электронное научно-техническое издание
НАУКА и ОБРАЗОВАНИЕ
Эл № ФС 77 - 30569. Государственная регистрация №0421100025. ISSN 1994-0408

Виртуализация в высокопроизводительных вычислительных системах

 # 03, март 2011

автор: Романова А. О.

УДК 519.6

МГТУ им. Н.Э. Баумана
anastasia.o.romanova@gmail.com

Введение

Повышенный интерес к компьютерным технологиям виртуализации в настоящее время не случаен. Вычислительная мощь современных процессоров быстро растет. И вопрос даже не в том, на что эту мощь расходовать, а в том, что современная «мода» на двухъядерные и многоядерные системы, проникшая уже и в персональные компьютеры как нельзя лучше позволяет реализовать богатейший потенциал идей виртуализации операционных систем и приложений, выводя удобство пользования компьютером на качественно новый уровень. Технологии виртуализации становятся одним из ключевых компонентов (в том числе, и маркетинговых) в самых новых процессорах *Intel* [1] и *AMD* [2], в операционных системах от *Microsoft* [3] и ряда других компаний. В данной статье рассматривается, что представляют собой аппаратная и програмная виртуализации, а также идеи использования этих технологий в высокопроизводительных системах.

Чаще всего слово «виртуальный», вопреки его изначальному происхождению, вызывает ассоциации с чем-то невещественным, несуществующим на самом деле. Но смысл его в вычислительной технике иной – «виртуальные» объекты здесь всегда означают некие абстрактные интерфейсы, за которыми скрывается реальное оборудование.

В основе виртуализации лежит возможность одного компьютера выполнять работу нескольких компьютеров благодаря распределению его ресурсов по нескольким средам. С помощью виртуальных серверов можно разместить несколько операционных систем и несколько приложений в едином местоположении, в т. ч. удаленно. Таким образом, физические и географические ограничения перестают иметь какое-либо значение. Помимо энергосбережения и сокращения расходов, благодаря более эффективному использованию аппаратных ресурсов, виртуальная инфраструктура обеспечивает высокий уровень доступности ресурсов, более эффективную систему управления сервером, повышенную безопасность и усовершенствованную систему восстановления в критических ситуациях.

1. Классификация виртуальных систем

В настоящее время не существует единых стандартов классификации виртуальных систем. Наиболее полный перечень видов виртуализации представлен в электронной энциклопедии [4]. В данной работе предлагается классификация по уровням виртуализации, наиболее полно отображающая возможности использования того или иного типа систем.

Следует выделить два больших класса способов виртуализации – аппаратную виртуализацию, которую в западных источниках называют аппаратной поддержкой виртуализации, и программную виртуализацию.

Аппаратная виртуализация. Под аппаратной виртуализацией [5] понимают аппаратную поддержку, обеспеченную самим процессором. Аппаратная виртуализация – это общее название для двух независимых, но очень похожих технологий *Intel* и *AMD*, которые направлены на улучшение производительности процессора для общих задач виртуализации. Необходимость поддержки аппаратной виртуализации заставила производителей процессоров несколько изменить их архитектуру за счет введения дополнительных инструкций для предоставления прямого доступа к ресурсам процессора из гостевых систем [6]. В режиме поддержки виртуализации запускается специальное программное обеспечение, являющееся «легковесной» прослойкой между гостевыми операционными системами и оборудованием – монитор виртуальных машин (*Virtual Machine Monitor*, *VMM*), носящий также название гипервизор (*hypervisor*). Гипервизор - это слой программного обеспечения, работающий непосредственно поверх оборудования, позволяющий виртуализовать это оборудование (либо его подмножество) для программного обеспечения. Интерфейс монитора виртуальных машин полностью (либо частично) повторяет интерфейс виртуализуемой системы, благодаря чему, программное обеспечение не может определить факт присутствия гипервизора.

В качестве примера можно привести технологии *Intel VT-X*, *VT-D* и *AMDV*. Используя такие технологии, монитор виртуальных машин может передавать управление физическими устройствами непосредственной виртуальной машине. Очевидно, что без аппаратной поддержки, использование виртуализации бессмысленно.

Программная виртуализация как раз является тем, что обычно подразумевают, произнося термин виртуализация, представляя типичную

виртуальную машину [7]. Данный вид виртуализации модно разделить на следующие уровни:

- серверная виртуализация;
- паравиртуализация;
- виртуализация уровня операционной системы;
- виртуализация ресурсов;
- виртуализация прикладных приложений.

Ниже даны характеристики первых трех типов программной виртуализации, используемых в высокопроизводительных системах.

2. Серверная виртуализация

Серверная виртуализация, иногда употребляют термин **полная виртуализация**, представляет собой полную эмуляцию оборудования на уровне программного обеспечения. Схема серверной виртуализации представлена на рисунке 1.

Рисунок 1 — Серверная виртуализация

Рисунок 1 показывает, что данный вид виртуализации основан на применении гипервизора. Примером такой реализации является виртуальная

машина (ВМ). ВМ – это окружение, которое представляется для «гостевой» операционной системы, как аппаратное. Однако на самом деле это программное окружение, которое эмулируется программным обеспечением *Host*-системы.

Операционная система, управляющая реальным оборудованием и предоставляющая функции для доступа к нему, называется *Host*-системой, или основной, управляющей операционной системой. Основная операционная система загружается самостоятельно и не требует виртуальной машины для своей работы. Операционные системы, работающие в виртуальных машинах, называются гостевыми операционными системами. На одном физическом компьютере может быть запущена одна основная и много гостевых операционных систем.

Общая системная архитектура виртуальной машины построена на взаимодействии трех основных компонентов:

- приложение виртуальной машины;
- драйвер виртуальных машин;
- монитор виртуальной машины.

Приложение виртуальной машины – это обычное приложение, выполняющееся под управлением основной операционной системы. Приложение виртуальной машины имеет графический интерфейс и позволяет пользователю взаимодействовать с виртуальной машиной и гостевой операционной системой. Приложение является непереносимым компонентом виртуальной машины, поскольку разрабатывается для конкретной управляющей операционной системы и использует ее функции для отображения графического интерфейса и доступа к внешним устройствам. Как правило, для портирования виртуальной машины под другую основную операционную систему, необходимо полностью переписать приложение.

Приложение виртуальной машины построено по многопоточной технологии и поддерживает три следующих основных потока:

- поток виртуализации для передачи управления монитору и обмена информационными сообщениями с ним;
- графический поток для отображения видеобуфера гостевой операционной системы;
- поток *GUI* для работы пользовательского интерфейса и передачи событий от «мыши» и клавиатуры гостевой операционной системе.

Для каждой виртуальной машины запускается своя копия приложения виртуальной машины. Приложение виртуальной машины выполняет следующие основные функции:

- создание, удаление и конфигурирование виртуальных машин;
- включение, выключение и управление работой виртуальных машин;
- обеспечение интерфейса пользователя с гостевой операционной системой ввод с клавиатуры («мыши») и отображение экрана гостевой операционной системы;
- выделение памяти для виртуальной машины и загрузка (инициализация) монитора виртуальной машины;
- взаимодействие с физическими ресурсами компьютера через функции управляющей операционной системы (работа с жесткими и гибкими дисками, видеокартой, последовательными и параллельными портами и т.д.).

Драйвер виртуальных машин – это системный драйвер, работающий на уровне привилегий ядра управляющей операционной системы. Драйвер является шлюзом между приложением и монитором виртуальной машины, позволяющий им передавать управление и обмениваться информационными сообщениями между собой. Кроме того, драйвер выполняет функции

взаимодействия с управляющей операционной системой, такие как выделение и закрепление страниц памяти по физическим адресам. Драйвер виртуальной машины является непереносимым компонентом виртуальной машины. Для портирования виртуальной машины под другую основную операционную систему необходимо полностью переписать драйвер, используя средства этой операционной системы. Все виртуальные машины пользуются одной копией драйвера виртуальных машин.

Монитор виртуальной машины или гипервизор – это управляющий компонент виртуальной машины. Благодаря аппартаной поддержки виртуализации стала возможной эффективная работа гипервизора. Монитор не зависит от конкретной управляющей операционной системы и отвечает за создание виртуальной среды для исполнения гостевой операционной системы. Логическая схема монитора виртуальных машин представлена на рисунке 2. Монитор работает на уровне привилегий ядра управляющей операционной системы и реализует выбранную технологию виртуализации. Монитор включает в себя блок эмуляции процессора и внешних устройств. Он вынужден обращаться к приложению виртуальной машины за доступом к реальным внешним устройствам. Для каждой виртуальной машины запускается своя копия монитора виртуальной машины.

Монитор может взаимодействовать с приложением двумя способами:

- синхронно при помощи обмена информационными сообщениями через драйвер виртуальных машин;
- асинхронно при помощи разделяемых системных структур и участков памяти.

Монитор работает в изолированном от управляющей операционной системы контексте и поддерживает свои собственные системные таблицы *GDT*,

LDT, *IDT* и т.д. При переключении контекста между монитором и управляющей операционной системой выполняется операция сохранения одного контекста и загрузка другого. Переключение контекста напоминает процедуру переключения задач операционной системы, но включает в себя дополнительный набор данных. Также, монитор должен выявлять и перенаправлять управляющей операционной системе все прерывания от реальных внешних устройств, что показано на рисунке 3.

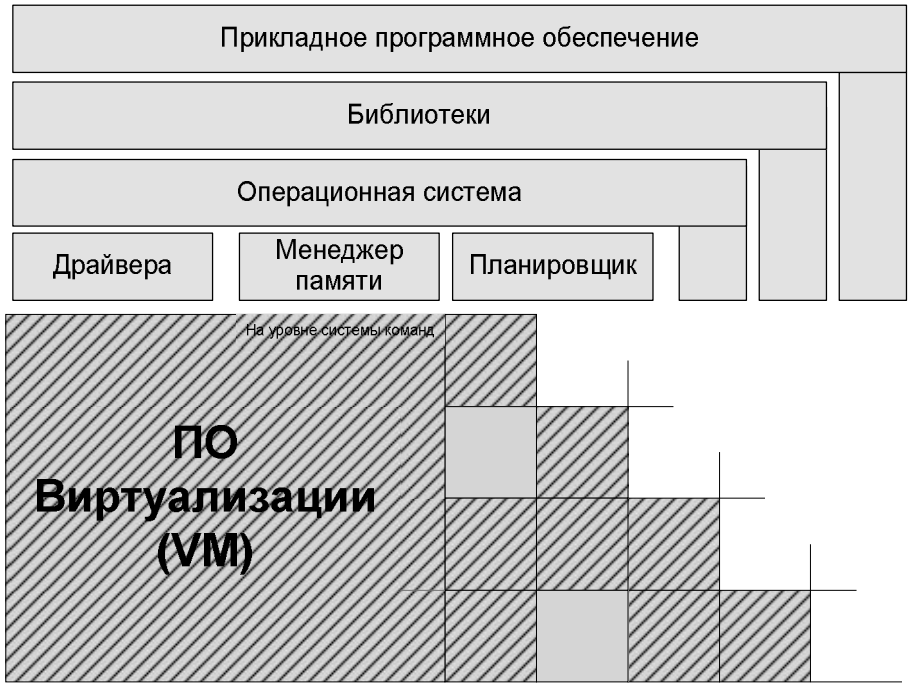

Рисунок 2 — Логическая схема монитора виртуальных машин

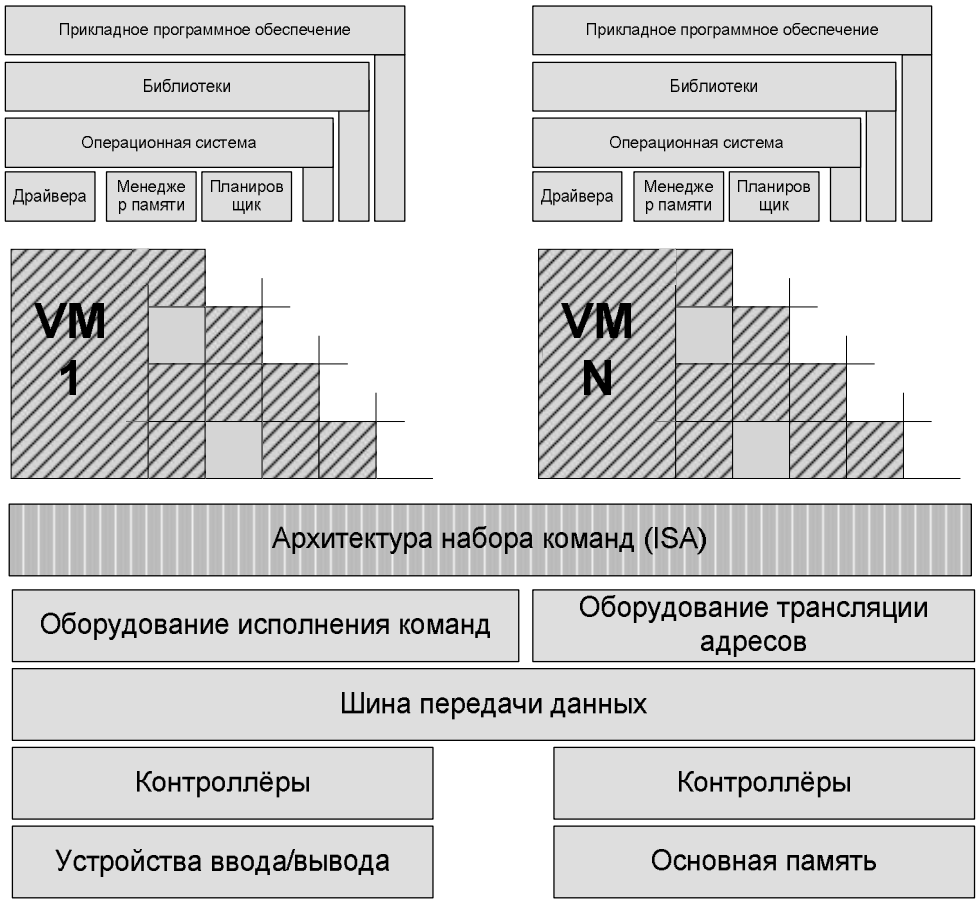

Рисунок 3 — Логическая схема эмулятора виртуальных машин

Подход к виртуализации, основанный на применении монитора виртуальных машин, предоставляет целый ряд преимуществ, по сравнению с другими технологиями виртуализации.

- Монитор предоставляют более цельную эмулируемую среду, позволяющую запускать «любое» программное обеспечение, в том числе и системного уровня, такого как операционные системы и т.п.
- Позволяет запускать любое число операционных систем одновременно на одном оборудовании.

- Предоставляет независимость гостевой операционной системы от оборудования, на котором он исполняется. Поскольку всё оборудование виртуализуется, вся система может быть динамически перенесена на другое оборудование, без потери работоспособности системы. На пример, для проведения технического обслуживания система может быть перенесена с одного мэйнфрейма на другой.
- Добавляет уровень изоляции для оборудования – не позволяя непроверенному коду исполняться на реальном оборудовании.

Общая схема логической организации монитора виртуальных машин изображена на рисунке 4.

Рисунок 4 — Логическая структура монитора виртуальных машин

Рассмотренный вид виртуализации позволяет создавать изолированные, масштабируемые гостевые виртуальные среды, поведение которых близко к поведению реальных серверов. Например, ВМ фирм *Parallels* и *VMware* с закрытым кодом и *Virtual Box* проекта *Open Source*. Именно аппаратная виртуализация чувствительна к наличию аппаратной поддержки.

Недостатками серверной виртуализации являются существенные накладные расходы, которые зависят от числа эмулируемых устройств и от

степени их похожести на физические. Технология плохо масштабируется – нет возможности эффективно и совместно использовать все доступные аппаратные ресурсы, а оперативная память физической машины фактически должна быть жестко разделена для того, чтобы предоставить каждой запущенной виртуальной свою собственную неразделяемую память. Кроме того, существенная доля памяти должна быть зарезервирована для накладных расходов самой системы виртуализации. Фактически на обычных серверах невозможно запустить более 10-15 виртуальных машин. Еще одним недостатком является жесткое разделение дискового пространства между ВМ. Обычно, файловая система для виртуальной машины расположена внутри файла управляющей ОС (или раздела диска) и уникальна для каждого экземпляра ВМ.

3. Паравиртуализация

Паравиртуализация – техника виртуализации [8], при которой гостевые операционные системы подготавливаются для исполнения в виртуализированной среде, для чего их ядро незначительно модифицируется (рисунок 5).

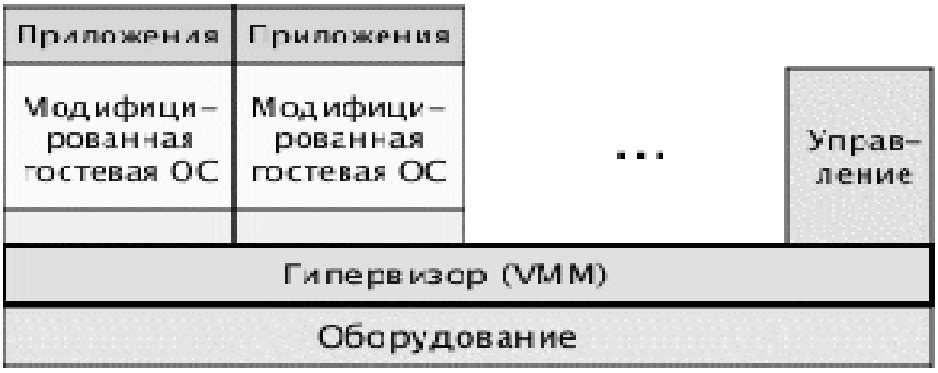

Рисунок 5 — К определению паравиртуализации

Из рисунка 5 видно, что схема паравиртуализации почти идентична схеме серверной виртуализации, за иключением изменения ядра гостевой операционной системы. Цель изменения заключается в сокращении доли времени выполнения операций в гостевой операционной системе. Это существенный недостаток, так как подобное изменение возможно лишь в случае, если гостевые системы имеют открытые исходные коды, которые можно модифицировать согласно лицензии. Преимуществом паравиртуализация является производительность близкая к производительности реальной, не виртуализированной системы. Как и при полной виртуализации, одновременно могут поддерживаться многочисленные различные операционные системы.

4. Виртуализация уровня операционной сиситмы

Виртуализация уровня ОС [9] также имеющая название виртуализации высокого уровня или совместной виртуализации ОС, виртуализирует сервера на уровне ядра операционной системы (рисунок 6).

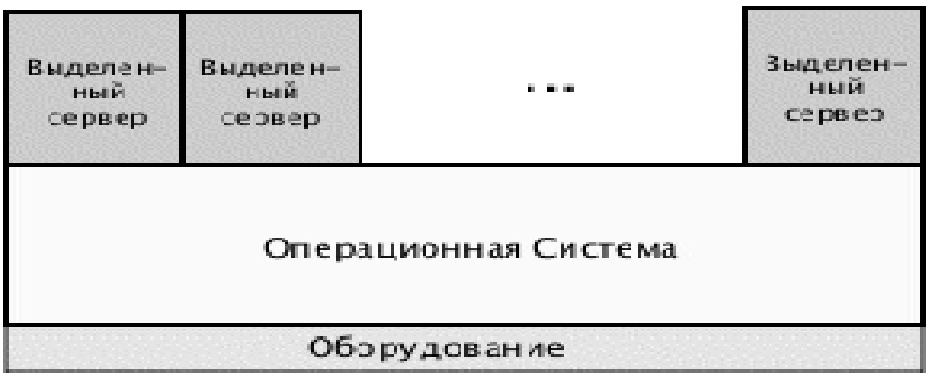

Рисунок 6 — К определению виртуализации уровня операционной системы

Виртуализация уровня ОС рационализирована для достижения лучшей производительности, управляемости и эффективности гостевой операционной системы. В основе лежит стандартная основная операционная система – это может быть *Windows* или *Linux-based* система. Далее идет слой виртуализации с собственной файловой системой и слой абстракции службы ядра, который обеспечивает изоляцию и безопасность ресурсов различных контейнеров.

Контейнер является виртуальной средой (ВС) исполнения – «высокоуровневой виртуализацией», которая обеспечивает каждой ВС свое собственное уникальное изолированное окружение: файлы и системные ресурсы, сервисы, системные способы связи с управляющей системой. Слой виртуализации позволяет каждому контейнеру функционировать как самостоятельный сервер. Наконец, в самом контейнере размещается приложение или рабочий процесс.

По сравнению с серверной виртуализацией, высокоуровневая виртуализация дает большие преимущества: гибкое управление памятью и файловым пространством, низкие затраты на переключение между виртуальными средами, возможность мгновенного получения всех ресурсов компьютера (рисунок 6).

Есть еще некоторые существенный отличия рассмотренных видов виртуализации. Например, в контейнере в любой момент возможен прямой административный доступ к файлам из управляющей ОС, тогда как в типовых реализациях виртуальной машины он возможен только по сети, если это сделал администратор соответствующей виртуальной машины. Наоборот, в ВМ имеется возможность сделать «снимок» состояния системы для дальнейшего возвращения к этому состоянию, а в контейнере это делается иначе – на уровне группы процессов, а не всего ядра.

5. Преимущества и проблемы виртуализации

Виртуализация вводит дополнительную гибкость в среду исполнения распределенных вычислений. Такая гибкость может быть использована для решения важных вопросов.

Консолидация нагрузки. В рамках предприятия, виртуализация в основном используется для сокращения числа физических серверов за счет консолидации нагрузки из нескольких систем на меньшее количество серверов. В зависимости от характера нагрузки и аппаратного обеспечения можно запускать 10 и более виртуальных сред на одном физическом сервере. Такое перераспределение ресурсов может быть с успехом использовано при работе большого набора не-ресурсоемких приложений, тогда как для распределенных вычислений необходимо выделять отдельные вычислительные ресурсы для достижения высокой производительности.

Восстановление после отказа. Современные кластерные системы из физических серверов используют стандартные менеджеры распределения ресурсов, такие как *Oracle Grid Engine* [10] или *Platform LSF* [11] для планирования задач. Проблемы возникают, когда выполнение задачи прерывается и оставляет систему в поврежденном состоянии. Производительность системы падает, так как необходимо перераспределить задачи по оставшимся ресурсам. Системные администраторы тратят значительное время на устранение этих проблем, в том числе на периодическую перезагрузку частей системы и их восстановление. Эта проблема не существует в виртуальной кластерной системе, так как виртуальные машины запускаются в четко определенном состоянии и не зависят от предыдущей работы. Аналогично, при запуске *MPI* [12] с помощью виртуальных машин одинаковое начальное состояние гарантировано.

Безопасность. Виртуализация позволяет запускать несколько задач на одном физическом сервере и обеспечивает барьер для предотвращения утечки информации между процессами или пользователями. Это обеспечивается такими свойствами виртуальной системы, как независимость и изолированность.

Портативность и управляемость. Виртуализация позволяет выполнять различные смешанные задачи в отдельных виртуальных машинах на одном физическом сервере. Это позволяет пользователям продолжать использовать существующие приложения при переходе на новые операционные системы или библиотеки, или легко сравнивать новые версии с предыдущими. Кроме того, некоторые конечные пользователи могут просто продолжать использовать старые версии операционных систем и приложений.

В случае необходимости смены аппаратного обеспечения, или изменения конфигурации управляющей системы, виртуальные среды могут быть быстро перемещены в любое окружение с поддержкой виртуализации.

Разработка и отладка программ. Виртуализация может быть использована в "обратной консолидации" для отладки масштабируемых приложений по мере их разработки. Например, правильно будет отлаживать новое приложения на небольшом виртуальном кластере, в таком случае можно провести корректное тестирование приложения в точно имитированной среде выполнения.

Гетерогенность. Высокопроизводительные системы из кластеров, по большей части, представляют собой совокупность гомогенных вычислительных узлов, с идентичной версией предварительно выбранной операционной системы. Хотя это значительно снижает административные затраты и позволяет поддерживаемому приложению обращаться к любому узлу, такая система является крайне негибкой. Рассмотрим, например, большие системы с широкой

базой пользователей и потенциально разнообразными требованиями. Реконфигурации узлов для удовлетворения этих потребностей пользователей не представляется возможной. Технология виртуализации может значительно улучшить эту ситуацию.

Несмотря на приведенные выше неоспоримые достоинства виртуализации, необходимо ответить на ряд вопросов:

- могут ли виртуализированные системы обеспечить производительность, сравнимую с физической?
- будет ли поведение программного обеспечения виртуализации и гостевой системы предсказуемым?
- будет ли настройка виртуальной системы дороже и дольше, нежели реальной?

6. Виртуализация в высокопроизводительных системах

Использование виртуализации может добавить новые возможности в высокопроизводительные вычислительные системы (*HPC*). Большинство таких систем стоится по принципу вычислительных кластеров. Вычислительный кластер состоит из совокупности компьютеров (персональных компьютеров или рабочих станций, называемых узлами кластера), объединенных локальной сетью в качестве коммуникационной среды. Общей физической оперативной памяти для узлов, как правило, не существует. Коммуникационная среда вычислительных кластеров позволяет узлам взаимодействовать между собой только посредством передачи сообщений. В целом, вычислительный кластер следует рассматривать как единую аппаратно-программную систему, имеющую единую коммуникационную систему, единый центр управления и планирования загрузки.

Более подробная классификация кластерных систем дана в [13]. Ниже описывается использование виртуализации в отдельном классе вычислительных кластеров – вычислительных сетях.

Грид-вычисления [14] – это форма распределённых вычислений, в которой суперкомпьютер представлен в виде кластера слабосвязанных компьютеров, соединённых с помощью сети, и работающих вместе, для выполнения огромного количества заданий. Эта технология применяется для решения научных задач, требующих значительных вычислительных ресурсов. Грид-вычисления используются также в коммерческой инфраструктуре для решения таких трудоёмких задач, как экономическое прогнозирование, сейсмоанализ, разработка и изучение свойств новых лекарств. Грид с точки зрения сетевой организации представляет собой согласованную, открытую и стандартизованную среду, которая обеспечивает гибкое, безопасное, скоординированное разделение вычислительных ресурсов и ресурсов хранения информации, которые являются частью этой среды, в рамках одной виртуальной организации.

Наиболее существенными препятствиями на пути применения Грид-технологий являются следующие проблемы:

- гетерогенность доступных вычислительных ресурсов (на уровне архитектур процессоров, операционных систем, сетевых настроек и т.д.);
- необходимость создания для многих ресурсоемких распределенных приложений целой системы из конфигурационных настроек, дополнительных служб, хранилищ данных и прочих компонентов информационно-вычислительной инфраструктуры.

Данные проблемы во многом могут быть преодолены при помощи использования рассматриваемых в данной работе видов виртуализации.

Совершенствование процессоров с поддержкой виртуализации на уровне ядра, интегрированных в материнскую плату гипервизоров, гипервизоров поддерживаемых распространенными операционными системами, развитие технологий быстрой передачи данных, удешевление Интернет-трафика делают эту задачу технически решаемой.

Хотя использование виртуализации в Грид-технологиях вызвало большой интерес, сравнительно мало внимания было уделено оценке применимости виртуальных сред к кластерам. Развертывание виртуальных кластеров различных топологий требует возможности размещения многих конфигураций ВМ так, чтобы правильно организовать совместное использование ресурсов. В связи с необходимостью настойки большого числа виртуальных машин, развертывание виртуального кластера может быть сильно дороже и дольше, нежели настройка физического.

Облачные вычисления. Облачные или рассеянные вычисления [15] представляют собой технологию обработки данных, в которой компьютерные ресурсы и мощности предоставляются пользователям как Интернет-сервисы. Пользователь не должен заботиться об инфраструктуре, операционной системе и собственно программном обеспечении, к которому предоставляется доступ. Облачная обработка данных – это парадигма, в рамках которой информация постоянно хранится на серверах в Интернете и временно кэшируется на клиентской стороне.

Для облачных вычислений управляющей проблемой является неравномерность запроса ресурсов со стороны клиентов. Для сглаживания неравномерности предоставления Интернет-сервисов, т. е. распределения ресурсов между реальным аппаратным обеспечением и облачным программным обеспечением, используют промежуточный слой серверной виртуализации (часто, виртуализации уровня ОС). Серверы, выполняющие приложения

виртуализируются, и балансировка нагрузки осуществляется как средствами програмного обеспечения, так и средствами распределения виртуальных серверов по реальным. Виртуализация в облаках позволяет получить доступ к потенциально большому количеству вычислительных мощностей без больших расходов на запуск крупномасштабных вычислений.

По мнению одного из ключевых сотрудников *VMWare* Джо Эндрюса (*Joe Andrews*) виртуализация – это фундамент для облачных вычислений [16, 17].

Первое требование при переходе к облачным вычислениям заключается в том, чтобы абстрагировать уровень физических ресурсов, то есть виртуализировать все аппаратное обеспечение, которое имеется в компании, превратить его в абстрагированные ИТ-ресурсы. Именно такие виртуализированные ресурсы можно перенести в облака, а выполняемые задачи трансформировать в облачные вычисления.

Ранее заказчик виртуализировал компьютеры, хранилища, сети, превращая их в виртуальную машину, которую системный администратор мог использовать для более эффективного распределения рабочей нагрузки и, тем самым, более оптимального использования ИТ-бюджетов. При переходе к облачным вычислениям абстракция происходит на еще более высоком уровне: мы объединяем оборудование с виртуализированными приложениями, позволяя ИТ-администратору предоставлять информационные технологии в качестве услуги для потребителей. Таким образом, можно создать каталог ИТ-услуг, и конечный пользователь будет выбирать то, что ему нужно, исходя из его запросов на аппаратное и программное обеспечение. Такая концепция называется *IT-as-a-Service*. Концепция базируется на слое *IaaS*, в который *VMware* добавляет уровни *PaaS* и *SaaS*, что позволяет пользователю брать вычислительные ресурсы самого разного уровня в аренду.

Таким образом, стремление максимально использовать вычислительные ресурсы облаков, являются одной из наиболее распространенных причин, по которой в *HPC* проявляется интерес к технологиям виртуализации.

7. Производительность виртуальных систем

Главной причиной отказа от использования виртуализации, является потеря производительности виртуальной вычислительной среды по сравнению с физической средой. В то же время, имеются примеры, в которых эти потери несущественны. Так виртуальная машина *Parallels* позволяет пользователям выполнять вычисления на виртуальных графических процессорных устройствах (ГПУ) с производительностью, практически равной производительности физических ГПУ [18]. Такой результат достигается, благодаря использованию технологию *Intel VT-d*, которая позволяет напрямую назначать выделенные графические процессоры виртуальным машинам.

Ранее были произведены некоторые оценки производительности виртуальных высокопроизводительных систем. В статье [19] даются результаты, показывающие, что хорошая производительность в виртуальной среде достижима. В Массачусетском Технологическом университете в 2008 году, было проведено сравнение производительности виртуальных систем с реальными на тестовом наборе задач *LAPACK* [20]. В качестве программной виртуализации был выбран *Xen* [21], в качестве операционных систем был выбран *Linux* с ядром 2.6.16, причем гостевые системы имеют поддержку *SMP* [22] и их ядра модифицированы версией *Xen* 3.0.4. Исследования проводились для выделенной памяти в 256 и 768 MB. Исходя из специфичной реализации *Xen*, исследования были проведены в двух виртуальных машинах, *Dom0* – основная, имеет больший приоритет при распределении ресурсов и запускается

при старте самого виртуального приложения, *DomU* – второстепенная. Используемые тесты *LAPACK* приведены в таблице 1.

Таблица 1. Список тестов LAPACK

Название задачи	Операция	Описание
GEmmNN	$C=\alpha AB+\beta C$	Произведение обычных плотных матриц
GEmmNT	$C=\alpha AB^T+\beta C$	Произведение нетранспанированой и транспонированой плотных матриц
GEmmTN	$C=\alpha A^TB+\beta C$	Произведение транспонированой и нетранспонированой плотных матриц
GEmmTT	$C=\alpha A^TB^T+\beta C$	Произведение транспонированных плотных матриц
GEmvN	$C=\alpha Ax+\beta y$	Векторо-матричное произведение
GEmvT	$C=\alpha A^Tx+\beta y$	Произведение вектора и трасноспонированой матрицы
GER	$C=\alpha xy^T+\beta A$	Произведение первого ранга

Некоторые результаты исследования представлены на рис. 7. Рисунок показывает незначительную разницу в производительности виртуальной и реальной систем. Эти результаты подтверждают исследования автора, сделанные в рамках магистерской диссертации [23]. Исследования проводились с использованием виртуальных машин фирм *VMWare* и *Parallels*, установленных и настроенных на операционной системе *Windows XP*. В качестве гостевых ОС были выбраны *Microsoft Windows XP* и *Ubuntu Linux*.

Тестовый набор однородных жестких дифференциальных уравнений решался в программных комплексах *MatLab* и *Mapple*, поддерживаемых каждой из гостевых систем. Тестовый набор уравнений приведен в таблице 2.

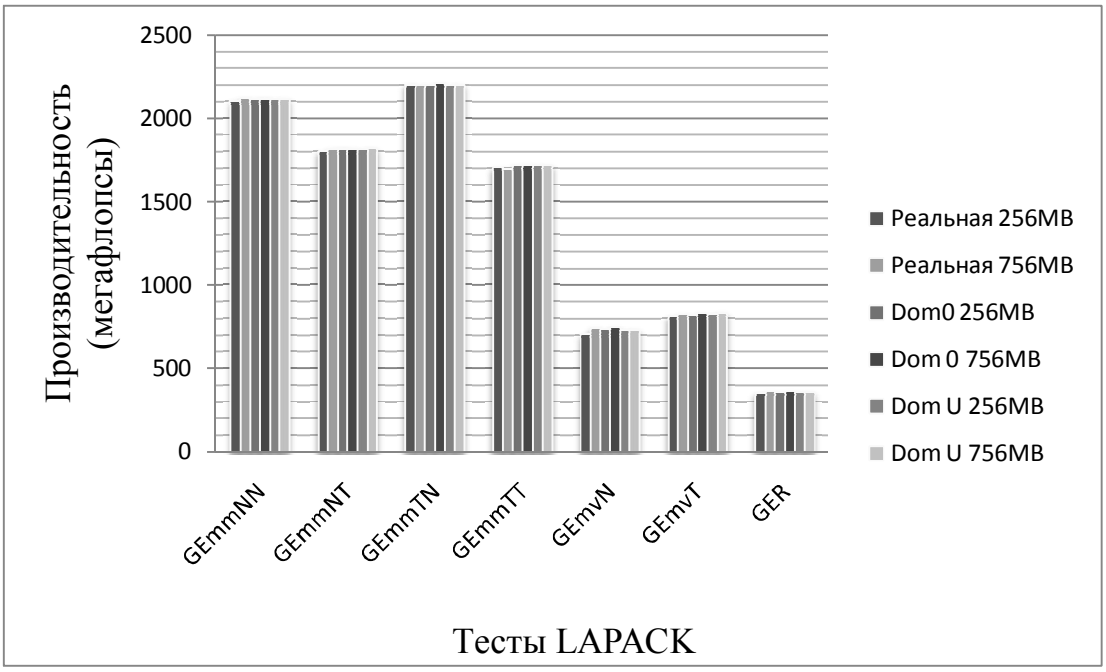

Рисунок 7 — Производительность реальной и виртуальной систем

Таблица 2. Тестовый набор систем обыкновенных дифференциальных уравнений

Система ОДУ	Уравнение системы	Описание системы
ОДУ 1	$x_1' = a \cdot x_1 + b \cdot x_2$ $x_2' = -b \cdot x_1 + a \cdot x_2$	Линейная система с сильно осциллирующим решением
ОДУ 2	$x_1' = a \cdot x_1 + b \cdot x_2$ $x_2' = -b \cdot x_1 + a \cdot x_2$	Линейная система с мнимыми частями в

	$x_3' = c \cdot x_3 + d \cdot x_4$ $x_4' = -d \cdot x_3 + c \cdot x_4$	собственных значениях и сильно осциллирующим решением
ОДУ 3	$x_1' = 77.27 \cdot (x_2 + x_1 \cdot (1 - 8.375 \cdot 10^{-6} \cdot x_1 - x_2))$ $x_2' = \frac{x_3 - (1 + x_1) \cdot x_2}{77.27}$ $x_3' = 0.161 \cdot (x_1 - x_3)$	*OREGO* – орегонатор, модель с периодическим решением, описывающая реакцию Белоусова-Жаботинского
ОДУ 4	$x_1' = -1.71 \cdot x_1 + 0.43 \cdot x_2 + 8.32 \cdot x_3 + 0.0007$ $x_2' = 1.71 \cdot x_1 - 8.75 \cdot x_2$ $x_3' = -10.03 \cdot x_3 + 0.43 \cdot x_4 + 0.035 \cdot x_5$ $x_4' = 8.32 \cdot x_2 + 1.71 \cdot x_3 - 1.12 \cdot x_4$ $x_5' = -1.745 \cdot x_5 + 0.43 \cdot x_6 + 0.43 \cdot x_7$ $x_6' = -280 \cdot x_6 \cdot x_8 + 0.69 \cdot x_4 + 1.71 \cdot x_5 - 0.43 \cdot x_6 + 0.69 \cdot x_7$ $x_7' = 280 \cdot x_6 \cdot x_8 - 1.81 \cdot x_7$ $x_8' = -280 \cdot x_6 \cdot x_8 + 1.81 \cdot x_7$	*HIRES* – модель химической реакции с участием восьми реагентов

Результаты исследований приведены в таблице 3, которая показывает, что потеря производительности не превышают 10%. Времена в таблице указаны в секундах.

Таблица 3. Сводная таблица времени решения тестовых систем ОДУ

Система ОДУ	Гостевая ОС				Основная ОС		Временные потери, %
	Windows		*Linux*				
	MatLab	*Maple*	*MatLab*	*Maple*	*MatLab*	*Maple*	
ОДУ 1	0,034	0,052	0,033	-	0,031	0,047	8
ОДУ 2	0,127	-	0,124	-	0,121	-	5

ОДУ 3	0,014	0,034	0,013	0,032	0,012	0,031	9
ОДУ 4	0,008	0,009	0,007	0,008	0,007	0,008	10

Также заслуживает внимания оценка производительности Грид системы, состоящей из виртуальных кластеров [24]. Данные исследования были проведены в университете Чикаго в 2009 году и включали в себя эксперимент для оценки производительности OSG виртуального кластера при работе с различными классами приложений.

Все эксперименты проводились на части кластеров национальной лабратории *Argonne*, где каждый физический сервер имеет сервер 500 *MHz Intel PIII CPUs*, 512 *MB RAM* и 9 GB свободного места на жестком диске. Все машины соединены с помощью 100 *Mbps LAN*. В качестве виртуализационного программного обеспечения был выбран *Xen*, гостевые системы созданы на основе ядра *Linux* 2.6.11.

В ходе этих испытаний, сравнивалось время выполнения приложения на кластерах, состоящих из физических узлов и на виртуальных кластерах, работающих на физических серверах. В изначальной конфигурации виртуальные машины, развернутые на кластере, имели доступ к меньшему количеству ресурсов (памяти и процессора), а некоторые ресурсы были отнесены к гипервизорным функциям (*Dom0*). Такие тесты дают более точное определение работоспособности приложения в виртуальной машине, так как при такой конфигурации учитываются накладные расходы на запуск виртуальной машины.

Для оценки производительности использовался *Fast Ocean Atmosphere Model* (*FOAM*) [25] тест, который использует связанные методы моделирования климата для решения вопросов, которые требуют многолетнего мониторинга.

FOAM представляет собой *MPI* тест с параллелизмом по данным. Результаты выполнения теста приведены на рис. 8.

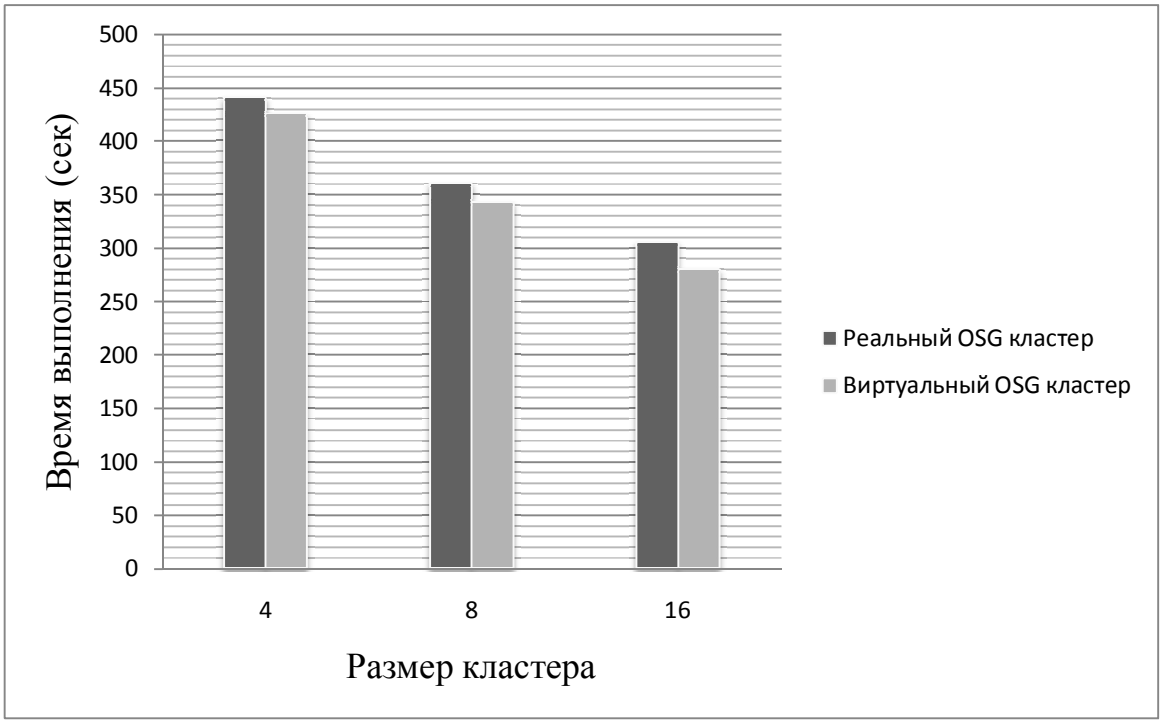

Рисунок 8 — Результаты выполнения теста на виртуальном и физическом кластерах

Таблица 4 показывает разницу в производительности между выполнением теста на виртуальном и физическом кластере, которая не превышает 5%. Это связано с тем, что общая доля времени, затраченная на коммуникации небольшая.

Таблица 4. Потеря производительности на виртуальном кластере

4 сервера	8 серверов	16 серверов

4,55%	4,05%	4,10%

Выводы

Виртуализация является гибким механизмом, помогающим в решении большого числа задач. Представленные в статье типы виртуализации могут быть широко использованы, как для решения простых пользовательских проблем, так и в высокопроизводительных системах. Несмотря на оставшиеся нерешенные вопросы, преимущества виртуализации неоспоримы. Использование данной технологии открывает новые возможности для высокопроизводительных систем. Принимая во внимание еще и небольшие потери в производительности, по сравнению с реальными системами, можно смело говорить о виртуализации *HPC*.

Литература

1. Радхакришна Х. Технология Intel® Virtualization Technology for Directed I/O (Intel® VT-d) //intel.com:Intel в России. 2011. URL http://www.intel.com/cd/corporate/europe/emea/rus/update/360260.htm (дата обращения:28.02.2011).
2. AMD Releases "Pacifica" Specification For AMD64 Technology //amd.com. 2011. URL http://www.amd.com/us/press-releases/Pages/Press_Release_98372.aspx (дата обращения:28.02.2011).
3. Microsoft® Hyper-V™ Server 2008 R2 //microsoft.com. 2011. URL http://www.microsoft.com/hyper-v-server/en/us/default.aspx (дата обращения:28.02.2011).
4. Википедия Виртуализация //ru.wikipedia.org:электронная энциклопедия. 2011. URL http://ru.wikipedia.org/wiki/Виртуализация (дата обращения:28.02.2011).

5. Gleb Reys What Hardware Virtualization Really Means //desktop-virtualization.com. 2011. URL http://www.desktop-virtualization.com/2008/05/14/what-hardware-virtualization-really-means/ (дата обращения:28.02.2011).

6. Самойленко А. Технологии аппаратной виртуализации //ixbt.com. 2011. URL http://www.ixbt.com/cm/virtualization-h.shtml (дата обращения:28.02.2011).

7. Wikipedia Virtual machine //en.wikipedia.org:Free Encyclopedia. 2011. URL http://en.wikipedia.org/wiki/Virtual_machine (дата обращения:28.02.2011).

8. Кухар А. Паравиртуализация //www.vmgu.ru:Виртуализация vSphere, Hyper-V, XenServer и Red Hat.2011. URL http://www.vmgu.ru/articles/Paravirtualizatsiya (дата обращения:28.02.2011).

9. Виртуализация ОС //parallels.com:Виртуальные машины Mac и виртуальные ПК. 2011. URL http://www.parallels.com/ru/products/pvc46/info/virtualization/ (дата обращения:28.02.2011).

10. Oracle Grid Engine //oracle.com. 2011. URL http://www.oracle.com/us/products/tools/oracle-grid-engine-075549.html (дата обращения:28.02.2011).

11. The HPC Workload Management Standard //platform.com. 2011. URL http://www.platform.com/workload-management/high-performance-computing (дата обращения:28.02.2011).

12. The Message Passing Interface (MPI) standard //www.mcs.anl.gov. 2011. URL http://www.mcs.anl.gov/research/projects/mpi/ (дата обращения:28.02.2011).

13. Карпенко А.П. Параллельные вычисления //bigor.bmstu.ru:База и Генератор Образовательных Ресурсов. 2011. URL

http://bigor.bmstu.ru/?cnt/?doc=Parallel/ch010104.mod/?cou=Parallel/base.cou (дата обращения:28.02.2011).

14. Демичев А.П., Ильин В.А., Крюков А.П. Введение в грид-технологии. Москва, 2007. 87 с. (Препринт НИИЯФ МГУ - 11/832).

15. Eric Knorr, Galen Gruman What cloud computing really means //infoworld.com. 2011. URL http://www.infoworld.com/d/cloud-computing/what-cloud-computing-really-means-031?source=footer (дата обращения:28.02.2011).

16. VMWare //vmware.com. 2011. URL http://vmware.com (дата обращения:28.02.2011).

17. Джо Эндрюс Виртуализация – это фундамент для облачных вычислений // pcweek.ua:новостной сайт. 2011. URL http://www.pcweek.ua/themes/detail.php?ID=131028 (дата обращения:28.02.2011).

18. Contain Your Costs,Control Your Grid //parallels.com:Виртуальные машины Mac и виртуальные ПК. 2011. URL http://www.parallels.com/products/extreme/audience/grid-computing/ (дата обращения:28.02.2011).

19. Lamia Youseff, Keith Seymour, Haihang You, Jack Dongarra, Rich Wolski. The Impact of Paravirtualized Memory Hierarchy on Linear Algebra Computational Kernels and Software. HPDC'08, June 23-27, 2008, Boston, Massachusetts, USA.

20. LAPACK – Linear Algebra PACKage //netlib.org. 2011. URL http://www.netlib.org/lapack/ (дата обращения:28.02.2011).

21. Xen //xen.org. 2011. URL http://www.xen.org/ (дата обращения:28.02.2011).

22. Википедия Симметричное мультипроцессирование //ru.wikipedia.org:электронная энциклопедия. 2011. URL

http://ru.wikipedia.org/wiki/симметричное_мультипроцессирование (дата обращения:28.02.2011).

23. Романова А.О. Исследование средств виртуализации для CAD, CAE и PDM систем САПР: Дис. ... магистр техники и технологии. Москва, 2010. 142 с.

24. Virtual Clusters for Grid Communities, I. Foster, T. Freeman, K. Keahey, D. Scheftner, B. Sotomayor, X. Zhang. CCGRID 2006, Singapore. May 2006.

25. Jacob, R., C. Schafer, I. Foster, M. Tobis, and J. Anderson. Computational Design and Performance of the Fast Ocean Atmosphere Model, Version One. in International Conference on Computational Science. 2001: Springer-Verlag.