение о том, находятся ли требуемые данные в локальной памяти данного узла, или размещаются в памяти удаленного узла. В последнем случае контроллеру удаленной памяти посылается сообщение для обращения к требуемым данным.

Чтобы обойти проблемы когерентности, разделяемые (общие) данные не кэшируются. Конечно, с помощью программного обеспечения можно реализовать некоторую схему кэширования разделяемых данных путем их копирования из общего адресного пространства в локальную память конкретного узла. В этом случае когерентностью памяти также будет управлять программное обеспечение. Преимуществом такого подхода является практически минимальная необходимая поддержка со стороны аппаратуры, хотя наличие, например, таких возможностей как блочное (групповое) копирование данных было бы весьма полезным. Недостатком такой организации является то, что механизмы программной поддержки когерентности подобного рода кэш-памяти компилятором весьма ограничены. Существующая в настоящее время методика в основном подходит для программ с хорошо структурированным параллелизмом на уровне программного цикла.

Машины с архитектурой, подобной Cray T3D, называют процессорами (машинами) с массовым параллелизмом (MPP Massively Parallel Processor). К машинам с массовым параллелизмом предъявляются взаимно исключающие требования. Чем больше объем устройства, тем большее число процессоров можно расположить в нем, тем длиннее каналы передачи управления и данных, а значит и меньше тактовая частота. Происшедшее возрастание нормы массивности для больших машин до 512 и даже 64К процессоров обусловлено не ростом размеров машины, а повышением степени интеграции схем, позволившей за последние годы резко повысить плотность размещения элементов в устройствах. Топология сети обмена между процессорами в такого рода системах может быть различной. На рис. 10.5 приведены характеристики сети обмена для некоторых коммерческих MPP.

Для построения крупномасштабных систем альтернативой рассмотренному в предыдущем разделе протоколу наблюдения может служить протокол на основе справочника, который отслеживает состояние кэшей. Такой подход предполагает, что логически единый справочник хранит состояние каждого блока памяти, который может кэшироваться. В справочнике обычно содержится информация о том, в каких кэшах имеются копии данного блока, модифицировался ли данный блок и т.д. В существующих реализациях этого направления справочник размещается рядом с памятью. Имеются также протоколы, в которых часть информации размещается в кэш-памяти. Положительной стороной хранения всей информации в едином справочнике является простота протокола, связанная с тем, что вся необходимая информация сосредоточена в одном месте. Недостатком такого рода справочников является его размер, который пропорционален общему объему памяти, а не

размеру кэш-памяти. Это не составляет проблемы для машин, состоящих, например, из нескольких сотен процессоров, поскольку связанные с реализацией такого справочника накладные расходы можно преодолеть. Но для машин большего размера необходима методика, позволяющая эффективно масштабировать структуру справочника.

Фирма	Название	Количество узлов	Базовая топология	Разрядность связи (бит)	Частота синхронизации (Мгц)	Пиковая полоса пропускания связи (Мбайт/с)	Общая полоса пропускания (Мбайт/с)	Год выпуска
Thinking Machines	CM-2	1024-4096	12-мерный куб	1	7	1	1024	1987
nCube	nCube/ten	1-1024	10-мерный куб	1	10	1.2	640	1987
Intel	iPSC/2	16-128	7-мерный куб	1	16	2	345	1988
Maspar	MP-1216	32-512	2-мерная сеть+ступенчатая Omega	1	25	3	1300	1989
Intel	Delta	540	2-мерная сеть	16	40	40	640	1991
Thinking Machines	CM-5	32-2048	многоступенчатое толстое дерево	4	40	20	10240	1991
Meiko	CS-2	2-1024	многоступенчатое толстое дерево	8	70	50	50000	1992
Intel	Paragon	4-1024	2-мерная сеть	16	100	200	6400	1992
Cray Research	T3D	16-1024	3-мерный тор	16	150	300	19200	1993

Рис. 10.5. Характеристики межсоединений некоторых коммерческих MPP

В частности, чтобы предотвратить появление узкого горла в системе, связанного с единым справочником, можно распределить части этого справочника вместе с устройствами распределенной локальной памяти. Таким образом можно добиться того, что обращения к разным справочникам (частям единого справочника) могут выполняться параллельно, точно также как обращения к локальной памяти в распределенной памяти могут выполняться

параллельно, существенно увеличивая общую полосу пропускания памяти. В распределенном справочнике сохраняется главное свойство подобных схем, заключающееся в том, что состояние любого разделяемого блока данных всегда находится во вполне определенном известном месте. На рис. 10.6 показан общий вид подобного рода машины с распределенной памятью. Вопросы детальной реализации протоколов когерентности памяти для таких машин выходят за рамки настоящего обзора.

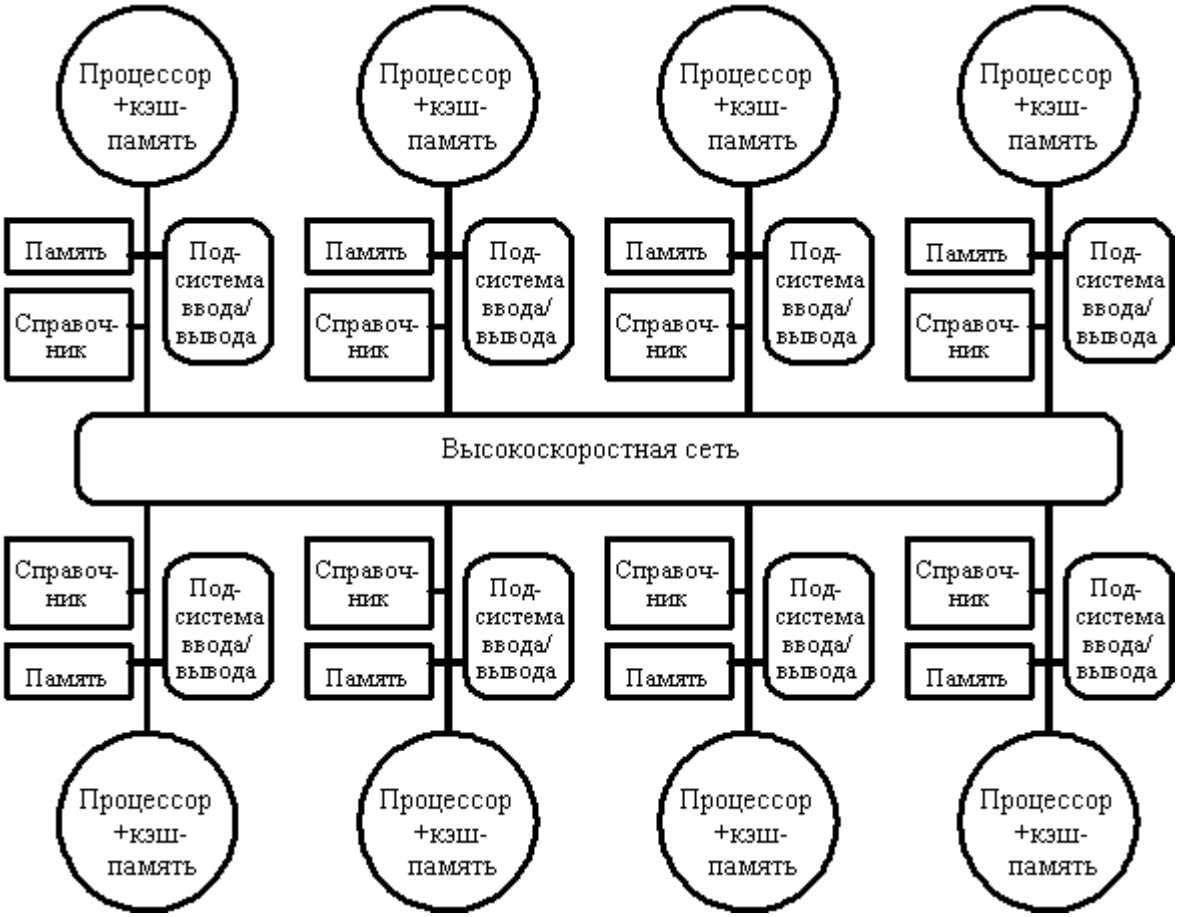


Рис. 10.6. Архитектура системы с распределенной внешней памятью и распределенным по узлам справочником

Системы высокой готовности и отказоустойчивые системы

- Системы высокой готовности и отказоустойчивые системы
  - Основные определения
  - Подсистемы внешней памяти высокой готовности
  - Требования, предъявляемые к системам высокой готовности
    - Конфигурации систем высокой готовности
    - Требования начальной установки системы
    - Требования к системному программному обеспечению
    - Требования высокой готовности к прикладному программному обеспечению
    - Требования к сетевой организации и к коммуникациям
  - "Кластеризация" как способ обеспечения высокой готовности системы
    - Базовая модель VAX/VMS кластеров
    - Критерии оценки кластеров Gartner Group
    - Кластеры Alpha/OSF компании DEC
    - UNIX-кластеры компании IBM
    - Кластеры AT&T GIS
    - Кластеры Sequent Computer Systems
    - Системы высокой готовности Hewlett-Packard
    - Кластерные решения Sun Microsystems
    - Отказоустойчивые решения Data General

Основные определения

Одной из основных проблем построения вычислительных систем во все времена остается задача обеспечения их продолжительного функционирования. Эта задача имеет три составляющих: *надежность*, *готовность* и *удобство обслуживания*. Все эти три составляющих предполагают, в первую очередь, борьбу с неисправностями системы, порождаемыми отказами и сбоями в ее работе. Эта борьба ведется по всем трем направлениям, которые взаимосвязаны и применяются совместно.

Повышение надежности основано на принципе предотвращения неисправностей путем снижения интенсивности отказов и сбоев за счет применения электронных схем и компонентов с высокой и сверхвысокой степенью интеграции, снижения уровня помех, облегченных режимов работы схем, обеспечения тепловых режимов их работы, а также за счет совершенствования методов сборки аппаратуры. Единицей измерения надежности является среднее время наработки на отказ (MTBF - Mean Time Between Failure).

Повышение готовности предполагает подавление в определенных пределах влияния отказов и сбоев на работу системы с помощью средств контроля и коррекции ошибок, а также средств автоматического восстановления вычислительного процесса после проявления неисправности, включая аппаратурную и программную избыточность, на основе которой реализуются различные варианты отказоустойчивых архитектур. Повышение готовности - есть способ борьбы за снижение времени простоя системы. Единицей измерения здесь является коэффициент готовности, который определяет вероятность пребывания системы в работоспособном состоянии в любой произвольный момент времени. Статистически коэффициент готовности определяется как MTBF/(MTBF+MTTR), где MTTR (Mean Time To Repair) - среднее время восстановления (ремонта), т.е. среднее время между моментом обнаружения неисправности и моментом возврата системы к полноценному функционированию.

Таким образом, основные эксплуатационные характеристики системы существенно зависят от удобства ее обслуживания, в частности от ремонтопригодности, контролепригодности и т.д.

В последние годы в литературе по вычислительной технике все чаще употребляется термин "системы высокой готовности", "системы высокой степени готовности", "системы с

высоким коэффициентом готовности". Все эти термины по существу являются синонимами, однако как и многие термины в области вычислительной техники, термин "высокая готовность" понимается по-разному отдельными поставщиками и потребителями вычислительных систем. Совершенно аналогично, некоторые слова, связанные с термином "высокая готовность", такие, например, как "кластеризация", также употребляются в различных значениях. Важно иметь стандартный набор определений для того, чтобы предложения различных поставщиков можно было сравнивать между собой на основе одинаковых терминов.

Ниже приведены общепринятые в настоящее время определения, которые мы будем использовать для различных типов систем, свойством которых является та или иная форма снижения планового и непланового времени простоя:

- *Высокая Готовность (High Availability).* Настоящие конструкции с высоким коэффициентом готовности для минимизации планового и непланового времени простоя используют обычную компьютерную технологию. При этом конфигурация системы обеспечивает ее быстрое восстановление после обнаружения неисправности, для чего в ряде мест используются избыточные аппаратные и программные средства. Длительность задержки, в течение которой программа, отдельный компонент или система простаивает, может находиться в диапазоне от нескольких секунд до нескольких часов, но более часто в диапазоне от 2 до 20 минут. Обычно системы высокой готовности хорошо масштабируются, предлагая пользователям большую гибкость, чем другие типы избыточности.
- *Эластичность к отказам (Fault Resiliency).* Ряд поставщиков компьютерного оборудования делит весь диапазон систем высокой готовности на две части, при этом в верхней его части оказываются системы эластичные к отказам. Ключевым моментом в определении эластичности к отказам является более короткое время восстановления, которое позволяет системе быстро откатиться назад после обнаружения неисправности.
- *Устойчивость к отказам (Fault Tolerance).* Отказоустойчивые системы имеют в своем составе избыточную аппаратуру для всех функциональных блоков, включая процессоры, источники питания, подсистемы ввода/вывода и подсистемы дисковой памяти. Если соответствующий функциональный блок неправильно функционирует, всегда имеется горячий резерв. В наиболее продвинутых отказоустойчивых системах избыточные аппаратные средства можно использовать для распараллеливания обычных работ. Время восстановления после обнаружения неисправности для переключения отказавших компонентов на избыточные для таких систем обычно меньше одной секунды.
- *Непрерывная готовность (Continuous Availability).* Вершиной линии отказоустойчивых систем являются системы, обеспечивающие непрерывную готовность. Продукт с непрерывной готовностью, если он работает корректно, устраняет любое время простоя как плановое, так и неплановое. Разработка такой системы охватывает как аппаратные средства, так и программное обеспечение и позволяет проводить модернизацию (upgrade) и обслуживание в режиме on-line. Дополнительным требованием к таким системам является отсутствие деградации в случае отказа. Время восстановления после отказа не превышает одной секунды.
- *Устойчивость к стихийным бедствияи (Disaster Tolerance).* Широкий ряд продуктов и услуг связан с обеспечением устойчивости к стихийным бедствиям. Иногда устойчивость к стихийным бедствиям рассматривается в контексте систем высокой готовности. Смысл этого термина в действительности означает возможность рестарта или продолжения операций на другой площадке, если основное месторасположение системы оказывается в нерабочем состоянии из-за наводнения, пожара или землетрясения. В простейшем случае, продукты, устойчивые к стихийным бедствиям, могут просто представлять собой резервные компьютеры, расположенные вне основного местоположения системы, сконфигурированные по спецификациям пользователя и доступные для использования в случае стихийного бедствия на основной площадке. В более сложных случаях устойчивость к стихийным бедствиям может означать полное (зеркальное) дублирование системы вне основного местоположения, позволяющее принять на себя работу немедленно после отказа системы на основной площадке.

Все упомянутые типы систем высокой готовности имеют общую цель - минимизацию времени простоя. Имеется два типа времени простоя компьютера: плановое и неплановое. Минимизация каждого из них требует различной стратегии и технологии. Плановое время простоя обычно включает время, принятое руководством, для проведения работ по модернизации системы и для ее обслуживания. Неплановое время простоя является результатом отказа системы или компонента. Хотя системы высокой готовности возможно больше ассоциируются с минимизацией неплановых простоев, они оказываются также полезными для уменьшения планового времени простоя.

Возможно наибольшим виновником планового времени простоя является резервное копирование данных. Некоторые конфигурации дисковых подсистем высокой готовности, особенно системы с зеркальными дисками, позволяют производить резервное копирование данных в режиме on-line. Следующим источником снижения планового времени простоя является организация работ по обновлению (модернизации) программного обеспечения. Сегодня некоторые отказоустойчивые системы и все системы с непрерывной готовностью позволяют производить модернизацию программного обеспечения в режиме on-line. Некоторые поставщики систем высокой готовности также обещают такие же возможности в течение ближайших нескольких лет.

В общем случае, неплановое время простоя прежде всего снижается за счет использования надежных частей, резервных магистралей или избыточного оборудования. Однако даже в этом случае система может требовать достаточно большого планового времени простоя.

Специальное программное обеспечение является существенной частью систем высокой готовности. При обнаружении неисправности системы оно обеспечивает управление конфигурацией аппаратных средств и программного обеспечения, а также в случае необходимости процедурами начальной установки, и перестраивает где надо структуры данных.

Высокая готовность не дается бесплатно. Общая стоимость подобных систем складывается из трех составляющих: начальной стоимости системы, издержек планирования и реализации, а также системных накладных расходов.

Для реализации системы высокой готовности пользователи должны в начале закупить собственно систему (или системы), включающую один или несколько процессоров в зависимости от требуемой вычислительной мощности и предполагаемой конфигурации, дополнительное программное обеспечение и дополнительное дисковое пространство.

Чтобы реализовать конфигурацию системы высокой готовности наиболее эффективным способом, особенно при использовании кластерных схем, требуется достаточно большое предварительное планирование. Например, чтобы иметь возможность переброски критичного приложения в случае отказа одного процессора на другой, пользователи должны определить, какие приложения являются наиболее критичными, проанализировать все возможные отказы и составить подробные планы восстановления на все случаи отказов.

Накладные расходы систем высокой готовности связаны с необходимостью поддержки довольно сложных программных продуктов, обеспечивающих высокую готовность. Для обеспечения дублирования записей на зеркальные диски в случае отсутствия специальных, предназначенных для этих целей процессоров, требуется поддержка дополнительной внешней памяти.

Стоимость системы высокой готовности в значительной степени зависит от выбранной конфигурации и ее возможностей. Ниже приведена некоторая информация, позволяющая грубо оценить различные типы избыточности.

Высокая Готовность. Дополнительная стоимость систем высокой готовности меняется в пределах от 10 до 100 процентов, обычно стремясь к середине этого диапазона. Дополнительная стоимость системы высокой готовности зависит от той степени, с которой пользователь способен использовать резервную систему для обработки своих приложений. Стоимость системы высокой готовности может реально превысить 100 процентов за счет программного обеспечения и необходимой начальной установки в случае применения резервной системы, которая не используется ни для чего другого. Однако обычно резервная система может быть использована для решения некритичных задач, значительно снижая стоимость.

Высокая эластичность. Дополнительная стоимость систем высокой эластичности к отказам, принадлежащих к верхнему уровню диапазона систем высокой готовности, лежит в пределах от 20 до 100 процентов, снова обычно стремясь к середине этого диапазона. Схемы высокой эластичности более сложны и предполагают более высокую стоимость планирования и большие накладные расходы, чем системы, принадлежащие нижнему уровню диапазона систем высокой готовности. В некоторых случаях однако, пользователь может в большей степени использовать общие процессорные ресурсы, тем самым уменьшая общую стоимость.

Непрерывная готовность. Надбавка к стоимости для систем с непрерывной готовностью находится в диапазоне от 20 до 100 или более процентов и обычно приближается к верхнему пределу этого диапазона. Программное обеспечение для обеспечения режима непрерывной готовности более сложное, чем для систем, обеспечивающих высокую эластичность к отказам. Большинство компонентов системы, такие как процессоры, источники питания, контроллеры и кабели, должны дублироваться, а иногда и троироваться. Пользователи систем непрерывной готовности, как и пользователи высоко эластичных к отказам систем, имеют возможность использовать весь набор ресурсов системы большую часть времени, по сравнению с пользователями более простых систем, принадлежащих нижнему уровню диапазона систем высокой готовности.

Устойчивость к стихийным бедствиям. Надбавка к стоимости для систем, устойчивым к стихийным бедствиям, может сильно варьироваться. С одной стороны, она может составлять, например, только несколько процентов стоимости системы при резервировании времени запасного компьютера, находящегося вне основной площадки. С другой стороны, стоимость системы может увеличиться в несколько раз, если необходимо обеспечить действительно быстрое переключение на другую систему, находящуюся на удаленной площадке с помощью высокоскоростных сетевых средств. Большинство предложений по системам, устойчивым к стихийным бедствиям, требуют существенного объема планирования.

Для того, чтобы снизить стоимость системы, следует тщательно оценивать действительно необходимый уровень готовности (т.е. осуществлять выбор между высокой готовностью, устойчивостью к отказам и/или устойчивостью к стихийным бедствиям) и вкладывать деньги только за обеспечение безопасности наиболее критичных для деятельности компании приложений и данных.

Подсистемы внешней памяти высокой готовности

Первым шагом на пути обеспечения высокой готовности является защита наиболее важной части системы, а именно - данных. Разные типы конфигураций избыточной внешней памяти обеспечивают разную степень защиты данных и имеют разную стоимость.

Имеются три основных типа подсистем внешней памяти с высокой готовностью. Для своей реализации они используют технологию Избыточных Массивов Дешевых Дисков (RAID - Redundant Arrays of Inexpensive Disks). Наиболее часто используются следующий решения (более подробно об уровнях RAID см. разд. 9.3.2): RAID уровня 1 или зеркальные диски, RAID уровня 3 с четностью и RAID уровня 5 с распределенной четностью. Эти три типа внешней памяти в общем случае имеют практически почти мгновенное время восстановления в случае отказа. Кроме того, подобные устройства иногда позволяют пользователям смешивать и подбирать типы RAID в пределах одного дискового массива. В общем случае дисковые массивы представляются прикладной задаче как один диск.

Технология RAID уровня 1 (или зеркалирования дисков) основана на применении двух дисков так, что в случае отказа одного из них, для работы может быть использована копия, находящаяся на дополнительном диске. Программные средства поддержки зеркальных дисков обеспечивают запись всех данных на оба диска. Недостатком организации зеркальных дисков является удвоение стоимости аппаратных средств и незначительное увеличение времени записи, поскольку данные должны быть записаны на оба диска. Положительные стороны этого подхода включают возможность обеспечения резервного копирования в режиме on-line, а также замену дисков в режиме on-line, что существенно снижает плановое время простоя. Как правило, структура устройств с зеркальными дисками устраняет также единственность точки

отказа, поскольку для подключения обоих дисков обычно предусматриваются два отдельных кабеля, а также два отдельных контроллера ввода/вывода.

В массивах RAID уровня 3 предусматривается использование одного дополнительного дискового накопителя, обеспечивающего хранение информации о четности (контрольной суммы) данных, записываемых на каждые два или четыре диска. Если один из дисков в массиве отказывает, информация о четности вместе с данными, находящимися на других оставшихся дисках, позволяет реконструировать данные, находившиеся на отказавшем накопителе.

Массив RAID уровня 5 является комбинацией RAID уровня 0, в котором данные расщепляются для записи на несколько дисков, и RAID уровня 3, в которых имеется один дополнительный диск. В RAID уровня 5 полезная информация четырех дисков и контрольная информация распределяется по всем пяти дискам так, что при отказе одного из них, оставшиеся четыре обеспечивают считывание необходимых данных. Методика расщепления данных позволяет также существенно увеличить скорость ввода/вывода при передаче больших объемов данных.

Диапазон возможных конструкций современных дисковых массивов достаточно широк. Он простираются от простых подсистем без многих дополнительных возможностей до весьма изощренных дисковых подсистем, которые позволяют пользователям смешивать и подбирать уровни RAID внутри одного устройства. Наиболее мощные дисковые подсистемы могут также содержать в своем составе процессоры, которые разгружают основную систему от выполнения рутинных операций ввода/вывода, форматирования дисков, защиты от ошибок и выполнения алгоритмов RAID. Большинство дисковых массивов снабжаются двумя портами, что позволяет пользователям подключать их к двум различным системам.

Добавка к стоимости дисковой подсистемы для организации зеркальных дисков стремится к 100%, поскольку требуемые диски должны дублироваться в избыточной конфигурации 1:1. Дополнительная стоимость дисков для RAID уровней 3 и 5 составляет либо 33% при наличии диска четности для каждых двух дисков, либо более часто 20% при наличии диска четности для каждых четырех накопителей. Кроме того, следует учитывать стоимость специального программного обеспечения, а также стоимость организации самого массива. Некоторые компании предлагают программные средства для организации зеркальных дисков при использовании обычных дисковых подсистем. Другие предлагают возможности зеркальной организации в подсистемах RAID, специально изготовленных для этих целей. Подсистемы, использующие RAID уровней 3 и 5, отличаются по стоимости в зависимости от своих возможностей.

Реализация внешней памяти высокой готовности может приводить также к увеличению системных накладных расходов. Например, основной процессор системы вынужден обрабатывать две операции при каждой записи информации на зеркальные диски, если эти диски не являются частью зеркального дискового массива, который имеет свои собственные средства обработки. Однако наиболее сложные дисковые массивы позволяют снизить накладные расходы за счет использования процессоров ввода/вывода, являющихся частью аппаратуры дискового массива.

В настоящее время для устройств внешней памяти характерна тенденция уменьшения соотношения стоимости на единицу емкости памяти (1, 0.5 и менее долларов за мегабайт). Эта тенденция делает сегодняшние избыточные решения по внешней памяти даже менее дорогими, по сравнению со стоимостью подсистем с обычными дисками, выпускавшимися только год назад. Поэтому использование подсистемы RAID очень быстро становится одним из базовых требований обычных систем, а не специальным свойством систем высокой готовности.

Требования, предъявляемые к системам высокой готовности

В настоящее время одним из ключевых требований пользователей UNIX-систем является возможность их наращивания с целью обеспечения более высокой степени готовности. Главными характеристиками систем высокой готовности по сравнению со стандартными системами являются пониженная частота отказов и более быстрый переход к нормальному режиму функционирования после возникновения неисправности посредством

быстрого восстановления приложений и сетевых сессий до того состояния, в котором они находились в момент отказа системы. Следует отметить, что во многих случаях пользователей вполне может устроить даже небольшое время простоя в обмен на меньшую стоимость системы высокой готовности по сравнению со значительно более высокой стоимостью обеспечения режима непрерывной готовности.

Конфигурации систем высокой готовности

Конфигурации систем высокой готовности, предлагаемые современной компьютерной промышленностью, простираются в широком диапазоне от "простейших" жестких схем, обеспечивающих дублирование основной системы отдельно стоящим горячим резервом в соотношении 1:1, до весьма свободных кластерных схем, позволяющих одной системе подхватить работу любой из нескольких систем в кластере в случае их неисправности.

Термин "кластеризация" на сегодня в компьютерной промышленности имеет много различных значений. Строгое определение могло бы звучать так: "реализация объединения машин, представляющегося единым целым для операционной системы, системного программного обеспечения, прикладных программ и пользователей". Машины, кластеризованные вместе таким способом могут при отказе одного процессора очень быстро перераспределить работу на другие процессоры внутри кластера. Это, возможно, наиболее важная задача многих поставщиков систем высокой готовности. Имеются несколько поставщиков, которые называют свои системы высокой готовности "кластерами" или "простыми кластерами", однако на сегодняшний день реально доступны только несколько кластеров, которые подпадают под строгое определение (см. ниже).

Современные конструкции систем высокой готовности предполагают использование горячего резерва (Fail-Over), включая переключение прикладных программ и пользователей на другую машину с гарантией отсутствия потерь или искажений данных во время отказа и переключения. В зависимости от свойств системы, некоторые или все эти процессы могут быть автоматизированы.

Системы высокой готовности связаны со своими резервными системами посредством очень небольшого программного демона "сердечный пульс", который позволяет резервной системе управлять основной системой или системами, которые она резервирует. Когда "пульс" пропадает, кластер переходит в режим переключения на резервную систему.

Такое переключение может выполняться вручную или автоматически, и имеется несколько уровней автоматизации этого процесса. Например, в некоторых случаях пользователи инструктируются о том, что они должны выйти и снова войти в систему. В других случаях переключение осуществляется более прозрачным для пользователя способом: он только должен подождать в течение короткого периода времени. Иногда пользователь может делать выбор между ручным и автоматическим переключением. В некоторых системах пользователи могут продолжить работу после переключения именно с той точки, где они находились во время отказа. В других случаях их просят повторить последнюю транзакцию.

Резервная система не обязательно должна полностью повторять систему, которую она резервирует (конфигурации систем могут отличаться). Это позволяет в ряде случаев сэкономить деньги за счет резервирования большой системы или систем с помощью системы меньшего размера и предполагает либо снижение производительности в случае отказа основной системы, либо переключение на резервную систему только критичных для жизнедеятельности организации приложений.

Следует добавить, что одни пользователи предпочитают не выполнять никаких приложений на резервной машине, хотя другие наоборот стараются немного нагрузить резервный сервер в кластере. Возможность выбора конфигурации системы с помощью процедур начальной установки дает пользователям большую гибкость, позволяя постепенно использовать весь заложенный в системе потенциал.

Требования начальной установки системы

Большинство систем высокой готовности требуют включения в свой состав процедур начальной установки (System Setup), обеспечивающих конфигурацию кластера для подобающего выполнения процедур переключения, необходимых в случае отказа. Пользователи могут запрограммировать "скрипты" начальной установки самостоятельно или попросить системного интегратора или поставщика проделать эту работу. В зависимости от того, насколько сложна начальная установка системы, и в зависимости от типа системы, с которой мигрирует пользователь, написание "скриптов", которые управляют действиями системы высокой готовности в случае отказа, может занять от одного - двух дней до нескольких недель или даже месяцев для опытных программистов. Многие поставщики обеспечивают несколько стандартных "скриптов" начальной установки. Кроме того, некоторые из них предоставляют сервисные услуги по начальной установке конфигурации, которые включают программирование сценариев переключения на горячий резерв в случае отказа, а также осуществляют работу с заказчиком по написанию или модификации "скриптов". Пользователи могут самостоятельно создавать "скрипты", однако для реализации подобающей конфигурации требуется высококвалифицированный программист - знаток UNIX и C.

Время простоя при переключении системы на резервную для систем высокой готовности может меняться в диапазоне от нескольких секунд до 20-40 и более минут. Процедура переключения на резерв включает в себя следующие этапы: резервная машина обнаруживает отказ основной и затем следует предписаниям скрипта, который вероятнее всего включает перезапуск системы, передачу адресов пользователей, получение и запуск необходимых приложений, а также выполнение определенных шагов по обеспечению корректного состояния данных. Время восстановления зависит главным образом от того, насколько быстро вторая машина сможет получить и запустить приложения, а также от того, насколько быстро операционная система и приложения, такие как базы данных или мониторы транзакций, смогут получить приведенные в порядок данные.

В общем случае аппаратное переключение на резерв занимает по порядку величины одну - две минуты, а система перезагружается за следующие одну - две минуты. В большинстве случаев от 5 до 20 минут требуется на то, чтобы получить и запустить приложение с полностью восстановленными данными. В противном случае пользователи инструктируются о необходимости заново ввести последнюю транзакцию.

Накладные системные расходы зависят от типа используемой системы и от сложности процедур ее начальной установки. Для простых процедур начальной установки при переходе на резерв они очень небольшие: от долей процента до 1.5%. Однако, чтобы получить истинную стоимость накладных расходов к этим накладным расходам необходимо добавить еще потери, связанные с недоиспользованием процессорной мощности резервной системы. Хотя покупатели стремятся использовать резервную систему для некритических приложений, она оказывается менее загруженной по сравнению с основной системой. Истинно кластерные системы, такие как VAXclasters компании DEC или кластер LifeKeeper Claster отделения NCR компании AT&T, являются примерами намного более сложного управления по сравнению с простыми процедурами начальной установки при переключении на резерв, и полностью используют все доступные процессоры. Однако организация таких систем влечет за собой и большие накладные расходы, которые увеличиваются с ростом числа узлов в кластере.

Требования к системному программному обеспечению

Поставщики, предлагающие системы с горячим резервом, обычно в своих версиях системы UNIX предоставляют также некоторые дополнительные свойства высокой готовности, обеспечивающие быструю загрузку и/или перезагрузку резервной системы в случае переключения при возникновении неисправности.

Журнализация файловой системы

Следствием журнализации изменений файловой системы является то, что файлы всегда находятся в готовом для использования состоянии. Когда система отказывает, журнализованная

файловая система гарантирует, что файлы сохранены в последнем согласованном состоянии. Это позволяет осуществлять переключение на резервную систему без какой-либо порчи данных, а также либо вообще без каких-либо потерь данных, либо с потерей только одной последней транзакции. Такой подход отличается от систем, которые осуществляют журнализацию только метаданных файловой системы - процедура, которая помогает управлять целостностью файловой системы, но не целостностью данных.

Изоляция неисправного процесса

Для активно используемых компонентов программного обеспечения, таких как файловая система, часто применяется технология изоляции неисправных процессов, гарантирующая изоляцию ошибок в одной системе и невозможность их распространения за пределы этой системы.

Мониторы обработки транзакций

Иногда для управления переключением на резерв используются мониторы транзакций, гарантирующие отсутствие потерь данных. При этом для незавершенных транзакций может быть произведен откат назад и базы данных возвращаются к известному согласованному состоянию. Для системы UNIX наиболее известными мониторами транзакций являются Tuxedo компании USL, Encina компании Transarc, CICS/6000 компании IBM и Top End компании NCR.

Другие функции программного обеспечения

В современных системах все возрастающую роль играет диагностика в режиме on-line, позволяющая предвосхищать проблемы, которые могут привести к простою системы. В настоящее время она специфична для каждой системы. В будущем, возможно, диагностика станет частью распределенного управления системой.

Требования высокой готовности к прикладному программному обеспечению

Первыми открытыми системами, построенными в расчете на высокую готовность, были приложения баз данных и систем коммуникаций. Базы данных высокой готовности гарантируют непрерывный доступ к информации, которая является жизненно важной для функционирования многих корпораций и стержнем сегодняшней информационной экономики. Программное обеспечение систем коммуникаций высокой готовности усиливает средства горячего резервирования систем и составляет основу для распределенных систем высокой готовности и систем, устойчивых к стихийным бедствиям.

Высокая готовность баз данных. Несколько компаний, поставляющих базы данных, такие как Oracle, Sybase, Informix имеют в составе своих продуктов программное обеспечение, позволяющее выполнять быструю реконструкцию файлов в случае отказа системы. Это снижает время простоя для зеркальных серверов и кластерных решений.

Продукт Oracle7 Parallel Server позволяет нескольким системам одновременно работать с единым представлением базы данных Oracle. Балансировка нагрузки и распределенный менеджер блокировок, которые позволяют множеству систем обращаться к базе данных в одно и то же время, увеличивают потенциальную готовность базы данных.

Первоначально модель вычислительного кластера была разработана компанией DEC в конце 80-х годов. Она известна под названием VMS-кластеров (или VAX-кластеров), которые представляли собой объединенные в кластер миникомпьютеры VAX, работающие под управлением операционной системы VMS. Компания DEC была первым поставщиком Oracle Parallel Server на VMS-кластерах в 1991 году. После появления версии Oracle 7, у компании Oracle появился интерес к концепции базы данных на UNIX-кластерах и в 1991 году она опубликовала программный интерфейс приложений (API) для своего менеджера блокировок. Публикация этого API означала открытое приглашение всех поставщиков вычислительных

систем для организации поддержки на их платформах Oracle 7 Parallel Server. В результате основные поставщики UNIX-кластеров в настоящее время поддерживают этот продукт.

Кроме того, существуют продукты третьих фирм, дополняющие возможности баз данных известных производителей и обеспечивающие увеличение степени высокой готовности. Например, компания Afic Computer Incorporated (New York) продает продукт под названием Multi Server Option (MSO), который позволяет пользователям осуществлять через сеть TCP/IP одновременную запись в любое количество распределенных зеркальных баз данных Sybase. Информация распространяется широковещательным способом одновременно ко всем базам данных. MSO обеспечивает также балансировку нагрузки по множеству серверов. Если один из серверов отказывает, программное обеспечение перенаправит запрос на дублирующую базу данных в течение не более 30 секунд.

Требования к сетевой организации и к коммуникациям

Системы высокой готовности требуют также высокой готовности коммуникаций. Чем быстрее коммуникации между машинами, тем быстрее происходит восстановление после отказа. Некоторые конфигурации систем высокой готовности имеют дублированные коммуникационные связи. В результате связь перестает быть точкой, отказ которой может привести к выходу из строя всей системы. Существующая в настоящее время тенденция в направлении сетей LAN и WAN с повышенной пропускной способностью обеспечивает как локальные, так и удаленные компьютеры более быстрыми коммуникациями, что приводит в итоге к более быстрому восстановлению в распределенных системах.

Современная сетевая технология сама по себе требует устранения таких точек, выход из строя которых, может привести к отказу всей сети. Сегодня при создании сетей характерно использование более сложных сетевых устройств от различных поставщиков, таких как маршрутизаторы и сетевые хабы. Маршрутизаторы, которые определяют путь данных в сетях, могут вычислить новый путь в случае отказа связи. Хабы могут иметь конфигурации с избыточными устройствами и могут изолировать отказы в физической сети для предотвращения отказа всей сети. Важную роль в поддержании оптимального функционирования систем играют также сетевые анализаторы, позволяющие вызвать системного менеджера по любому симптому, который может потенциально привести к простою.

Высокая готовность сетевой организации всегда очень зависит от размера сети. По мере своего разрастания сеть стремится приобрести свойства встроенной надежности: чем больше сеть, тем более желательно использование альтернативных маршрутов, чтобы отказ одного компонента блокировал только какую-то небольшую ее часть.

В настоящее время ведутся активные разработки в направлении создания высокоскоростных сетей ATM. Широкое внедрение этой технологии позволит достаточно быстро передавать большие объемы информации через WAN и обеспечит высокоскоростное переключение в случае отказа между удаленными системами.

Цены на сети высокой готовности в значительной степени зависят от их конфигурации. Ниже даны несколько примеров для одноранговых сетей высокой готовности.

Для небольшой сети, расположенной на одной площадке и имеющей менее 500 узлов, пользователь должен ожидать примерно 50-процентного увеличения стоимости за обеспечение высокой готовности.

Средняя по размеру сеть, предполагающая размещение на нескольких площадках и имеющая до 5000 узлов потребует от 30 до 40 процентов надбавки к стоимости обычной сети.

Для очень большой сети с более 5000 узлов, расположенных на одной площадке надбавка к стоимости за обеспечение высокой готовности уменьшается примерно до 20 - 30%.

В отказоустойчивых системах стоимость сети еще больше увеличивается. Надбавка к стоимости за базовую отказоустойчивую сеть обычно находится в пределах 80-100 и более процентов, и связана скорее не с размером сети, а с платой за высокую готовность.

"Кластеризация" как способ обеспечения высокой готовности системы

Базовая модель VAX/VMS кластеров

Компания DEC первой анонсировала концепцию кластерной системы в 1983 году, определив ее как группу объединенных между собой вычислительных машин, представляющих собой единый узел обработки информации. По существу VAX-кластер представляет собой слабосвязанную многомашинную систему с общей внешней памятью, обеспечивающую единый механизм управления и администрирования.

VAX-кластер обладает следующими свойствами:

Разделение ресурсов. Компьютеры VAX в кластере могут разделять доступ к общим ленточным и дисковым накопителям. Все компьютеры VAX в кластере могут обращаться к отдельным файлам данных как к локальным.

Высокая готовность. Если происходит отказ одного из VAX-компьютеров, задания его пользователей автоматически могут быть перенесены на другой компьютер кластера. Если в системе имеется несколько контроллеров HSC и один из них отказывает, другие контроллеры HSC автоматически подхватывают его работу.

Высокая пропускная способность. Ряд прикладных систем могут пользоваться возможностью параллельного выполнения заданий на нескольких компьютерах кластера.

Удобство обслуживания системы. Общие базы данных могут обслуживаться с единственного места. Прикладные программы могут инсталлироваться только однажды на общих дисках кластера и разделяться между всеми компьютерами кластера.

Расширяемость. Увеличение вычислительной мощности кластера достигается подключением к нему дополнительных VAX-компьютеров. Дополнительные накопители на магнитных дисках и магнитных лентах становятся доступными для всех компьютеров, входящих в кластер.

Работа VAX-кластера определяется двумя главными компонентами. Первым компонентом является высокоскоростной механизм связи, а вторым - системное программное обеспечение, которое обеспечивает клиентам прозрачный доступ к системному сервису. Физически связи внутри кластера реализуются с помощью трех различных шинных технологий с различными характеристиками производительности.

Основные методы связи в VAX-кластере представлены на рис. 11.1.

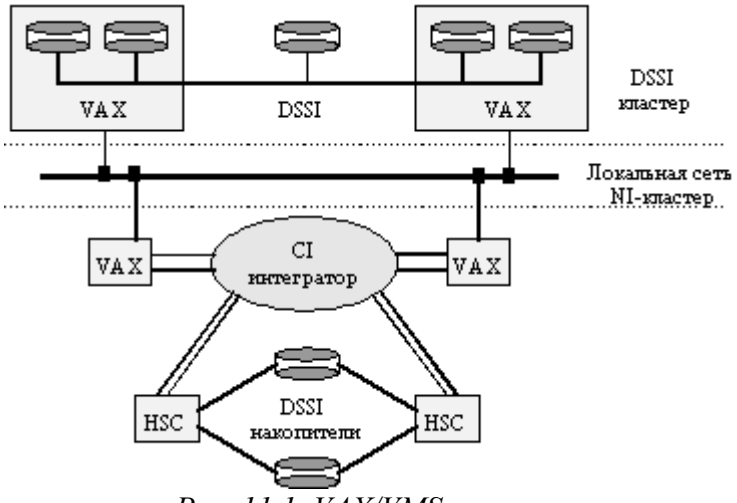


Рис. 11.1. VAX/VMS-кластер

Шина связи компьютеров CI (Computer Interconnect) работает со скоростью 70 Мбит/с и используется для соединения компьютеров VAX и контроллеров HSC с помощью коммутатора Star Coupler. Каждая связь CI имеет двойные избыточные линии, две для передачи и две для приема, используя базовую технологию CSMA, которая для устранения коллизий использует специфические для данного узла задержки. Максимальная длина связи CI составляет 45 метров. Звездообразный коммутатор Star Coupler может поддерживать подключение до 32 шин CI, каждая из которых предназначена для подсоединения компьютера VAX или контроллера HSC. Контроллер HSC представляет собой интеллектуальное устройство, которое управляет работой дисковых и ленточных накопителей.

Компьютеры VAX могут объединяться в кластер также посредством локальной сети

Ethernet, используя NI - Network Interconnect (так называемые локальные VAX-кластеры), однако производительность таких систем сравнительно низкая из-за необходимости делить пропускную способность сети Ethernet между компьютерами кластера и другими клиентами сети.

В начале 1992 года компания DEC анонсировала поддержку построения кластера на основе шины DSSI (Digital Storage System Interconnect). На шине DSSI могут объединяться до четырех компьютеров VAX нижнего и среднего класса. Каждый компьютер может поддерживать несколько адаптеров DSSI. Отдельная шина DSSI работает со скоростью 4 Мбайт/с (32 Мбит/с) и допускает подсоединение до 8 устройств. Поддерживаются следующие типы устройств: системный адаптер DSSI, дисковый контроллер серии RF и ленточный контроллер серии TF. DSSI ограничивает расстояние между узлами в кластере 25 метрами.

Во всем мире насчитывалось более 20000 установок VAX-кластеров. Почти все из них построены с использованием шинного интерфейса CI.

Системное программное обеспечение VAX-кластеров

Для гарантии правильного взаимодействия процессоров друг с другом при обращениях к общим ресурсам, таким, например, как диски, компания DEC использует распределенный менеджер блокировок DLM (Distributed Lock Manager). Очень важной функцией DLM является обеспечение когерентного состояния дисковых кэшей для операций ввода/вывода операционной системы и прикладных программ. Например, в приложениях реляционных СУБД DLM несет ответственность за поддержание согласованного состояния между буферами базы данных на различных компьютерах кластера.

Задача поддержания когерентности кэш-памяти ввода/вывода между процессорами в кластере подобна задаче поддержания когерентности кэш-памяти в сильно связанной многопроцессорной системе, построенной на базе некоторой шины. Блоки данных могут одновременно появляться в нескольких кэшах и если один процессор модифицирует одну из этих копий, другие существующие копии не отражают уже текущее состояние блока данных. Концепция захвата блока (владения блоком) является одним из способов управления такими ситуациями. Прежде чем блок может быть модифицирован должно быть обеспечено владение блоком.

Работа с DLM связана со значительными накладными расходами. Накладные расходы в среде VAX/VMS могут быть большими, требующими передачи до шести сообщений по шине CI для одной операции ввода/вывода. Накладные расходы могут достигать величины 20% для каждого процессора в кластере. Поставщики баз данных при использовании двухпроцессорного VAX-кластера обычно рассчитывают получить увеличение пропускной способности в 1.8 раза для транзакций выбора и в 1.3 раза для транзакций обновления базы данных.

Критерии оценки кластеров Gartner Group

Различные группы промышленных аналитиков считают VAX-кластеры образцом, по которому должны реализовываться другие кластерные системы, и примером свойств, которыми они должны обладать. VAX-кластеры являются очень надежными и высокопроизводительными

вычислительными системами, основанными на миникомпьютерах. Хотя они базируются на патентованной операционной среде (VMS), опыт их эксплуатации показал, что основные свойства, которыми они обладают очень хорошо соответствуют потребностям коммерческих учреждений.

В настоящее время на смену VAX-кластерам приходят UNIX-кластеры. При этом VAX-кластеры предлагают проверенный набор решений, который устанавливает критерии для оценки подобных систем. Одна из известных аналитических групп (Gartner Group) опубликовала модель для сравнения, которая получила название "Критерии оценки кластеров" (MCS Research Note T-470-1411, November 22, 1993). Эта модель базируется на наборе свойств VAX-кластера и используется для сравнения UNIX-кластеров. Критерии этой модели представлены в табл. 11.1.

Таблица 11.1

Архитектурная модель	VAX-кластер
Масштабируемость	да
Масштабируемость (>6 узлов)	да
Смешанный размер	да
Смешанные поколения	да
Единственное представление данных	да
Балансировка нагрузки	да
Общий планировщик работ	да
Общий менеджер систем	да
Надежность на уровне кластера	да
Готовность	да
Подавление одиночного отказа или сбоя	да
Симметричный резерв аппаратных	да
средств	да
Симметричный резерв программных	да
средств	
Автоматическая реконфигурация	
Автоматическое восстановление клиента	
Размещение на нескольких площадках	

В настоящее время UNIX-кластеры все еще отстают от VAX-кластеров по функциональным возможностям. Одно из наибольших отличий связано с реализацией восстановления клиентов в случае отказа. В VAX-кластерах такое восстановление осуществляется средствами программного обеспечения самого VAX-кластера. В UNIX-кластерах эти возможности обычно реализуются отдельным уровнем программного обеспечения, называемым монитором транзакций. В UNIX-кластерах мониторы транзакций кроме того используются также для целей балансировки нагрузки, в то время как VAX-кластеры имеют встроенную в программное обеспечение утилиту балансировки загрузки.

Ожидается, что UNIX-кластеры подойдут и обгонят по функциональным возможностям VAX-кластеры в период 1996-1997 года.

Кластеры Alpha/OSF компании DEC

Стратегическая линия новых продуктов компании DEC основана на новой аппаратной платформе Alpha AXP, которая поддерживает несколько операционных систем (в их числе OpenVMS, DEC OSF/1 и Windows NT). Компания объявила о создании UNIX-кластеров на своей новой платформе Alpha/OSF в 1993 году. Главной задачей при этом считается достижение тех же функциональных возможностей, которыми обладали VAX/VMS-кластеры.

В основу нового кластерного решения положен высокоскоростной коммутатор GigaSwitch, максимальная пропускная способность которого достигает 3.6 Гбайт/с.

GigaSwitch представляет собой протокольно независимое устройство коммутации пакетов, которое может иметь 36 портов и поддерживать протоколы Ethernet, FDDI и ATM. Поддержка различных протоколов реализована с помощью набора специальных сменных карт. Предполагается, что к GigaSwitch помимо рабочих станций Alpha смогут подключаться и UNIX-станции других производителей.

В апреле 1994 года компания DEC анонсировала продукт под названием Integrated Cluster Server, который по своим функциональным характеристикам должен соответствовать стандарту VAX/VMS-кластеров. Основу этого решения составляет новая технология так называемой рефлективной памяти, имеющейся в каждом узле кластера. Устройства рефлективной памяти разных узлов кластера объединяются с помощью высокоскоростных каналов на основе интерфейса PCI (см. рис. 11.2). Обмен информацией между узлами типа "память-память" по этим каналам может производиться со скоростью 100 Мбайт/с, что на порядок превышает скорость обмена по обычным каналам ввода/вывода. Появление этого продукта ожидается в конце 1995 года.

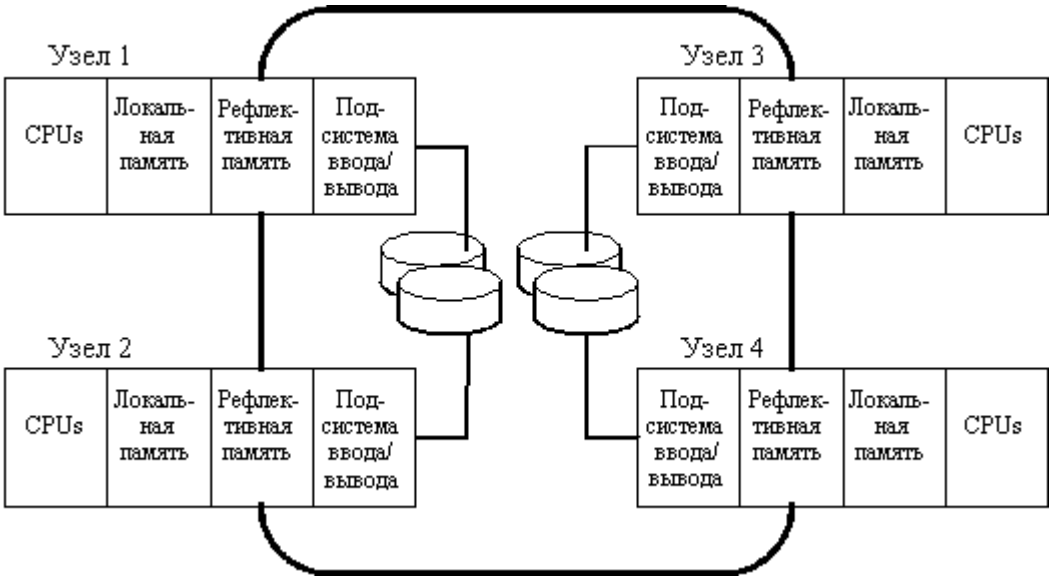


Рис. 11.2. Архитектура кластера компании DEC на базе рефлективной памяти и каналов "память-память"

Компания DEC выпускает RAID массивы StorageWorks, которые поддерживают RAID уровней 0, 1 и 5 и обеспечивают замену компонентов без выключения питания. В стоечной конфигурации поддерживается до 70 накопителей и дополнительные источники питания. Они могут выполнять до 400 операций ввода/вывода в секунду и подключаются к основным компьютерам через интерфейс Differential Fast SCSI-2. Скоростью интерфейса составляет 10 Мбайт/с, максимальная длина связи - 25 м.

UNIX-кластеры компании IBM

Компания IBM предлагает несколько типов слабо связанных систем на базе RS/6000, объединенных в кластеры и работающих под управлением программного продукта High-Availability Clastered Multiprocessor/6000 (HACMP/6000). В этих системах поддерживаются три режима автоматического восстановления системы после отказа:

Режим 1 - в конфигурации с двумя системами, одна из которых является основной, а другая находится в горячем резерве, в случае отказа обеспечивает автоматическое переключение с основной системы на резервную.

Режим 2 - в той же двухмашинной конфигурации позволяет резервному процессору обрабатывать некритичные приложения, выполнение которых в случае отказа основной системы можно либо прекратить совсем, либо продолжать их обработку в режиме деградации.

Режим 3 - можно действительно назвать кластерным решением, поскольку системы в этом режиме работают параллельно, разделяя доступ к логическим и физическим ресурсам пользуясь возможностями менеджера блокировок, входящего в состав HACMP/6000.

Начиная с объявления в 1991 году продукт HACMP/6000 постоянно развивался. В его состав были включены параллельный менеджер ресурсов, распределенный менеджер блокировок и параллельный менеджер логических томов, причем последний обеспечил возможность балансировки загрузки на уровне всего кластера. Максимальное количество узлов в кластере возросло до восьми. В настоящее время в составе кластера появились узлы с симметричной многопроцессорной обработкой, построенные по технологии Data Crossbar Switch, обеспечивающей линейный рост производительности с увеличением числа процессоров.

Первоначально обязательным требованием режима 3 было использование высокопроизводительной дисковой подсистемы IBM 9333, которая использовала последовательный дисковый интерфейс с поддержкой 17.6 Гбайт дискового пространства, а также дисковой подсистемы IBM 9334 с интерфейсом SCSI, обеспечивающим дисковое пространство в 8.2 Гбайт. В августе 1993 года была анонсирована первая подсистема RAID: две модели 7135 RAIDiant Array могут поддерживать до 768 Гбайт дискового пространства в стоечной конструкции серии 900 и 96 Гбайт в напольных тумбовых конструкциях, при этом реализуют RAID уровней 1, 3 и 5.

Кластеры RS/6000 строятся на базе локальных сетей Ethernet, Token Ring или FDDI и могут быть сконфигурированы различными способами с точки зрения обеспечения повышенной надежности:

- Горячий резерв или простое переключение в случае отказа. В этом режиме активный узел выполняет прикладные задачи, а резервный может выполнять некритичные задачи, которые могут быть остановлены в случае необходимости переключения при отказе активного узла.
- Симметричный резерв. Аналогичен горячему резерву, но роли главного и резервного узлов не фиксированы.
- Взаимный подхват или режим с распределением нагрузки. В этом режиме каждый узел в кластере может "подхватывать" задачи, которые выполняются на любом другом узле кластера.

Кластеры AT&T GIS

Отделение GIS (Global Information Systems) образовалось после покупки AT&T компании NCR, успешно работавшей в направлении создания систем с симметричной многопроцессорной обработкой (SMP) и систем с массовым параллелизмом (MPP) на базе микропроцессоров Intel. В 1993 году NCR анонсировала программное обеспечение для поддержки высокой готовности, получившее название LifeKeeper FRS (Fault Recilient Systems) Clastering Software, которое вместе с дисковыми массивами NCR позволяло строить высоконадежные кластерные решения. В состав кластеров NCR могут входить многопроцессорные системы серий 3400 и 3500, каждая из которых включает от 1 до 8 процессоров 486DX2 или Pentium (рис. 11.3).

Disk Array Subsystem 6298 включает до 20 дисковых накопителей емкостью 1 Гбайт и поддерживает RAID уровней 0, 1, 3 и 5 в любой комбинации. Подсистема обеспечивает замену дисковых накопителей, вентиляторов и источников питания в режиме on-line, т.е. без приостановки работы системы. В ней предусмотрено три порта и возможна поставка с избыточными контроллерами.

Программное обеспечение LifeKeeper допускает построение кластеров высокой готовности с четырьмя узлами, причем любой из узлов кластера может служить в качестве резервного для других узлов. Плановое время простоя для инсталляции программного обеспечения может быть значительно снижено, поскольку переключение на резерв можно инициировать вручную, затем модифицировать программное обеспечение и произвести

обратное переключение с резерва. LifeKeeper обеспечивает также восстановление системы после обнаружения ошибок в системных, прикладных программах и периферийном оборудовании. Он обеспечивает автоматическое переключение при обнаружении отказа и инициируемое оператором обратное переключение. Все связи узлов кластера с помощью Ethernet, Token Ring и FDDI дублированы, а дисковые подсистемы, как уже отмечалось, могут подключаться сразу к нескольким узлам кластера. Все это обеспечивает построение системы, устойчивой к одиночным отказам, причем программное обеспечение выполняет автоматическое обнаружение отказов и восстановление системы.

Рис. 11.3. Архитектура двухмашинного кластера AT&T GIS LifeKeeper FRS

При использовании Oracle Parallel Server распределенный менеджер блокировок, входящий в состав LifeKeeper, позволяет параллельной базе данных работать с системой высокой готовности.

В планы компании входит построение крупномасштабных кластеров для университетских кампусов, а также глобальных кластеров, способных продолжать работу в случае стихийных бедствий.

Кластеры Sequent Computer Systems

Sequent была, по-видимому, второй после IBM компанией, осуществившей поставки UNIX-кластеров баз данных в середине 1993 года. Она предлагает решения, соответствующие среднему и высокому уровню готовности своих систем. Первоначально Sequent Hi-Av Systems обеспечивали дублирование систем, которые разделяли общие диски. Пользователи могли выбирать ручной или автоматический режим переключения на резерв в случае отказа. Программный продукт ptx/CLASTERS, который может использоваться совместно с продуктом Hi-Av Systems, включает ядро, отказоустойчивый распределенный менеджер блокировок, обеспечивающий разделение данных между приложениями. Продукт ptx/NQS предназначен для балансировки пакетных заданий между узлами кластера, а ptx/LAT расширяет возможности управления пользовательскими приложениями в режиме on-line. Продукт ptx/ARGUS обеспечивает централизованное управление узлами кластера, а ptx/SVM (распределенный менеджер томов) представляет собой инструментальное средство управления внешней памятью системы. Hi-Av Systems обеспечивает также горячее резервирование IP адресов и позволяет кластеру, в состав которого входят до четырех узлов, иметь единственный сетевой адрес (рис. 11.4).

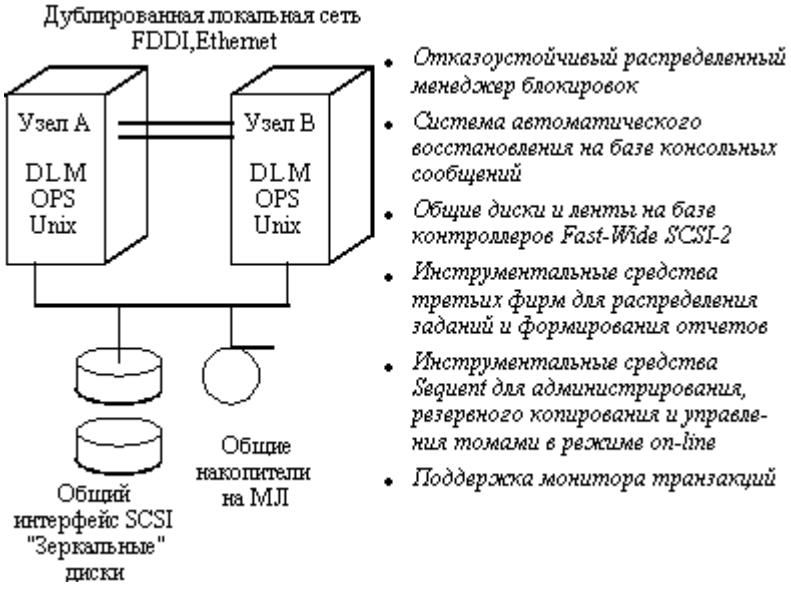


Рис. 11.4. Архитектура двухмашинного кластера SE90 компании Sequent

Компания Sequent одной из первых освоила технологию Fast-Wide SCSI, что позволило ей добиться значительного увеличения производительности систем при обработке транзакций. Компания поддерживает дисковые подсистемы RAID уровней 1, 3 и 5. Кроме того она предлагает в качестве разделяемого ресурса ленточные накопители SCSI. Модель SE90 поддерживает кластеры, в состав которых могут входить два, три или четыре узла, представляющих собой многопроцессорные системы Symmetry 2000 или Symmetry 5000 в любой комбинации. Это достаточно мощные системы. Например, Sequent Symmetry 5000 Series 790 может иметь от 2 до 30 процессоров Pentium 66 МГц, оперативную память емкостью до 2 Гбайт и дисковую память емкостью до 840 Гбайт.

При работе с Oracle Parallel Server все узлы кластера работают с единственной копией базы данных, расположенной на общих разделяемых дисках.

Системы высокой готовности Hewlett-Packard

Продукты высокой готовности HP включают дисковые массивы, программное обеспечение Switchover/UX и SharePlex, а также заказные услуги по организации систем, устойчивых к стихийным бедствиям. HP до настоящего времени не поддерживает Oracle Parallel Server, хотя имеются планы по организации такой поддержки. Схема "слабо связанного" кластера HP включает до 8 серверов HP 9000 Series 800, причем семь из них являются основными системами, а восьмая находится в горячем резерве и готова заменить любую из основных систем в случае отказа (рис. 11.5). Для этого в нормальном режиме работы основные системы посылают резервной сообщения (так называемый "пульс"), подтверждающие их работоспособное состояние. Если резервная система обнаруживает потерю "пульса" какой-либо основной системой, она прекращает выполнение своих процессов, берет на себя управление дисками отказавшей системы, осуществляет перезагрузку, переключает на себя сетевой адрес отказавшей системы и затем перезапускает приложения. Весь процесс переключения может занимать от 10 до 20 или более минут в зависимости от приложения.

Продукты Switchover и Shareplex могут использоваться совместно. При этом Switchover обеспечивает высокую готовность, а Shareplex расширяет возможности кластерной системы, предоставляя поддержку общего управления системами, отказоустойчивости при стихийных бедствиях и разделяемого доступа к распределенным ресурсам систем.

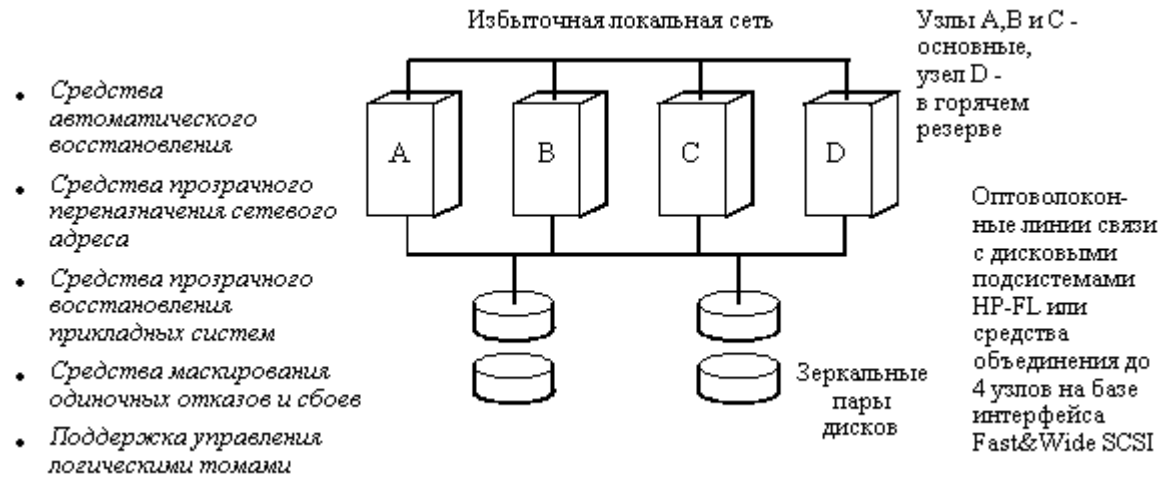


Рис. 11.5. Отказоустойчивая архитектура Switchover/UX компании Hewlett Packard

Хотя Oracle Parallel Server еще не поставлялся в составе кластерных систем HP, компания проводит активные работы в этом направлении. В частности, в последнее время появились новые программные продукты: MC/LockManager - менеджер блокировок, обеспечивающий связь между экземплярами Oracle, функционирующими в кластере, и MC/ServiceGuard - продукт, заменяющий ранее поставлявшийся Switchover.

Продукты HP известны своей высокой надежностью и хорошими показателями наработки на отказ. Дисковые массивы HP-FL поддерживают RAID уровней 0, 3 и 5, а также замену без выключения питания. В настоящее время в системах HP используются устройства Fast/Wide SCSI Disk Array, поддерживающие RAID уровней 0, 1, 3 и 5 с 50 Гбайт общего дискового пространства. В отличие от устройств HP-FL, подсоединение которых осуществлялось с помощью оптоволоконного кабеля, SCSI Disk Array подключается посредством обычных медных кабелей, длина которых может достигать 25 метров.

Новая версия ОС Version 10.0 HP-UX обеспечивает журнализацию файлов и распределенное управление блокировками, что позволяет HP активно конкурировать с продукцией высокой готовности других поставщиков.

Кластерные решения Sun Microsystems

Sun Microsystems предлагает кластерные решения на основе своего продукта SPARCclaster PDB Server, в котором в качестве узлов используются многопроцессорные SMP-серверы SPARCserver 1000 и SPARCcenter 2000. Максимально в состав SPARCserver 1000 могут входить до восьми процессоров, а в SPARCcenter 2000 до 20 процессоров SuperSPARC. В комплект базовой поставки входят следующие компоненты: два кластерных узла на основе SPARCserver 1000/1000E или SPARCcenter 2000/2000E, два дисковых массива SPARCstorage Array, а также пакет средств для построения кластера, включающий дублированное оборудование для осуществления связи, консоль управления кластером Claster Management Console, программное обеспечение SPARCclaster PDB Software и пакет сервисной поддержки кластера.

Для обеспечения высокой производительности и готовности коммуникаций кластер поддерживает полное дублирование всех магистралей данных. Узлы кластера объединяются с помощью каналов SunFastEthernet с пропускной способностью 100 Мбит/с. Для подключения дисковых подсистем используется оптоволоконный интерфейс Fibre Channel с пропускной способностью 25 Мбит/с, допускающий удаление накопителей и узлов друг от друга на расстояние до 2 км. Все связи между узлами, узлами и дисковыми подсистемами дублированы на аппаратном уровне. Аппаратные, программные и сетевые средства кластера обеспечивают отсутствие такого места в системе, одиночный отказ или сбой которого выводил бы всю систему из строя.

SPARCclaster PDB Server поддерживает полностью автоматическое обнаружение отказов, их изоляцию и восстановление после отказа, причем обнаружение и изоляция отказа

выполняются на разных уровнях в зависимости от отказавшего компонента (системы связи между узлами, дисковой подсистемы, сетевого подключения или целиком узла). При этом процесс восстановления осуществляется достаточно быстро, поскольку в случае подобных отказов не требуется полной перезагрузки системы.

В состав программного обеспечения кластера входят четыре основных компонента: отказоустойчивый распределенный менеджер блокировок, распределенный менеджер томов, программные средства управления обнаружением отказов и управления восстановлением, программное обеспечение управления кластером.

Sun Microsystems выпускает дисковые массивы, обеспечивающие RAID уровней 0 и 1. Ожидается появление поддержки RAID уровня 5. Максимальная емкость дискового массива для SPARCserver 1000 составляет 63 Гбайт при использовании SPARCstorage Array Model 100 и 324 Гбайт при использовании SPARCstorage Array Model 200. Для SPARCcenter 2000 эти цифры составляют соответственно 105 Гбайт и 324 Гбайт.

Отказоустойчивые решения Data General

Линия продуктов высокой готовности компании Data General включает системы, основными свойствами которых являются: полностью автоматическое восстановление без потери данных, конфигурируемые дисковые и ленточные массивы и средства поддержания высокой готовности, встроенные в системное программное обеспечение.

Data General поставляет многопроцессорные SMP-серверы серий AV 5500, AV 8500 и AV 9500. Эти серверы поддерживают работу с отказоустойчивыми дисковыми и ленточными подсистемами CLARiiON, средства автоматической диагностики AV/Alert, инициируемые оператором или автоматически средства переключения на резервную систему, управление внешней памятью в режиме on-line, управление вводом/выводом и быстрое восстановление файлов.

При обнаружении отказа процессора, памяти или компоненты ввода/вывода система автоматически начинает процесс выключения и затем осуществляет собственную перезагрузку с исключением отказавших компонент. Стандартным средством указанных систем является наличие избыточных источников питания.

Максимальная степень готовности достигается при подключении двух серверов к высоконадежному дисковому массиву CLARiiON. Дисковые массивы CLARiiON Series C2000 Disk Array обеспечивают RAID уровней 0, 1, 3 и 5 в любых сочетаниях, до 20 накопителей в одном шасси общей емкостью до 80 Гбайт и возможность замены накопителя без выключения питания. В конструкции дискового массива используются избыточные интеллектуальные контроллеры с дублированными связями, обеспечивающие отказоустойчивость. Ленточный массив CLARiiON Series 4000 поддерживает отказоустойчивое резервное копирование и восстановление. В составе массива используется специальный процессор, реализующий схему расщепления данных, подобную RAID уровня 5. Ленточный массив обеспечивает не только высокую пропускную способность, но и реализует защиту от отказов носителя и накопителя. В действительности, даже при отказе накопителя или картриджа, операции по резервному копированию или восстановлению данных продолжаются без потери данных. В массив можно устанавливать 3, 5 или 7 накопителей. При двухкратной компрессии данных общая емкость ленточного массива может достигать 48 Гбайт.

Классификация компьютеров по областям применения

- Классификация компьютеров по областям применения
  - Персональные компьютеры и рабочие станции
  - X-терминалы
  - Серверы
  - Мейнфреймы
  - Кластерные архитектуры

Персональные компьютеры и рабочие станции

Персональные компьютеры (ПК) появились в результате эволюции миникомпьютеров при переходе элементной базы машин с малой и средней степенью интеграции на большие и сверхбольшие интегральные схемы. ПК, благодаря своей низкой стоимости, очень быстро завоевали хорошие позиции на компьютерном рынке и создали предпосылки для разработки новых программных средств, ориентированных на конечного пользователя. Это прежде всего - "дружественные пользовательские интерфейсы", а также проблемно-ориентированные среды и инструментальные средства для автоматизации разработки прикладных программ.

Миникомпьютеры стали прародителями и другого направления развития современных систем - 32-разрядных машин. Создание RISC-процессоров и микросхем памяти емкостью более 1 Мбит привело к окончательному оформлению настольных систем высокой производительности, которые сегодня известны как рабочие станции. Первоначальная ориентация рабочих станций на профессиональных пользователей (в отличие от ПК, которые в начале ориентировались на самого широкого потребителя непрофессионала) привела к тому, что рабочие станции - это хорошо сбалансированные системы, в которых высокое быстродействие сочетается с большим объемом оперативной и внешней памяти, высокопроизводительными внутренними магистралями, высококачественной и быстродействующей графической подсистемой и разнообразными устройствами ввода/вывода. Это свойство выгодно отличает рабочие станции среднего и высокого класса от ПК и сегодня. Даже наиболее мощные IBM PC совместимые ПК не в состоянии удовлетворить возрастающие потребности систем обработки из-за наличия в их архитектуре ряда "узких мест".

Тем не менее быстрый рост производительности ПК на базе новейших микропроцессоров Intel в сочетании с резким снижением цен на эти изделия и развитием технологии локальных шин (VESA и PCI), позволяющей устранить многие "узкие места" в архитектуре ПК, делают современные персональные компьютеры весьма привлекательной альтернативой рабочим станциям. В свою очередь производители рабочих станций создали изделия так называемого "начального уровня", которые по стоимостным характеристикам близки к высокопроизводительным ПК, но все еще сохраняют лидерство по производительности и возможностям наращивания. Насколько успешно удаться ПК на базе процессоров 486 и Pentium бороться против рабочих станций UNIX, покажет будущее, но уже в настоящее время появилось понятие "персональной рабочей станции", которое объединяет оба направления.

Современный рынок "персональных рабочих станций" не просто определить. По сути он представляет собой совокупность архитектурных платформ персональных компьютеров и рабочих станций, которые появились в настоящее время, поскольку поставщики компьютерного оборудования уделяют все большее внимание рынку продуктов для коммерции и бизнеса. Этот рынок традиционно считался вотчиной миникомпьютеров и мейнфреймов, которые поддерживали работу настольных терминалов с ограниченным интеллектом. В прошлом персональные компьютеры не были достаточно мощными и не располагали достаточными функциональными возможностями, чтобы служить адекватной заменой подключенных к главной машине терминалов. С другой стороны, рабочие станции на платформе UNIX были очень сильны в научном, техническом и инженерном секторах и были почти также неудобны, как и ПК для того чтобы выполнять серьезные офисные приложения. С

тех пор ситуация изменилась коренным образом. Персональные компьютеры в настоящее время имеют достаточную производительность, а рабочие станции на базе UNIX имеют программное обеспечение, способное выполнять большинство функций, которые стали ассоциироваться с понятием "персональной рабочей станции". Вероятно оба этих направления могут серьезно рассматриваться в качестве сетевого ресурса для систем масштаба предприятия. В результате этих изменений практически ушли со сцены старомодные миникомпьютеры с их патентованной архитектурой и использованием присоединяемых к главной машине терминалов. По мере продолжения процесса разукрупнения (downsizing) и увеличения производительности платформы Intel наиболее мощные ПК (но все же чаще открытые системы на базе UNIX) стали использоваться в качестве серверов, постепенно заменяя миникомпьютеры.

Среди других факторов, способствующих этому процессу, следует выделить:

- Применение ПК стало более разнообразным. Помимо обычных для этого класса систем текстовых процессоров, даже средний пользователь ПК может теперь работать сразу с несколькими прикладными пакетами, включая электронные таблицы, базы данных и высококачественную графику.
- Адаптация графических пользовательских интерфейсов существенно увеличила требования пользователей ПК к соотношению производительность/стоимость. И хотя оболочка MS Windows может работать на моделях ПК 386SX с 2 Мбайтами оперативной памяти, реальные пользователи хотели бы использовать все преимущества подобных систем, включая возможность комбинирования и эффективного использования различных пакетов.
- Широкое распространение систем мультимедиа прямо зависит от возможности использования высокопроизводительных ПК и рабочих станций с адеквантными аудио- и графическими средствами, и объемами оперативной и внешней памяти.
- Слишком высокая стоимость мейнфреймов и даже систем среднего класса помогла сместить многие разработки в область распределенных систем и систем клиент-сервер, которые многим представляются вполне оправданной по экономическим соображениям альтернативой. Эти системы прямо базируются на высоконадежных и мощных рабочих станциях и серверах.

В начале представлялось, что необходимость сосредоточения высокой мощности на каждом рабочем месте приведет к переходу многих потребителей ПК на UNIX-станции. Это определялось частично тем, что RISC-процессоры, использовавшиеся в рабочих станциях на базе UNIX, были намного более производительными по сравнению с CISC-процессорами, применявшимися в ПК, а частично мощностью системы UNIX по сравнению с MS-DOS и даже OS/2.

Производители рабочих станций быстро отреагировали на потребность в низкостоимостных моделях для рынка коммерческих приложений. Потребность в высокой мощности на рабочем столе, объединенная с желанием поставщиков UNIX-систем продавать как можно больше своих изделий, привела такие компании как Sun Microsystems и Hewlett Packard на рынок рабочих станций для коммерческих приложений. И хотя значительная часть систем этих фирм все еще ориентирована на технические приложения, наблюдается беспрецедентный рост продаж продукции этих компаний для работы с коммерческими приложениями, требующими все большей и большей мощности для реализации сложных, сетевых прикладных систем, включая системы мультимедиа.

Это привело к временному отступлению производителей ПК на базе микропроцессоров Intel. Острая конкуренция со стороны производителей UNIX-систем и потребности в повышении производительности огромной уже инсталлированной базы ПК, заставили компанию Intel форсировать разработку высокопроизводительных процессоров семейства 486 и Pentium. Процессоры 486 и Pentium, при разработке которого были использованы многие подходы, применявшиеся ранее только в RISC-процессорах, а также использование других технологических усовершенствований, таких как архитектура локальной шины, позволили снабдить ПК достаточной мощностью, чтобы составить конкуренцию рабочим станциям во

многих направлениях рынка коммерческих приложений. Правда для многих других приложений, в частности, в области сложного графического моделирования, ПК все еще сильно отстают.

X-терминалы

X-терминалы представляют собой комбинацию бездисковых рабочих станций и стандартных ASCII-терминалов. Бездисковые рабочие станции часто применялись в качестве дорогих дисплеев и в этом случае не полностью использовали локальную вычислительную мощь. Одновременно многие пользователи ASCII-терминалов хотели улучшить их характеристики, чтобы получить возможность работы в многооконной системе и графические возможности. Совсем недавно, как только стали доступными очень мощные графические рабочие станции, появилась тенденция применения "подчиненных" X-терминалов, которые используют рабочую станцию в качестве локального сервера.

На компьютерном рынке X-терминалы занимают промежуточное положение между персональными компьютерами и рабочими станциями. Поставщики X-терминалов заявляют, что их изделия более эффективны в стоимостном выражении, чем рабочие станции высокого ценового класса, и предлагают увеличенный уровень производительности по сравнению с персональными компьютерами. Быстрое снижение цен, прогнозируемое иногда в секторе X-терминалов, в настоящее время идет очевидно благодаря обострившейся конкуренции в этом секторе рынка. Многие компании начали активно конкурировать за распределение рынка, а быстрый рост объемных поставок создал предпосылки для создания такого рынка. В настоящее время уже достигнута цена в $1000 для X-терминалов начального уровня, что делает эту технологию доступной для широкой пользовательской базы.

Как правило, стоимость X-терминалов составляет около половины стоимости сравнимой по конфигурации бездисковой машины и примерно четверть стоимости полностью оснащенной рабочей станции.

Что такое X-терминал?

Типовой X-терминал включает следующие элементы:

- Экран высокого разрешения - обычно размером от 14 до 21 дюйма по диагонали;
- Микропроцессор на базе Motorola 68xxx или RISC-процессор типа Intel i960, MIPS R3000 или AMD29000;
- Отдельный графический сопроцессор в дополнение к основному процессору, поддерживающий двухпроцессорную архитектуру, которая обеспечивает более быстрое рисование на экране и прокручивание экрана;
- Базовые системные программы, на которых работает система X-Windows и выполняются сетевые протоколы;
- Программное обеспечение сервера X11;
- Переменный объем локальной памяти (от 2 до 8 Мбайт) для дисплея, сетевого интерфейса, поддерживающего TCP/IP и другие сетевые протоколы.
- Порты для подключения клавиатуры и мыши.

X-терминалы отличаются от ПК и рабочих станций не только тем, что не выполняет функции обычной локальной обработки. Работа X-терминалов зависит от главной (хост) системы, к которой они подключены посредством сети. Для того, чтобы X-терминал мог работать, пользователи должны инсталлировать программное обеспечение многооконного сервера X11 на главном процессоре, выполняющим прикладную задачу (наиболее известная версия X11 Release 5). X-терминалы отличаются также от стандартных алфавитно-цифровых ASCII и традиционных графических дисплейных терминалов тем, что они могут быть подключены к любой главной системе, которая поддерживает стандарт X-Windows. Более того, локальная вычислительная мощь X-терминала обычно используется для обработки отображения, а не обработки приложений (называемых клиентами), которые выполняются

удаленно на главном компьютере (сервере). Вывод такого удаленного приложения просто отображается на экране X-терминала.

Минимальный объем требуемой для работы памяти X-терминала составляет 1 Мбайт, но чаще 2 Мбайта. В зависимости от функциональных возможностей изделия оперативная память может расширяться до 32 Мбайт и более.

Оснащенный стандартной системой X-Windows, X-терминал может отображать на одном и том же экране множество приложений одновременно. Каждое приложение может выполняться в своем окне и пользователь может изменять размеры окон, их месторасположение и манипулировать ими в любом месте экрана.

X-Windows - результат совместной работы Масачусетского технологического института (MIT) и корпорации DEC. Система X-Windows (известная также под именем X) в настоящее время является открытым де-факто стандартом для доступа к множеству одновременно выполняющихся приложений с возможностями многооконного режима и графикой высокого разрешения на интеллектуальных терминалах, персональных компьютерах, рабочих станциях и X-терминалах. Она стала стандартом для обеспечения интероперабельности (переносимости) продуктов многих поставщиков и для организации доступа к множеству приложений. В настоящее время X-Windows является стандартом для разработки пользовательского интерфейса. Более 90% поставщиков UNIX-рабочих станций и многие поставщики персональных компьютеров адаптировали систему X-Windows и применяют в качестве стандарта.

Серверы

Прикладные многопользовательские коммерческие и бизнес-системы, включающие системы управления базами данных и обработки транзакций, крупные издательские системы, сетевые приложения и системы обслуживания коммуникаций, разработку программного обеспечения и обработку изображений все более настойчиво требуют перехода к модели вычислений "клиент-сервер" и распределенной обработке. В распределенной модели "клиент-сервер" часть работы выполняет сервер, а часть пользовательский компьютер (в общем случае клиентская и пользовательская части могут работать и на одном компьютере). Существует несколько типов серверов, ориентированных на разные применения: файл-сервер, сервер базы данных, принт-сервер, вычислительный сервер, сервер приложений. Таким образом, тип сервера определяется видом ресурса, которым он владеет (файловая система, база данных, принтеры, процессоры или прикладные пакеты программ).

С другой стороны существует классификация серверов, определяющаяся масштабом сети, в которой они используются: сервер рабочей группы, сервер отдела или сервер масштаба предприятия (корпоративный сервер). Эта классификация весьма условна. Например, размер группы может меняться в диапазоне от нескольких человек до нескольких сотен человек, а сервер отдела обслуживать от 20 до 150 пользователей. Очевидно в зависимости от числа пользователей и характера решаемых ими задач требования к составу оборудования и программного обеспечения сервера, к его надежности и производительности сильно варьируются.

Файловые серверы небольших рабочих групп (не более 20-30 человек) проще всего реализуются на платформе персональных компьютеров и программном обеспечении Novell NetWare. Файл-сервер, в данном случае, выполняет роль центрального хранилища данных. Серверы прикладных систем и высокопроизводительные машины для среды "клиент-сервер" значительно отличаются требованиями к аппаратным и программным средствам.

Типичными для небольших файл-серверов являются: процессор 486DX2/66 или более быстродействующий, 32-Мбайт ОЗУ, 2 Гбайт дискового пространства и один адаптер Ethernet 10BaseT, имеющий быстродействие 10 Мбит/с. В состав таких серверов часто включаются флоппи-дисковод и дисковод компакт-дисков. Графика для большинства серверов несущественна, поэтому достаточно иметь обычный монохромный монитор с разрешением VGA.

Скорость процессора для серверов с интенсивным вводом/выводом некритична. Они должны быть оснащены достаточно мощными блоками питания для возможности установки

дополнительных плат расширения и дисковых накопителей. Желательно применение устройства бесперебойного питания. Оперативная память обычно имеет объем не менее 32 Мбайт, что позволит операционной системе (например, NetWare) использовать большие дисковые кэши и увеличить производительность сервера. Как правило, для работы с многозадачными операционными системами такие серверы оснащаются интерфейсом SCSI (или Fast SCSI). Распределение данных по нескольким жестким дискам может значительно повысить производительность.

При наличии одного сегмента сети и 10-20 рабочих станций пиковая пропускная способность сервера ограничивается максимальной пропускной способностью сети. В этом случае замена процессоров или дисковых подсистем более мощными не увеличивают производительность, так как узким местом является сама сеть. Поэтому важно использовать хорошую плату сетевого интерфейса.

Хотя влияние более быстрого процессора явно на производительности не сказывается, оно заметно снижает коэффициент использования ЦП. Во многих серверах этого класса используется процессоры 486DX2/66, Pentium с тактовой частотой 60 и 90 МГц, microSPARC-II и PowerPC. Аналогично процессорам влияние типа системной шины (EISA со скоростью 33 Мбит/с или PCI со скоростью 132 Мбит/с) также минимально при таком режиме использования.

Однако для файл-серверов общего доступа, с которыми одновременно могут работать несколько десятков, а то и сотен человек, простой однопроцессорной платформы и программного обеспечения Novell может оказаться недостаточно. В этом случае используются мощные многопроцессорные серверы с возможностями наращивания оперативной памяти до нескольких гигабайт, дискового пространства до сотен гигабайт, быстрыми интерфейсами дискового обмена (типа Fast SCSI-2, Fast&Wide SCSI-2 и Fiber Channel) и несколькими сетевыми интерфейсами. Эти серверы используют операционную систему UNIX, сетевые протоколы TCP/IP и NFS. На базе многопроцессорных UNIX-серверов обычно строятся также серверы баз данных крупных информационных систем, так как на них ложится основная нагрузка по обработке информационных запросов. Подобного рода серверы получили название суперсерверов.

По уровню общесистемной производительности, функциональным возможностям отдельных компонентов, отказоустойчивости, а также в поддержке многопроцессорной обработки, системного администрирования и дисковых массивов большой емкости суперсерверы вышли в настоящее время на один уровень с мейнфреймами и мощными миникомпьютерами. Современные суперсерверы характеризуются:

- наличием двух или более центральных процессоров RISC, либо Pentium, либо Intel 486;
- многоуровневой шинной архитектурой, в которой запатентованная высокоскоростная системная шина связывает между собой несколько процессоров и оперативную память, а также множество стандартных шин ввода/вывода, размещенных в том же корпусе;
- поддержкой технологии дисковых массивов RAID;
- поддержкой режима симметричной многопроцессорной обработки, которая позволяет распределять задания по нескольким центральным процессорам или режима асимметричной многопроцессорной обработки, которая допускает выделение процессоров для выполнения конкретных задач.

Как правило, суперсерверы работают под управлением операционных систем UNIX, а в последнее время и Windows NT (на Digital 2100 Server Model A500MP), которые обеспечивают многопотоковую многопроцессорную и многозадачную обработку. Суперсерверы должны иметь достаточные возможности наращивания дискового пространства и вычислительной мощности, средства обеспечения надежности хранения данных и защиты от несанкционированного доступа. Кроме того, в условиях быстро растущей организации, важным условием является возможность наращивания и расширения уже существующей системы.

Мейнфреймы

Мейнфрейм - это синоним понятия "большая универсальная ЭВМ". Мейнфреймы и до сегодняшнего дня остаются наиболее мощными (не считая суперкомпьютеров) вычислительными системами общего назначения, обеспечивающими непрерывный круглосуточный режим эксплуатации. Они могут включать один или несколько процессоров, каждый из которых, в свою очередь, может оснащаться векторными сопроцессорами (ускорителями операций с суперкомпьютерной производительностью). В нашем сознании мейнфреймы все еще ассоциируются с большими по габаритам машинами, требующими специально оборудованных помещений с системами водяного охлаждения и кондиционирования. Однако это не совсем так. Прогресс в области элементно-конструкторской базы позволил существенно сократить габариты основных устройств. Наряду со сверхмощными мейнфреймами, требующими организации двухконтурной водяной системы охлаждения, имеются менее мощные модели, для охлаждения которых достаточно принудительной воздушной вентиляции, и модели, построенные по блочно-модульному принципу и не требующие специальных помещений и кондиционеров.

Основными поставщиками мейнфреймов являются известные компьютерные компании IBM, Amdahl, ICL, Siemens Nixdorf и некоторые другие, но ведущая роль принадлежит безусловно компании IBM. Именно архитектура системы IBM/360, выпущенной в 1964 году, и ее последующие поколения стали образцом для подражания. В нашей стране в течение многих лет выпускались машины ряда ЕС ЭВМ, являвшиеся отечественным аналогом этой системы.

В архитектурном плане мейнфреймы представляют собой многопроцессорные системы, содержащие один или несколько центральных и периферийных процессоров с общей памятью, связанных между собой высокоскоростными магистралями передачи данных. При этом основная вычислительная нагрузка ложится на центральные процессоры, а периферийные процессоры (в терминологии IBM - селекторные, блок-мультиплексные, мультиплексные каналы и процессоры телеобработки) обеспечивают работу с широкой номенклатурой периферийных устройств.

Первоначально мейнфреймы ориентировались на централизованную модель вычислений, работали под управлением патентованных операционных систем и имели ограниченные возможности для объединения в единую систему оборудования различных фирм-поставщиков. Однако повышенный интерес потребителей к открытым системам, построенным на базе международных стандартов и позволяющим достаточно эффективно использовать все преимущества такого подхода, заставил поставщиков мейнфреймов существенно расширить возможности своих операционных систем в направлении совместимости. В настоящее время они демонстрирует свою "открытость", обеспечивая соответствие со спецификациями POSIX 1003.3, возможность использования протоколов межсоединений OSI и TCP/IP или предоставляя возможность работы на своих компьютерах под управлением операционной системы UNIX собственной разработки.

Стремительный рост производительности персональных компьютеров, рабочих станций и серверов создал тенденцию перехода с мейнфреймов на компьютеры менее дорогих классов: миникомпьютеры и многопроцессорные серверы. Эта тенденция получила название "разукрупнение" (downsizing). Однако этот процесс в самое последнее время несколько замедлился. Основной причиной возрождения интереса к мейнфреймам эксперты считают сложность перехода к распределенной архитектуре клиент-сервер, которая оказалась выше, чем предполагалось. Кроме того, многие пользователи считают, что распределенная среда не обладает достаточной надежностью для наиболее ответственных приложений, которой обладают мейнфреймы.

Очевидно выбор центральной машины (сервера) для построения информационной системы предприятия возможен только после глубокого анализа проблем, условий и требований конкретного заказчика и долгосрочного прогнозирования развития этой системы.

Главным недостатком мейнфреймов в настоящее время остается относительно низкое соотношение производительность/стоимость. Однако фирмами-поставщиками мейнфреймов предпринимаются значительные усилия по улучшению этого показателя.

Следует также помнить, что в мире существует огромная инсталлированная база мейнфреймов, на которой работают десятки тысяч прикладных программных систем. Отказаться от годами наработанного программного обеспечения просто не разумно. Поэтому в

настоящее время ожидается рост продаж мейнфреймов по крайней мере до конца этого столетия. Эти системы, с одной стороны, позволят модернизировать существующие системы, обеспечив сокращение эксплуатационных расходов, с другой стороны, создадут новую базу для наиболее ответственных приложений.

Кластерные архитектуры

Двумя основными проблемами построения вычислительных систем для критически важных приложений, связанных с обработкой транзакций, управлением базами данных и обслуживанием телекоммуникаций, являются обеспечение высокой производительности и продолжительного функционирования систем. Наиболее эффективный способ достижения заданного уровня производительности - применение параллельных масштабируемых архитектур. Задача обеспечения продолжительного функционирования системы имеет три составляющих: надежность, готовность и удобство обслуживания. Все эти три составляющих предполагают, в первую очередь, борьбу с неисправностями системы, порождаемыми отказами и сбоями в ее работе. Эта борьба ведется по всем трем направлениям, которые взаимосвязаны и применяются совместно.

Повышение надежности основано на принципе предотвращения неисправностей путем снижения интенсивности отказов и сбоев за счет применения электронных схем и компонентов с высокой и сверхвысокой степенью интеграции, снижения уровня помех, облегченных режимов работы схем, обеспечение тепловых режимов их работы, а также за счет совершенствования методов сборки аппаратуры. Повышение уровня готовности предполагает подавление в определенных пределах влияния отказов и сбоев на работу системы с помощью средств контроля и коррекции ошибок, а также средств автоматического восстановления вычислительного процесса после проявления неисправности, включая аппаратурную и программную избыточность, на основе которой реализуются различные варианты отказоустойчивых архитектур. Повышение готовности есть способ борьбы за снижение времени простоя системы. Основные эксплуатационные характеристики системы существенно зависят от удобства ее обслуживания, в частности от ремонтопригодности, контролепригодности и т.д.

В последние годы в литературе по вычислительной технике все чаще употребляется термин "системы высокой готовности" (High Availability Systems). Все типы систем высокой готовности имеют общую цель - минимизацию времени простоя. Имеется два типа времени простоя компьютера: плановое и неплановое. Минимизация каждого из них требует различной стратегии и технологии. Плановое время простоя обычно включает время, принятое руководством, для проведения работ по модернизации системы и для ее обслуживания. Неплановое время простоя является результатом отказа системы или компонента. Хотя системы высокой готовности возможно больше ассоциируются с минимизацией неплановых простоев, они оказываются также полезными для уменьшения планового времени простоя.

Существует несколько типов систем высокой готовности, отличающиеся своими функциональными возможностями и стоимостью. Следует отметить, что высокая готовность не дается бесплатно. Стоимость систем высокой готовности на много превышает стоимость обычных систем. Вероятно поэтому наибольшее распространение в мире получили кластерные системы, благодаря тому, что они обеспечивают достаточно высокий уровень готовности систем при относительно низких затратах. Термин "кластеризация" на сегодня в компьютерной промышленности имеет много различных значений. Строгое определение могло бы звучать так: "реализация объединения машин, представляющегося единым целым для операционной системы, системного программного обеспечения, прикладных программ и пользователей". Машины, кластеризованные вместе таким способом могут при отказе одного процессора очень быстро перераспределить работу на другие процессоры внутри кластера. Это, возможно, наиболее важная задача многих поставщиков систем высокой готовности.

Первой концепцию кластерной системы анонсировала компания DEC, определив ее как группу объединенных между собой вычислительных машин, представляющих собой единый узел обработки информации. По существу VAX-кластер представляет собой слабосвязанную многомашинную систему с общей внешней памятью, обеспечивающую единый механизм

управления и администрирования. В настоящее время на смену VAX-кластерам приходят UNIX-кластеры. При этом VAX-кластеры предлагают проверенный набор решений, который устанавливает критерии для оценки подобных систем.

VAX-кластер обладает следующими свойствами:

Разделение ресурсов. Компьютеры VAX в кластере могут разделять доступ к общим ленточным и дисковым накопителям. Все компьютеры VAX в кластере могут обращаться к отдельным файлам данных как к локальным.

Высокая готовность. Если происходит отказ одного из VAX-компьютеров, задания его пользователей автоматически могут быть перенесены на другой компьютер кластера. Если в системе имеется несколько контроллеров внешних накопителей и один из них отказывает, другие контроллеры автоматически подхватывают его работу.

Высокая пропускная способность. Ряд прикладных систем могут пользоваться возможностью параллельного выполнения заданий на нескольких компьютерах кластера.

Удобство обслуживания системы. Общие базы данных могут обслуживаться с единственного места. Прикладные программы могут инсталлироваться только однажды на общих дисках кластера и разделяться между всеми компьютерами кластера.

Расширяемость. Увеличение вычислительной мощности кластера достигается подключением к нему дополнительных VAX-компьютеров. Дополнительные накопители на магнитных дисках и магнитных лентах становятся доступными для всех компьютеров, входящих в кластер.

Работа любой кластерной системы определяется двумя главными компонентами: высокоскоростным механизмом связи процессоров между собой и системным программным обеспечением, которое обеспечивает клиентам прозрачный доступ к системному сервису. Более подробно различные способы организации связи компьютеров внутри кластера и особенности системного программного обеспечения различных фирм-поставщиков рассмотрены в разд. 1.11 настоящего обзора.

В настоящее время широкое распространение получила также технология параллельных баз данных. Эта технология позволяет множеству процессоров разделять доступ к единственной базе данных. Распределение заданий по множеству процессорных ресурсов и параллельное их выполнение позволяет достичь более высокого уровня пропускной способности транзакций, поддерживать большее число одновременно работающих пользователей и ускорить выполнение сложных запросов. Существуют три различных типа архитектуры, которые поддерживают параллельные базы данных:

- Симметричная многопроцессорная архитектура с общей памятью (Shared Memory SMP Architecture). Эта архитектура поддерживает единую базу данных, работающую на многопроцессорном сервере под управлением одной операционной системы. Увеличение производительности таких систем обеспечивается наращиванием числа процессоров, устройств оперативной и внешней памяти.
- Архитектура с общими (разделяемыми) дисками (Shared Disk Architecture). Это типичный случай построения кластерной системы. Эта архитектура поддерживает единую базу данных при работе с несколькими компьютерами, объединенными в кластер (обычно такие компьютеры называются узлами кластера), каждый из которых работает под управлением своей копии операционной системы. В таких системах все узлы разделяют доступ к общим дискам, на которых собственно и располагается единая база данных. Производительность таких систем может увеличиваться как путем наращивания числа процессоров и объемов оперативной памяти в каждом узле кластера, так и посредством увеличения количества самих узлов.
- Архитектура без разделения ресурсов (Shared Nothing Architecture). Как и в архитектуре с общими дисками, в этой архитектуре поддерживается единый образ базы данных при работе с несколькими компьютерами, работающими под управлением своих копий операционной системы. Однако в этой архитектуре каждый узел системы имеет собственную оперативную

память и собственные диски, которые не разделяются между отдельными узлами системы. Практически в таких системах разделяется только общий коммуникационный канал между узлами системы. Производительность таких систем может увеличиваться путем добавления процессоров, объемов оперативной и внешней (дисковой) памяти в каждом узле, а также путем наращивания количества таких узлов.

Таким образом, среда для работы параллельной базы данных обладает двумя важными свойствами: высокой готовностью и высокой производительностью. В случае кластерной организации несколько компьютеров или узлов кластера работают с единой базой данных. В случае отказа одного из таких узлов, оставшиеся узлы могут взять на себя задания, выполнявшиеся на отказавшем узле, не останавливая общий процесс работы с базой данных. Поскольку логически в каждом узле системы имеется образ базы данных, доступ к базе данных будет обеспечиваться до тех пор, пока в системе имеется по крайней мере один исправный узел. Производительность системы легко масштабируется, т.е. добавление дополнительных процессоров, объемов оперативной и дисковой памяти, и новых узлов в системе может выполняться в любое время, когда это действительно требуется.

Параллельные базы данных находят широкое применение в системах обработки транзакций в режиме on-line, системах поддержки принятия решений и часто используются при работе с критически важными для работы предприятий и организаций приложениями, которые эксплуатируются по 24 часа в сутки.

Оценка производительности вычислительных систем

•Оценка производительности вычислительных систем
- •Общие замечания
- •MIPS
- •MFLOPS
- •SPECint92, SPECfp92
- •SPECrate_int92, SPECrate_fp92
- •TPC-A, TPC-B, TPC-C
- •AIM

Общие замечания

Основу для сравнения различных типов компьютеров между собой дают стандартные методики измерения производительности. В процессе развития вычислительной техники появилось несколько таких стандартных методик. Они позволяют разработчикам и пользователям осуществлять выбор между альтернативами на основе количественных показателей, что дает возможность постоянного прогресса в данной области.

Единицей измерения производительности компьютера является время: компьютер, выполняющий тот же объем работы за меньшее время является более быстрым. Время выполнения любой программы измеряется в секундах. Часто производительность измеряется как скорость появления некоторого числа событий в секунду, так что меньшее время подразумевает большую производительность.

Однако в зависимости от того, что мы считаем, время может быть определено различными способами. Наиболее простой способ определения времени называется астрономическим временем, временем ответа (response time), временем выполнения(execution time) или прошедшим временем (elapsed time). Это задержка выполнения задания, включающая буквально все: работу процессора, обращения к диску, обращения к памяти, ввод/вывод и накладные расходы операционной системы. Однако при работе в мультипрограммном режиме во время ожидания ввода/вывода для одной программы, процессор может выполнять другую программу, и система не обязательно будет минимизировать время выполнения данной конкретной программы.

Для измерения времени работы процессора на данной программе используется специальный параметр - время ЦП (CPU time), которое не включает время ожидания ввода/вывода или время выполнения другой программы. Очевидно, что время ответа, видимое пользователем, является полным временем выполнения программы, а не временем ЦП. Время ЦП может далее делиться на время, потраченное ЦП непосредственно на выполнение программы пользователя и называемое пользовательским временем ЦП, и время ЦП, затраченное операционной системой на выполнение заданий, затребованных программой, и называемое системным временем ЦП.

В ряде случаев системное время ЦП игнорируется из-за возможной неточности измерений, выполняемых самой операционной системой, а также из-за проблем, связанных со сравнением производительности машин с разными операционными системами. С другой стороны, системный код на некоторых машинах является пользовательским кодом на других и, кроме того, практически никакая программа не может работать без некоторой операционной системы. Поэтому при измерениях производительности процессора часто используется сумма пользовательского и системного времени ЦП.

В большинстве современных процессоров скорость протекания процессов взаимодействия внутренних функциональных устройств определяется не естественными задержками в этих устройствах, а задается единой системой синхросигналов, вырабатываемых некоторым генератором тактовых импульсов, как правило, работающим с постоянной скоростью. Дискретные временные события называются тактами синхронизации (clock ticks), просто тактами (ticks), периодами синхронизации (clock periods), циклами (cycles) или циклами

синхронизации (clock cycles).Разработчики компьютеров обычно говорят о периоде синхронизации, который определяется либо своей длительностью (например, 10 наносекунд), либо частотой (например, 100 МГц). Длительность периода синхронизации есть величина, обратная к частоте синхронизации.

Таким образом, время ЦП для некоторой программы может быть выражено двумя способами: количеством тактов синхронизации для данной программы, умноженным на длительность такта синхронизации, либо количеством тактов синхронизации для данной программы, деленным на частоту синхронизации.

Важной характеристикой, часто публикуемой в отчетах по процессорам, является среднее количество тактов синхронизации на одну команду - CPI (clock cycles per instruction). При известном количестве выполняемых команд в программе этот параметр позволяет быстро оценить время ЦП для данной программы.

Таким образом, производительность ЦП зависит от трех параметров: такта (или частоты) синхронизации, среднего количества тактов на команду и количества выполняемых команд. Невозможно изменить ни один из указанных параметров изолированно от другого, поскольку базовые технологии, используемые для изменения каждого из этих параметров, взаимосвязаны: частота синхронизации определяется технологией аппаратных средств и функциональной организацией процессора; среднее количество тактов на команду зависит от функциональной организации и архитектуры системы команд; а количество выполняемых в программе команд определяется архитектурой системы команд и технологией компиляторов. Когда сравниваются две машины, необходимо рассматривать все три компоненты, чтобы понять относительную производительность.

В процессе поиска стандартной единицы измерения производительности компьютеров было принято несколько популярных единиц измерения, вследствие чего несколько безвредных терминов были искусственно вырваны из их хорошо определенного контекста и использованы там, для чего они никогда не предназначались. В действительности единственной подходящей и надежной единицей измерения производительности является время выполнения реальных программ, и все предлагаемые замены этого времени в качестве единицы измерения или замены реальных программ в качестве объектов измерения на синтетические программы только вводят в заблуждение.

Опасности некоторых популярных альтернативных единиц измерения (MIPS и MFLOPS) будут рассмотрены в соответствующих подразделах.

MIPS

Одной из альтернативных единиц измерения производительности процессора (по отношению к времени выполнения) является MIPS - (миллион команд в секунду). Имеется несколько различных вариантов интерпретации определения MIPS.

В общем случае MIPS есть скорость операций в единицу времени, т.е. для любой данной программы MIPS есть просто отношение количества команд в программе к времени ее выполнения. Таким образом, производительность может быть определена как обратная к времени выполнения величина, причем более быстрые машины при этом будут иметь более высокий рейтинг MIPS.

Положительными сторонами MIPS является то, что эту характеристику легко понять, особенно покупателю, и что более быстрая машина характеризуется большим числом MIPS, что соответствует нашим интуитивным представлениям. Однако использование MIPS в качестве метрики для сравнения наталкивается на три проблемы. Во-первых, MIPS зависит от набора команд процессора, что затрудняет сравнение по MIPS компьютеров, имеющих разные системы команд. Во-вторых, MIPS даже на одном и том же компьютере меняется от программы к программе. В-третьих, MIPS может меняться по отношению к производительности в противоположенную сторону.

Классическим примером для последнего случая является рейтинг MIPS для машины, в состав которой входит сопроцессор плавающей точки. Поскольку в общем случае на каждую команду с плавающей точкой требуется большее количество тактов синхронизации, чем на целочисленную команду, то программы, используя сопроцессор плавающей точки вместо

соответствующих подпрограмм из состава программного обеспечения, выполняются за меньшее время, но имеют меньший рейтинг MIPS. При отсутствии сопроцессора операции над числами с плавающей точкой реализуются с помощью подпрограмм, использующих более простые команды целочисленной арифметики и, как следствие, такие машины имеют более высокий рейтинг MIPS, но выполняют настолько большее количество команд, что общее время выполнения значительно увеличивается. Подобные аномалии наблюдаются и при использовании оптимизирующих компиляторов, когда в результате оптимизации сокращается количество выполняемых в программе команд, рейтинг MIPS уменьшается, а производительность увеличивается.

Другое определение MIPS связано с очень популярным когда-то компьютером VAX 11/780 компании DEC. Именно этот компьютер был принят в качестве эталона для сравнения производительности различных машин. Считалось, что производительность VAX 11/780 равна 1MIPS (одному миллиону команд в секунду).

В то время широкое распространение получил синтетический тест Dhrystone, который позволял оценивать эффективность процессоров и компиляторов с языка С для программ нечисловой обработки. Он представлял собой тестовую смесь, 53% которой составляли операторы присваивания, 32% - операторы управления и 15% - вызовы функций. Это был очень короткий тест: общее число команд равнялось 100. Скорость выполнения программы из этих 100 команд измерялась в Dhrystone в секунду. Быстродействие VAX 11/780 на этом синтетическом тесте составляло 1757Dhrystone в секунду. Таким образом 1MIPS равен 1757 Dhrystone в секунду.

Следует отметить, что в настоящее время тест Dhrystone практически не применяется. Малый объем позволяет разместить все команды теста в кэш-памяти первого уровня современного микропроцессора и он не позволяет даже оценить эффект наличия кэш-памяти второго уровня, хотя может хорошо отражать эффект увеличения тактовой частоты.

Третье определение MIPS связано с IBM RS/6000 MIPS. Дело в том, что ряд производителей и пользователей (последователей фирмы IBM) предпочитают сравнивать производительность своих компьютеров с производительностью современных компьютеров IBM, а не со старой машиной компании DEC. Соотношение между VAX MIPS и RS/6000 MIPS никогда широко не публиковались, но 1 RS/6000 MIPS примерно равен 1.6 VAX 11/780 MIPS.

MFLOPS

Измерение производительности компьютеров при решении научно-технических задач, в которых существенно используется арифметика с плавающей точкой, всегда вызывало особый интерес. Именно для таких вычислений впервые встал вопрос об измерении производительности, а по достигнутым показателям часто делались выводы об общем уровне разработок компьютеров. Обычно для научно-технических задач производительность процессора оценивается в MFLOPS (миллионах чисел-результатов вычислений с плавающей точкой в секунду, или миллионах элементарных арифметических операций над числами с плавающей точкой, выполненных в секунду).

Как единица измерения, MFLOPS, предназначена для оценки производительности только операций с плавающей точкой, и поэтому не применима вне этой ограниченной области. Например, программы компиляторов имеют рейтинг MFLOPS близкий к нулю вне зависимости от того, насколько быстра машина, поскольку компиляторы редко используют арифметику с плавающей точкой.

Ясно, что рейтинг MFLOPS зависит от машины и от программы. Этот термин менее безобидный, чем MIPS. Он базируется на количестве выполняемых операций, а не на количестве выполняемых команд. По мнению многих программистов, одна и та же программа, работающая на различных компьютерах, будет выполнять различное количество команд, но одно и то же количество операций с плавающей точкой. Именно поэтому рейтинг MFLOPS предназначался для справедливого сравнения различных машин между собой.

Однако и с MFLOPS не все обстоит так безоблачно. Прежде всего, это связано с тем, что наборы операций с плавающей точкой не совместимы на различных компьютерах. Например, в суперкомпьютерах фирмы Cray Research отсутствует команда деления (имеется, правда,

операция вычисления обратной величины числа с плавающей точкой, а операция деления может быть реализована с помощью умножения делимого на обратную величину делителя). В то же время многие современные микропроцессоры имеют команды деления, вычисления квадратного корня, синуса и косинуса.

Другая, осознаваемая всеми, проблема заключается в том, что рейтинг MFLOPS меняется не только на смеси целочисленных операций и операций с плавающей точкой, но и на смеси быстрых и медленных операций с плавающей точкой. Например, программа со 100% операций сложения будет иметь более высокий рейтинг, чем программа со 100% операций деления.

Решение обеих проблем заключается в том, чтобы взять "каноническое" или "нормализованное" число операций с плавающей точкой из исходного текста программы и затем поделить его на время выполнения. На рис. 3.1 показано, каким образом авторы тестового пакета "Ливерморские циклы", о котором речь пойдет ниже, вычисляют для программы количество нормализованных операций с плавающей точкой в соответствии с операциями, действительно находящимися в ее исходном тексте. Таким образом, рейтинг реальных MFLOPS отличается от рейтинга нормализованных MFLOPS, который часто приводится в литературе по суперкомпьютерам.

Реальные операции с ПТ	Нормализованные операции с ПТ
Сложение, вычитание, сравнение, умножение	1
Деление, квадратный корень	4
Экспонента, синус, ...	8

Рис. 3.1. Соотношение между реальными и нормализованными операциями с плавающей точкой, которым пользуются авторы "ливерморских циклов" для вычисления рейтинга MFLOPS

Наиболее часто MFLOPS, как единица измерения производительности, используется при проведении контрольных испытаний на тестовых пакетах "Ливерморские циклы" и

LINPACK.

Ливерморские циклы - это набор фрагментов фортран-программ, каждый из которых взят из реальных программных систем, эксплуатируемых в Ливерморской национальной лаборатории им.Лоуренса (США). Обычно при проведении испытаний используется либо малый набор из 14 циклов, либо большой набор из 24 циклов.

Пакет Ливерморских циклов используется для оценки производительности вычислительных машин с середины 60-х годов. Ливерморские циклы считаются типичными фрагментами программ численных задач. Появление новых типов машин, в том числе векторных и параллельных, не уменьшило важности Ливерморских циклов, однако изменились значения производительности и величины разброса между разными циклами.

На векторной машине производительность зависит не только от элементной базы, но и от характера самого алгоритма, т.е. коэффициента векторизуемости. Среди Ливерморских циклов коэффициент векторизуемости колеблется от 0 до 100%, что еще раз подтверждает их ценность для оценки производительности векторных архитектур. Кроме характера алгоритма, на коэффициент векторизуемости влияет и качество векторизатора, встроенного в компилятор.

На параллельной машине производительность существенно зависит от соответствия между структурой аппаратных связей вычислительных элементов и структурой вычислений в алгоритме. Важно, чтобы тестовый пакет представлял алгоритмы различных структур. В Ливерморских циклах встречаются последовательные, сеточные, конвейерные, волновые вычислительные алгоритмы, что подтверждает их пригодность и для параллельных машин. Однако обобщение результатов измерения производительности, полученных для одной параллельной машины, на другие параллельные машины или хотя бы некоторый подкласс параллельных машин, может дать неверный результат, ибо структуры аппаратных связей в таких машинах гораздо более разнообразны, чем, скажем, в векторных машинах.

LINPACK - это пакет фортран-программ для решения систем линейных алгебраических уравнений. Целью создания LINPACK отнюдь не было измерение производительности. Алгоритмы линейной алгебры весьма широко используются в самых разных задачах, и поэтому измерение производительности на LINPACK представляют интерес для многих пользователей. Сведения о производительности различных машин на пакете LINPACK публикуются сотрудником Аргоннской национальной лаборатории (США) Дж. Донгаррой и периодически обновляются.

В основе алгоритмов действующего варианта LINPACK лежит метод декомпозиции. Исходная матрица размером 100x100 элементов (в последнем варианте размером 1000x1000) сначала представляется в виде произведения двух матриц стандартной структуры, над которыми затем выполняется собственно алгоритм нахождения решения. Подпрограммы, входящие в LINPACK, структурированы. В стандартном варианте LINPACK выделен внутренний уровень базовых подпрограмм, каждая из которых выполняет элементарную операцию над векторами. Набор базовых подпрограмм называется BLAS (Basic Linear Algebra Subprograms). Например, в BLAS входят две простые подпрограммы SAXPY (умножение вектора на скаляр и сложение векторов) и SDOT (скалярное произведение векторов). Все операции выполняются над числами с плавающей точкой, представленными с двойной точностью. Результат измеряется в MFLOPS.

Использование результатов работы тестового пакета LINPACK с двойной точностью как основы для демонстрации рейтинга MFLOPS стало общепринятой практикой в компьютерной промышленности. При этом следует помнить, что при использовании исходной матрицы размером 100x100, она полностью может размещаться в кэш-памяти емкостью, например, 1 Мбайт. Если при проведении испытаний используется матрица размером 1000x1000, то емкости такого кэша уже недостаточно и некоторые обращения к памяти будут ускоряться благодаря наличию такого кэша, другие же будут приводить к промахам и потребуют большего времени на обработку обращений к памяти. Для многопроцессорных систем также имеются параллельные версии LINPACK и такие системы часто показывают линейное увеличение производительности с ростом числа процессоров.

Однако, как и любая другая единица измерения, рейтинг MFLOPS для отдельной программы не может быть обобщен на все случаи жизни, чтобы представлять единственную единицу измерения производительности компьютера, хотя очень соблазнительно характеризовать машину единственным рейтингом MIPS или MFLOPS без указания программы.

SPECint92, SPECfp92

Важность создания пакетов тестов, базирующихся на реальных прикладных программах широкого круга пользователей и обеспечивающих эффективную оценку производительности процессоров, была осознана большинством крупнейших производителей компьютерного оборудования, которые в 1988 году учредили бесприбыльную корпорацию SPEC (Standard Performance Evaluation Corporation). Основной целью этой организации является разработка и поддержка стандартизованного набора специально подобранных тестовых программ для оценки производительности новейших поколений высокопроизводительных компьютеров. Членом SPEC может стать любая организация, уплатившая вступительный взнос.

Главными видами деятельности SPEC являются:

1. Разработка и публикация наборов тестов, предназначенных для измерения производительности компьютеров. Перед публикацией объектные коды этих наборов вместе с исходными текстами и инструментальными средствами интенсивно проверяются на предмет возможности импортирования на разные платформы. Они доступны для широкого круга пользователей за плату, покрывающую расходы на разработку и административные издержки. Специальное лицензионное соглашение регулирует вопросы выполнения тестирования и публикации результатов в соответствии с документацией на каждый тестовый набор.

2. SPEC публикует ежеквартальный отчет о новостях SPEC и результатах тестирования: "The SPEC Newsletter", что обеспечивает централизованный источник информации для результатов тестирования на тестах SPEC.

Основным результатом работы SPEC являются наборы тестов. Эти наборы разрабатываются SPEC с использованием кодов, поступающих из разных источников. SPEC работает над импортированием этих кодов на разные платформы, а также создает инструментальные средства для формирования из кодов, выбранных в качестве тестов, осмысленных рабочих нагрузок. Поэтому тесты SPEC отличаются от свободно распространяемых программ. Хотя они могут существовать под похожими или теми же самыми именами, время их выполнения в общем случае будет отличаться.

В настоящее время имеется два базовых набора тестов SPEC, ориентированных на интенсивные расчеты и измеряющих производительность процессора, системы памяти, а также эффективность генерации кода компилятором. Как правило, эти тесты ориентированы на операционную систему UNIX, но они также импортированы и на другие платформы. Процент времени, расходуемого на работу операционной системы и функции ввода/вывода, в общем случае ничтожно мал.

Набор тестов CINT92, измеряющий производительность процессора при обработке целых чисел, состоит из шести программ, написанных на языке Си и выбранных из различных прикладных областей: теория цепей, интерпретатор языка Лисп, разработка логических схем, упаковка текстовых файлов, электронные таблицы и компиляция программ.

Набор тестов CFP92, измеряющий производительность процессора при обработке чисел с плавающей точкой, состоит из 14 программ, также выбранных из различных прикладных областей: разработка аналоговых схем, моделирование методом Монте-Карло, квантовая химия, оптика, робототехника, квантовая физика, астрофизика, прогноз погоды и другие научные и инженерные задачи. Две программы из этого набора написаны на языке Си, а остальные 12 - на Фортране. В пяти программах используется одинарная, а в остальных - двойная точность.

Результаты прогона каждого индивидуального теста из этих двух наборов выражаются отношением времени выполнения одной копии теста на тестируемой машине к времени ее выполнения на эталонной машине. В качестве эталонной машины используется VAX 11/780. SPEC публикует результаты прогона каждого отдельного теста, а также две составные оценки: SPECint92 - среднее геометрическое 6 результатов индивидуальных тестов из набора CINT92 и SPECfp92 - среднее геометрическое 14 результатов индивидуальных тестов из набора CFP92.

Следует отметить, что результаты тестирования на наборах CINT92 и CFT92 сильно зависят от качества применяемых оптимизирующих компиляторов. Для более точного выяснения возможностей аппаратных средств с середины 1994 года SPEC ввел две дополнительные составные оценки: SPECbase_int92 и SPECbase_fp92, которые накладывает определенные ограничения на используемые компиляторы поставщиками компьютеров при проведении испытаний.

SPECrate_int92, SPECrate_fp92

Составные оценки SPECint92 и SPECfp92 достаточно хорошо характеризуют производительность процессора и системы памяти при работе в однозадачном режиме, но они совершенно не подходят для оценки производительности многопроцессорных и однопроцессорных систем, работающих в многозадачном режиме. Для этого нужна оценка пропускной способности системы или ее емкости, показывающая количество заданий, которое система может выполнить в течение заданного интервала времени. Пропускная способность системы определяется прежде всего количеством ресурсов (числом процессоров, емкостью оперативной и кэш-памяти, пропускной способностью шины), которые система может предоставить в распоряжение пользователя в каждый момент времени. Именно такую оценку, названную SPECrate и заменившую ранее применявшуюся оценку SPECthruput89, SPEC предложила в качестве единицы измерения производительности многопроцессорных систем.

При этом для измерения выбран метод "однородной нагрузки" (homogenous capacity metod), заключающийся в том, что одновременно выполняются несколько копий одной и той же тестовой программы. Результаты этих тестов показывают, как много задач конкретного типа могут быть выполнены в указанное время, а их средние геометрические значения (SPECrate_int92 - на наборе тестов, измеряющих производительность целочисленных операций и SPECrate_fp92 - на наборе тестов, измеряющих производительность на операциях с плавающей точкой) наглядно отражают пропускную способность однопроцессорных и многопроцессорных конфигураций при работе в многозадачном режиме в системах коллективного пользования. В качестве тестовых программ для проведения испытаний на пропускную способность выбраны те же наборы CINT92 и CFT92.

При прогоне тестового пакета делаются независимые измерения по каждому отдельному тесту. Обычно такой параметр, как количество запускаемых копий каждого отдельного теста, выбирается исходя из соображений оптимального использования ресурсов, что зависит от архитектурных особенностей конкретной системы. Одной из очевидных возможностей является установка этого параметра равным количеству процессоров в системе. При этом все копии отдельной тестовой программы запускаются одновременно, и фиксируется время завершения последней из всех запущенных программ.

С середины 1994 года SPEC ввела две дополнительные составные оценки: SPECrate_base_int92 и SPECrate_base_fp92, которые накладывает ограничения на используемые компиляторы.

TPC-A, TPC-B, TPC-C

По мере расширения использования компьютеров при обработке транзакций в сфере бизнеса все более важной становится возможность справедливого сравнения систем между собой. С этой целью в 1988 году был создан Совет по оценке производительности обработки транзакций (TPC - Transaction Processing Performance Council), который представляет собой бесприбыльную организацию. Любая компания или организация может стать членом TPC после уплаты соответствующего взноса. На сегодня членами TPC являются практически все крупнейшие производители аппаратных платформ и программного обеспечения для автоматизации коммерческой деятельности. К настоящему времени TPC создал три тестовых пакета для обеспечения объективного сравнения различных систем обработки транзакций и планирует создать новые оценочные тесты.

В компьютерной индустрии термин транзакция (transaction) может означать почти любой вид взаимодействия или обмена информацией. Однако в мире бизнеса "транзакция" имеет вполне определенный смысл: коммерческий обмен товарами, услугами или деньгами. В настоящее время практически все бизнес-транзакции выполняются с помощью компьютеров. Наиболее характерными примерами систем обработки транзакций являются системы управления учетом, системы резервирования авиабилетов и банковские системы. Таким образом, необходимость стандартов и тестовых пакетов для оценки таких систем все больше усиливается.

До 1988 года отсутствовало общее согласие относительно методики оценки систем обработки транзакций. Широко использовались два тестовых пакета: Дебет/Кредит и TPI. Однако эти пакеты не позволяли осуществлять адекватную оценку систем: они не имели полных, основательных спецификаций; не давали объективных, проверяемых результатов; не содержали полного описания конфигурации системы, ее стоимости и методологии тестирования; не обеспечивали объективного, беспристрастного сравнения одной системы с другой.

Чтобы решить эти проблемы, и была создана организация TPC, основной задачей которой является точное определение тестовых пакетов для оценки систем обработки транзакций и баз данных, а также для распространения объективных, проверяемых данных в промышленности.

TPC публикует спецификации тестовых пакетов, которые регулируют вопросы, связанные с работой тестов. Эти спецификации гарантируют, что покупатели имеют объективные значения данных для сравнения производительности различных вычислительных

систем. Хотя реализация спецификаций оценочных тестов оставлена на усмотрение индивидуальных спонсоров тестов, сами спонсоры, объявляя результаты TPC, должны представить TPC детальные отчеты, документирующие соответствие всем спецификациям. Эти отчеты, в частности, включают конфигурацию системы, методику калькуляции цены, диаграммы значений производительности и документацию, показывающую, что тест соответствует требованиям атомарности, согласованности, изолированности и долговечности (ACID - atomicity, consistency, isolation, and durability), которые гарантируют, что все транзакции из оценочного теста обрабатываются должным образом.

Работой TPC руководит Совет Полного Состава (Full Council), который принимает все решения; каждая компания-участник имеет один голос, а для того, чтобы провести какое-либо решение требуется две трети голосов. Управляющий Комитет (Steering Committee), состоящий из пяти представителей и избираемый ежегодно, надзирает за работой администрации TPC, поддерживает и обеспечивает все направления и рекомендации для членов Совета Полного Состава и Управляющего Комитета.

В составе TPC имеются два типа подкомитетов: постоянные подкомитеты, которые управляют администрацией TPC, осуществляют связи с общественностью и обеспечивают выпуск документации; и технические подкомитеты, которые формируются для разработки предложений по оценочным тестам и распускаются после того, как их работа по разработке завершена.

Тесты TPC

TPC определяет и управляет форматом нескольких тестов для оценки производительности OLTP (On-Line Transaction Processing), включая тесты TPC-A, TPC-B и TPC-C. Как уже отмечалось, создание оценочного теста является ответственностью организации, выполняющей этот тест. TPC требует только, чтобы при создании оценочного теста выполнялись определенные условия. Хотя упомянутые тесты TPC не являются характерными тестами для оценки производительности баз данных, системы реляционных баз данных являются ключевыми компонентами любой системы обработки транзакций.

Следует отметить, что как и любой другой тест, ни один тест TPC не может измерить производительность системы, которая применима для всех возможных сред обработки транзакций, но эти тесты действительно могут помочь пользователю справедливо сравнивать похожие системы. Однако, когда пользователь делает покупку или планирует решение о покупке, он должен понимать, что никакой тест не может заменить его конкретную прикладную задачу.

Тест TPC-A

Выпущенный в ноябре 1989 года, тест TCP-A предназначался для оценки производительности систем, работающих в среде интенсивно обновляемых баз данных, типичной для приложений интерактивной обработки данных (OLDP - on-line data processing). Такая среда характеризуется:

- множеством терминальных сессий в режиме on-line
- значительным объемом ввода/вывода при работе с дисками
- умеренным временем работы системы и приложений
- целостностью транзакций.

Практически при выполнении теста эмулируется типичная вычислительная среда банка, включающая сервер базы данных, терминалы и линии связи. Этот тест использует одиночные, простые транзакции, интенсивно обновляющие базу данных. Одиночная транзакция (подобная обычной операции обновления счета клиента) обеспечивает простую, повторяемую единицу работы, которая проверяет ключевые компоненты системы OLTP.

Тест TPC-A определяет пропускную способность системы, измеряемую количеством транзакций в секунду (tps A), которые система может выполнить при работе с множеством

терминалов. Хотя спецификация TPC-A не определяет точное количество терминалов, компании-поставщики систем должны увеличивать или уменьшать их количество в соответствии с нормой пропускной способности. Тест TPC-A может выполняться в локальных или региональных вычислительных сетях. В этом случае его результаты определяют либо "локальную" пропускную способность(TPC-A-local Throughput), либо "региональную" пропускную способность (TPC-A wide Throughput). Очевидно, эти два тестовых показателя нельзя непосредственно сравнивать. Спецификация теста TPC-A требует, чтобы все компании полностью раскрывали детали работы своего теста, свою конфигурацию системы и ее стоимость (с учетом пятилетнего срока обслуживания). Это позволяет определить нормализованную стоимость системы ($/tpsA).

Тест TPC-B

В августе 1990 года TPC одобрил TPC-B, интенсивный тест базы данных, характеризующийся следующими элементами:

- значительный объем дискового ввода/вывода
- умеренное время работы системы и приложений
- целостность транзакций.

TPC-B измеряет пропускную способность системы в транзакциях в секунду (tpsB). Поскольку имеются существенные различия между двумя тестами TPC-A и TPC-B (в частности, в TPC-B не выполняется эмуляция терминалов и линий связи), их нельзя прямо сравнивать. На рис. 3.2 показаны взаимоотношения между TPC-A и TPC-B.

Тест TPC-C

Тестовый пакет TPC-C моделирует прикладную задачу обработки заказов. Он моделирует достаточно сложную систему OLTP, которая должна управлять приемом заказов, управлением учетом товаров и распространением товаров и услуг. Тест TPC-C осуществляет тестирование всех основных компонентов системы: терминалов, линий связи, ЦП, дискового в/в и базы данных.

TPC-C требует, чтобы выполнялись пять типов транзакций:

- новый заказ, вводимый с помощью сложной экранной формы
- простое обновление базы данных, связанное с платежом
- простое обновление базы данных, связанное с поставкой
- справка о состоянии заказов •справка по учету товаров

Среди этих пяти типов транзакций по крайней мере 43%должны составлять платежи. Транзакции, связанные со справками о состоянии заказов, состоянии поставки и учета, должны составлять по 4%. Затем измеряется скорость транзакций по новым заказам, обрабатываемых совместно со смесью других транзакций, выполняющихся в фоновом режиме.

База данных TPC-C основана на модели оптового поставщика с удаленными районами и товарными складами. База данных содержит девять таблиц: товарные склады, район, покупатель, заказ, порядок заказов, новый заказ, статья счета, складские запасы и история.

Обычно публикуются два результата. Один из них, tpm-C, представляет пиковую скорость выполнения транзакций (выражается в количестве транзакций в минуту). Второй результат, $/tpm-C, представляет собой нормализованную стоимость системы. Стоимость системы включает все аппаратные средства и программное обеспечение, используемые в тесте, плюс стоимость обслуживания в течение пяти лет.

Будущие тесты TPC

Совсем недавно (см. ComputerWorld-Moscow, N15, 1995) TPC объявил об отмене тестов TPC-A и TPC-B. Отныне для оценки систем будут применяться существующий тестовый пакет TPC-C, новые тесты TPC-D и TPC-E, а также два еще полностью не разработанных теста. Представленный в первом квартале 1995 года тест TPC-D предназначен для оценки производительности систем принятия решений. Для оценки систем масштаба предприятия во втором квартале 1995 года TPC должен был представить тест TPC-E и его альтернативный вариант, не имеющий пока названия. Кроме того, TPC продолжает разработку тестовых пакетов для оценки баз данных и систем клиент/сервер. Первые результаты, полученных с помощью этих новых методов, уже начали публиковаться.

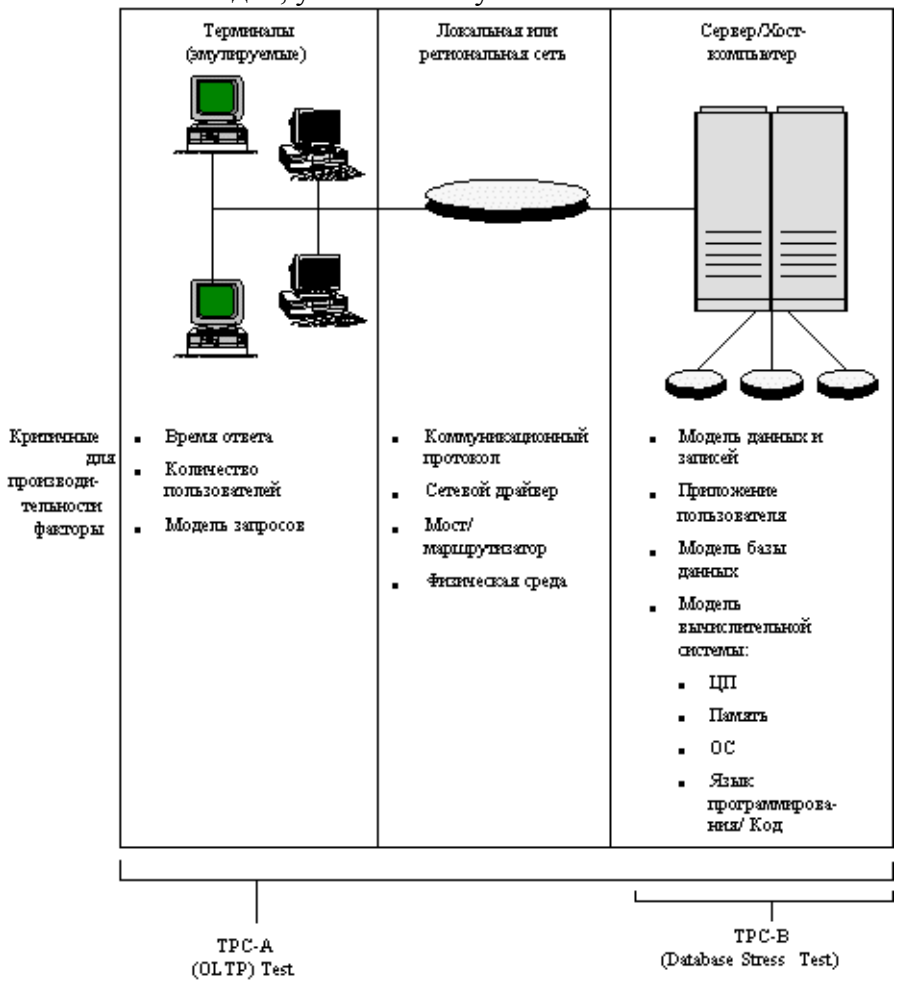


Рис. 3.2. Типовая среда обработки транзакций и соответствующие оценочные тесты TPC

AIM

Одной из независимых организаций, осуществляющей оценку производительности вычислительных систем, является частная компания AIM Technology, которая была основана в 1981 году. Компания разрабатывает и поставляет программное обеспечение для измерения производительности систем, а также оказывает услуги по тестированию систем конечным пользователям и поставщикам вычислительных систем и сетей, которые используют промышленные стандартные операционные системы, такие как UNIX и OS/2.

За время своего существования компания разработала специальное программное обеспечение, позволяющее легко создавать различные рабочие нагрузки, соответствующие уровню тестируемой системы и требованиям по ее использованию. Это программное обеспечение состоит из двух основных частей: генератора тестовых пакетов (Benchmark Generator) и нагрузочных смесей (Load Mixes) прикладных задач.

Генератор тестовых пакетов представляет собой программную систему, которая обеспечивает одновременное выполнение множества программ. Он содержит большое число отдельных тестов, которые потребляют определенные ресурсы системы, и тем самым акцентируют внимание на определенных компонентах, из которых складывается ее общая

производительность. При каждом запуске генератора могут выполняться любые отдельные или все доступные тесты в любом порядке и при любом количестве проходов, позволяя тем самым создавать для системы практически любую необходимую рабочую нагрузку. Все это дает возможность тестовому пакету моделировать любой тип смеси при постоянной смене акцентов (для лучшего представления реальной окружающей обстановки) и при обеспечении высокой степени конфигурирования.

Каждая нагрузочная смесь представляют собой формулу, которая определяет компоненты требуемой нагрузки. Эта формула задается в терминах количества различных доступных тестов, которые должны выполняться одновременно для моделирования рабочей нагрузки.

Используя эти две части программного обеспечения AIM, можно действительно создать для тестируемой системы любую рабочую нагрузку, определяя компоненты нагрузки в терминах тестов, которые должны выполняться генератором тестовых пакетов. Если некоторые требуемые тесты отсутствуют в составе генератора тестовых пакетов, то они могут быть легко туда добавлены.

Генератор тестовых пакетов во время своей работы "масштабирует" или увеличивает нагрузку на систему. Первоначально он выполняет и хронометрирует одну копию нагрузочной смеси. Затем одновременно выполняет и хронометрирует три копии нагрузочной смеси и т.д. По мере увеличения нагрузки, на основе оценки производительности системы, выбираются различные уровни увеличения нагрузки. В конце концов может быть нарисована кривая пропускной способности, показывающая возможности системы по обработке нагрузочной смеси в зависимости от числа моделируемых нагрузок. Это позволяет с достаточной достоверностью дать заключение о возможностях работы системы при данной нагрузке или при изменении нагрузки.

Очевидно, что сам по себе процесс моделирования рабочей нагрузки мало что дал бы для сравнения различных машин между собой при отсутствии у AIM набора хорошо подобранных смесей, которые представляют собой ряд важных для пользователя прикладных задач.

Все смеси AIM могут быть разделены на две категории: стандартные и заказные. Заказные смеси создаются для точного моделирования особенностей среды конечного пользователя или поставщика оборудования. Заказная смесь может быть тесно связана с определенными тестами, добавляемыми к генератору тестовых пакетов. В качестве альтернативы заказная смесь может быть связана с очень специфическим приложением, которое создает для системы необычную нагрузку. В общем случае заказные смеси разрабатываются на основе одной из стандартных смесей AIM путем ее "подгонки" для более точного представления определенной ситуации. Обычно заказные смеси разрабатываются заказчиком совместно с AIM Technology, что позволяет использовать многолетний опыт AIM по созданию и моделированию нагрузочных смесей.

К настоящему времени AIM создала восемь стандартных смесей, которые представляют собой обычную среду прикладных задач. В состав этих стандартных смесей входят:

1. Универсальная смесь для рабочих станций (General Workstation Mix) - моделирует работу рабочей станции в среде разработки программного обеспечения.
2. Смесь для механического САПР (Mechanical CAD Mix) моделирует рабочую станцию, используемую для трехмерного моделирования и среды системы автоматизации проектирования в механике.
3. Смесь для геоинформационных систем (GIS Mix) - моделирует рабочую станцию, используемую для обработки изображений и в приложениях геоинформацинных систем.
4. Смесь универсальных деловых приложений (General Business) - моделирует рабочую станцию, используемую для выполнения таких стандартных инструментальных средств, как электронная почта, электронные таблицы, база данных, текстовый процессор и т.д.
5. Многопользовательская смесь (Shared/Multiuser Mix) моделирует многопользовательскую систему, обеспечивающую обслуживание приложений для множества работающих в ней пользователей.
6. Смесь для вычислительного (счетного) сервера (ComputeServer Mix) - моделирует систему, используемую для выполнения заданий с большим объемом вычислений, таких как

маршрутизация PCB, гидростатическое моделирование, вычислительная химия, взламывание кодов и т.д.

7. Смесь для файл-сервера (File Server Mix) - моделирует запросы, поступающие в систему, используемую в качестве централизованного файлового сервера, включая ввод/вывод и вычислительные мощности для других услуг по запросу.
8. Смесь СУБД (RBMS Mix) - моделирует систему, выполняющую ответственные приложения управления базой данных.

Одним из видов деятельность AIM Technology является выпуск сертифицированных отчетов по результатам тестирования различных систем. В качестве примера рассмотрим форму отчета AIM Performance Report II - независимое сертифицированное заключение о производительности системы.

Ключевыми частями этого отчета являются:

- стоимость системы,
- детали конфигурации системы,
- результаты измерения производительности, показанные на трех тестовых пакетах AIM.

Используются следующие три тестовых пакета:

- многопользовательский тестовый пакет AIM (набор III),
- тестовый пакет утилит AIM (Milestone),
- тестовый пакет для оценки различных подсистем (набор II).

В частности, набор III, разработанный компанией AIM Technology, используется в различных формах уже более 10 лет. Он представляет собой пакет тестов для системы UNIX, который пытается оценить все аспекты производительности системы, включая все основные аппаратные средства, используемые в многопрограммной среде. Этот тестовый пакет моделирует многопользовательскую работу в среде разделения времени путем генерации возрастающих уровней нагрузки на ЦП, подсистему ввода/вывода, переключение контекста и измеряет производительность системы при работе с множеством процессов.

Для оценки и сравнения систем в AIM Performance Report II используются следующие критерии:

- Пиковая производительность (рейтинг производительности по AIM)
- Максимальная пользовательская нагрузка •Индекс производительности утилит
- Пропускная способность системы

Рейтинг производительности по AIM - стандартная единица измерения пиковой производительности, установленная AIM Technology. Этот рейтинг определяет наивысший уровень производительности системы, который достигается при оптимальном использовании ЦП, операций с плавающей точкой и кэширования диска. Рейтинг вездесущей машины VAX 11/780 обычно составляет 1 AIM. В отчетах AIM представлен широкий ряд UNIX-систем, которые можно сравнивать по этому параметру.

Максимальная пользовательская нагрузка - определяет "емкость" (capacity) системы, т.е. такую точку, начиная с которой производительность системы падает ниже приемлемого уровня для N-го пользователя (меньше чем одно задание в минуту на одного пользователя).

Индекс производительности утилит - определяет количество пользовательских нагрузок пакета Milestone, которые данная система выполняет в течение одного часа. Набор тестов Milestone многократно выполняет выбранные утилиты UNIX в качестве основных и фоновых заданий при умеренных пользовательских нагрузках. Этот параметр показывает возможности системы по выполнению универсальных утилит UNIX.

Максимальная пропускная способность - определяет пиковую производительность мультипрограммной системы, измеряемую количеством выполненных заданий в минуту. Приводящийся в отчете график пропускной способности системы показывает, как она работает при различных нагрузках.

Отчет по производительности разработан с использованием набора тестов AIM собственной разработки. В отличие от многих популярных тестовых пакетов, которые измеряют только производительность ЦП в однозадачном режиме и/или на операциях с плавающей точкой, тестовые пакеты AIM проверяют итоговую производительность системы и всех ее основных компонентов в многозадачной среде, включая ЦП, плавающую точку, память, диски, системные и библиотечные вызовы.

Синтетические ядра и натуральные тесты не могут служить в качестве настоящих тестовых пакетов для оценки систем: они не могут моделировать точно среду конечного пользователя и оценивать производительность всех относящихся к делу компонентов системы. Без такой гарантии результаты измерения производительности остаются под вопросом.