УДК 004.272.2+004.75+544.18

ВИРТУАЛЬНЫЕ ВЫЧИСЛИТЕЛЬНЫЕ СРЕДЫ: ИСПОЛЬЗОВАНИЕ НА GRID ПОЛИГОНАХ

В.М. Волохов, Д.А. Варламов, Н.Ф. Сурков, А.В. Пивушков

VIRTUAL COMPUTING ENVIRONMENTS: UTILIZATION ON THE GRID POLYGONS

V.M. Volokhov, D.A. Varlamov, N.F. Surkov, A.V. Pivushkov

Статья посвящена анализу применимости различных технологий виртуализации в распределенных GRID средах, включая создание виртуальных вычислительных и сетевых ресурсов и динамически формируемых виртуальных программных <контейнеров»с поддержкой различных параллельных программных сред. Разработанные методы позволяют запускать различные параллельные приложения на произвольных ресурсных узлах GRID полигонов. Приведен пример подготовки и запуска квантовохимического пакета G AMESS-US в распределенной среде gLite на ресурсах полигона EGEE-RDIG.

Ключевые слова: виртуализация, распределенные среды, параллельные вычисления, G A MESS

The article is devoted to the analysis of applying of various virtualization technologies in the distributed GRID environment including creation of the virtual computational and network resources as well as dynamically forming virtual program ^containers» with support of various parallel program environments. The worked out methods make it possible to start different parallel supplements on random resource units of GRID polygons. As a case in point, the authors give preparation and launching of quantum-chemical GAMESS-US package in the distributed gLite environment on the basis of the polygon EGEE-RDIG resources.

Keywords: virtualization, distributed environments, parallel computing, GAMESS

Введение

Распределенные вычисления или GRID вычисления являются результатом развития научной концепции создания глобальных вычислительных ресурсов, доступных в любых объемах в любой точке планеты посредством высокоскоростных сетей, в том числе через Интернет. Суть GRID технологий заключается в организации вычислений путем объединения через сеть мощностей многих компьютеров, географически удаленных друг от друга. Сейчас эта технология используется в основном для проведения научных расчетов, при этом ее потенциал оценивается очень высоко. По сути, она имеет стратегический характер, и в будущем GRID вычисления претендуют на то, чтобы стать одним из основных двигателей развития информационных технологий, подобно тому, как такими двигателями стали персональный компьютер и Интернет. Решение многих задач вычислительной химии, таких как моделирование молекулярных структур и их динамики, поведение и энергетика сложных химических реакций, наномоделирование невозможно без применения GRID вычислений.

Например, время расчета поведения нанотрубок, легированных металлами, состоящих из $\eta \cdot 10^3$ атомов, может достигать 3 - 4 лет для однопроцессорных систем.

На основе опыта работы в различных распределенных вычислительных средах [1 - 3] авторами был сделан вывод, что наиболее существенными препятствиями на пути применения GRID технологий в вычислительной химии 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 применением интенсивно развиваемых в последнее время технологий виртуализации вычислений на уровне: (а) распределенных ресурсов, (б) виртуальных <контейнеров-приложений> и (в) полнофункциональных виртуальных машин, передаваемых удаленным ресурсам на исполнение. Появление процессоров с поддержкой виртуализации на уровне ядра, интегрированных в материнскую плату гипервизоров, поддерживаемых распространенными операционными системами, развитие технологий быстрой передачи данных, удешевление Интернет-трафика привели в последние два года к <взрывному>росту исследований и рынка ПО виртуализации.

Термин <виртуализация>здесь и далее нами используется в двух основных смыслах: виртуализация вычислительного GRID ресурса и виртуализация вычислительного объекта (ОС, приложение и т.д.), перемещаемого в GRID среде. Потребность в виртуализации для распределенных вычислительных сред продиктована необходимостью создания и поддержки стандартных механизмов взаимодействия между пользователями и вычислительными ресурсами (сервисами), одинаковых со стороны ресурса (поставщика сервисов) и со стороны пользователя.

Спектр использования средств виртуализации и виртуальных машин сегодня очень широк: объединение различных вычислительных и управляющих сервисов на единых физических ресурсах, возможность разработки и тестирования прикладного ПО в различных средах, миграция приложений и служб - все это постоянно продвигает технологии виртуализации вперед. Возможность снизить затраты на приобретение дополнительных физических серверов, электропитание и охлаждение для них, возможность виртуальной изоляции чрезмерно ресурсоемких или потенциально опасных приложений, мобильность и управляемость виртуальных ресурсов и приложений создают ощутимые преимущества для вычислительных центров и обеспечивают создание виртуальной вычислительной инфраструктуры в относительно небольшие сроки.

В настоящее время более 40 фирм и сообществ программистов развивают различные направления виртуализации: разработка виртуальных платформ, виртуализация приложений и служб, средства разработки, управления и сопровождения и т.д. Все типы представлены как коммерческими разработками, так и Open Source проектами. По данным консалтинговых компаний до 40% компаний среднего и крупного бизнеса уже используют или готовы использовать те или иные технологии виртуализации. Применение методов виртуализации ресурсов и приложений для проведения различных расчетов за этот год вышло на лидирующие позиции в области информационных технологий. По оценкам агентства Gartner (http://www.gartner.com) технология виртуализации вычислительных объектов и хранилищ данных совместно с виртуализацией приложений поднялась в ежегодной десятке ИТ-технологий с 5-й позиции в top-10 2008 года на первую в top-10 (http://www.cnews.ru/news/top/index.shtml72008/10/27/324944) на 2009 год. Для эффектив-

ного развития GRID технологий в области вычислительной химии авторами были проведены интенсивные исследования применимости различных методов виртуализации к GRID вычислениям.

1. Применение технологий виртуализации в распределенных средах

Авторы поставили целью изучить и применить на реальных расчетах в области вычислительной химии ряд технологий виртуализации, включая:

- создание и применение виртуальных машин на существующих ресурсных узлах входящих в различные распределенные среды. Это позволит: расширить функциональность ресурсных узлов; решать различные прикладные задачи на базе разных программных архитектур; разделять ресурсы; повысить безопасность;

- адаптация прикладного программного обеспечения для работы в составе виртуальной динамически формируемой параллельной среды;

- создание образа виртуальной машины со встроенным прикладным пакетом сложной конфигурации для запуска на удаленных ресурсах с использованием GRID инфраструктуры.

1.1. Создание виртуальных вычислительных ресурсов в виде ВМ (виртуальных машин) на базе существующих физических узлов

Для работы многих приложений требуется создание целой системы из служб, сетей, хранилищ данных и прочих компонентов современной информационно-вычислительной инфраструктуры. Создание такой системы иногда невозможно на ресурсном узле из-за несовместимости ПО. Для ряда пакетов ПО и сервисов нужно создавать комплексные среды с необходимым набором приложений и политиками безопасности. Ключевыми требованиями являются скорость и простота предоставления таких сред, их тщательная изоляция друг от друга, квотирование вычислительных ресурсов для каждой среды, независимость от базовых настроек узла. Зачастую все это необходимо делать без прерывания работы узлов и остановки вычислительной среды, особенно в рамках <production farms>, т.е. ресурсных узлов, не допускающих остановок и переконфигурирования. Данная задача может быть решена путем установки и настройки виртуальных машин на существующих ресурсных узлах. Они будут обслуживать сервисы и конкретные входящие приложения с наибольшей эффективностью и практически полной безопасностью для инфраструктуры ресурсного узла (включая ОС и прикладное ПО).

Идея использования виртуализации для распределенных вычислений не является новой (в России известны проекты <Дубна-Грид>, <SKIF@Home> и т.п.), Однако увеличение производительности узлов и развитие архитектуры процессоров и соответствующего ПО позволило на новом уровне подойти к решению данной проблемы (например, с использованием преимуществ нетипичных параллельных сред в условиях кластеров). Кроме того, авторам неизвестно применение виртуальных машин работающих в роли ресурсных узлов для конкретных распределенных ресурсоемких приложений в Ъбласти вычислительной химии.

При создании виртуальных машин (ВМ), в качестве ресурса, пользователю может быть предоставлена изолированная виртуальная вычислительная среда, по своим свойствам практически не уступающая физическому серверу, в которой может быть предоставлен любой

вычислительный сервис. Приложения, реализованные в ВМ, в этом случае не зависят как от самой операционной системы, в которой выполняется ВМ, так и от ее окружения. Еще одним преимуществом такой виртуализации является полное отделение конкретной службы или программы от внешней среды, а также и внешней среды от нее. Это возможно благодаря использованию дополнительного слоя программного обеспечения - виртуального оборудования, позволяющего выполнять обычное приложение так, как если бы оно было запущено на отдельном компьютере. Возможно создать образ виртуальной машины с предустановленной операционной системой и полностью сконфигурированными приложениями, нацеленной на решение конкретных задач. Этот образ может быть запущен на удаленном узле. Для этого не требуется перенастройки физического узла под конкретные задачи. Возможность подготовки и запуска образа виртуальной машины существенно облегчает адаптацию прикладного ПО для работы в распределенных средах.

Большим преимуществом виртуальной машины является возможность ее переноса, при необходимости, с одного сервера на другой и толерантность к аппаратным сбоям. В случае поломки одного физического узла копия виртуальной машины легко запускается на другом. Более того, в настоящее время ПО виртуализации позволяет перенести работающую виртуальную машину с одного физического узла на другой.

Для создания и применения виртуальных ресурсов авторами было протестировано несколько свободно распространяемых гипервизоров виртуальных машин, среди которых полнофункциональные среды Xen (http://xen.org, разработка XenSource, Inc.), VirtualBox (http://www.virtualbox.org, разработка Innotek Inc. - подразделения Sun Microsystems), VMware (http://www.vmware.com, разработка VMware Inc. - свободно распространяемая версия с сокращенной функциональностью), а также отдельной виртуальной машины - QEMU (http://beliard.org/qemu).

Изучение выбранного ПО показало схожесть функциональных характеристик указанных пакетов для поставленных в проекте задач и возможность применения всех их в распределенных средах. Все они позволяют производить виртуализацию распространенных ОС и осуществлять запуск нескольких виртуальных машин на одном физическом узле.

Для создания рабочих виртуальных машин нами был выбран гипервизор Xen, который входит как компонент в состав ОС Scientific Linux 5 (https://www.scientificlinux.org), на ней реализован ресурсный GRID узел ИПХФ РАН [3]. В настоящее время пакет Xen можно установить из репозитория CERN на ОС Scientific Linux 4, являющуюся базовой для платформы gLite.

Для более эффективной оценки возможностей одна из рабочих станций ресурсного узла GRID ИПХФ была модернизирована с наращиванием RAM до 8 Гб, установкой дополнительных дисковых накопителей (по одному на виртуальную машину (ВМ)) и сетевых адаптеров. На нее была установлена ОС Scientific Linux 5.1 со встроенной поддержкой гипервизора Xen (версия 3.1) и пользовательского интерфейса к нему В дальнейшем созданные на ней виртуальные машины запускались на не модернизированных узлах ресурсного центра ИПХФ.

На данной машине были созданы образы следующих виртуальных машин: ОС Scientific Linux (версия 4.6), Ubuntu (8.10), AltLinux, а также,в качестве тестовой, Microsoft Windows XP. Тестирование машин показало их полную работоспособность. Производительность виртуальных машин была сопоставима с производительностью базовой ОС. Для дальнейших работ были оставлены виртуальные машины на базе Scientific Linux 4.6 и Linux Ubuntu 8.10. Данные виртуальные машины были сконфигурированы как расчетные узлы (worknodes) для сред gLite и Unicore соответственно. Виртуальная машина, сконфигурированная как 2-х процессорный рабочий узел с поддержкой PBS Torque и MPI для ресурсного узла gLite, была запущена на этой же физической машине и введена в список рабочих узлов на Computing

Elements ресурсного узла gLite, после чего с ее использованием был просчитан ряд тестовых и практических задач. С точки зрения внешнего пользователя, никаких отличий от физического узла в использовании виртуальной машины не наблюдалось, т.е. в составе ресурсного узла произошло добавление расчетного узла с примерно теми же вычислительными параметрами, что и на физических машинах.

Виртуальные машины на основе ОС Linux Ubuntu 8.10 были сконфигурированы для работы в качестве расчетных узлов созданного в ИПХФ ресурсного сайта СКИФ-Полигона на базе middleware Unicore [4]. Созданные виртуальные машины были перенесены на расчетные узлы кластера ИПХФ (ОС Scientific Linux 4.5, входят в состав ресурсного GRID узла gLite) и запущены там под управлением установленного гипервизора Хеп. Проведенное тестирование показало их полную работоспособность в данной обстановке. Был проведен эксперимент по запуску ВМ Ubuntu на основе вышеупомянутой физической машины с выполнением базового сервиса ресурсного узла Unicore - TSI (Target system interface).

В настоящее время в составе ресурсного сайта Unicore (полигон СКИФ-Полигон, https://unicorgw.iep.ae.ru:8080) работают две виртуальные машины в качестве расчетных узлов на базе физических машин, входящих, в свою очередь, в состав ресурсного узла gLite. А на базе физической машины, поддерживающей базовые компоненты ресурсного узла Unicore (Gateway; UAS, Unicore atomic services; XUUDB, Unicore user database) была протестирована работа изолированной ВМ с поддержкой базового сервиса TSI.

Таким образом, показана возможность установки, запуска и работы виртуальных машин с поддержкой различных сервисов на существующих физических узлах без значительного вмешательства в рабочее пространство распределенных и/или параллельных сред ресурсных узлов. Это дает возможность разворачивать распределенные вычислительные полигоны на уже существующих вычислительных кластерах без кардинальной перенастройки их аппаратно-программной конфигурации, что особенно важно для постоянно работающих центров класса <production farm>.

В настоящее время нами проводится изучение версии 2.0, выпущенной в октябре 2008 г., гипервизора VirtualBox, для которого заявлена существенно улучшенная, по сравнению с предыдущей версией, функциональность и взаимодействия с аппаратными компонентами.

Среди обнаруженных проблем виртуализации следует отметить две основные. Первая относится к учету ресурсов физического узла и мониторингу выполняемых ресурсным узлом задач. Учет ресурсов, востребованных ВМ, пока проводится на уровне гипервизора и не вполне корректно оценивается внешним мониторингом распределенной среды gLite, что может вести к недоучету ресурсов и завышенными ожиданиями со стороны входящих задач. Заметим, что в планах развития большинства гипервизоров заявлена возможность решения этих проблем. Вторая проблема связана со сложностью привязки изученных нами виртуальных машин к нескольким физическим сетевым интерфейсам, что может накладывать ограничения на работу ВМ внутри ресурсных узлов со сложной топологией интерконнекта.

1.2. Создание виртуального <контейнера> с образом среды исполнения параллельного приложения

Одним из вариантов виртуализации прикладного программного обеспечения для работы в распределенных средах становится виртуальное приложение, которое в виде виртуального <контейнера>доставляется на ресурсный узел вместе со всеми конфигурационными настройками, относящимися к операционной системе и приложению, и не требует процедуры предварительной установки. При этом отсутствуют конфликты приложения с другими, уже установленными на узле программами и даже с другими экземплярами этого же приложения. Суть виртуализации приложения заключается в создании персональной копии

необходимой части системных файлов и настроек операционной системы и доставки приложения совместно с этой информацией в качестве GRID задачи с последующим запуском в изолированном <контейнере>.

В рамках вычислительной химии данная концепция становится весьма востребованной, поскольку крупные пакеты (типа GAMESS, Gaussian, Dalton, CPMD) требуют весьма значительной настройки ОС всех узлов участвующих в расчете. Применение таких <контейнеров>позволяет передавать заранее настроенную среду как единое задание, не требующее дополнительного конфигурирования и сложной процедуры установки и настройки, производимых, как правило, вручную администратором кластера.

Авторами был разработан метод создания виртуальных «контейнеров:», запускаемых стандартными средствами middleware на удаленном ресурсном узле распределенной среды. В настоящее время этот метод применим для исполнения на узлах под ОС Linux.

Участниками проекта был проведен анализ процедуры исполнения типичного параллельного задания на ресурсном узле GRID, позволивший определить требования к создаваемому виртуальному образу среды исполнения, а также принципиальную возможность динамической организации среды исполнения для тестируемых типов ресурсов. Также был сделан анализ систем параллельного программирования для выбора оптимального виртуального образа среды исполнения параллельного приложения. В результате проведенного анализа в качестве базового пакета для разработки виртуального образа среды исполнения параллельного приложения был выбран пакет Mpich-2.

В настоящее время в качестве распределенных ресурсов GRID используются, как правило, узлы в виде кластеров рабочих станций с операционной системой и некоторым набором приложений, настроенных для работы в среде GRID и специфичных для каждого из этих распределенных ресурсов. Для проведения параллельных вычислений требуется наличие установленной на ресурсных узлах какой-либо системы параллельного программирования (например, MPI, OpenMP и др.), причем единые стандарты на установку данного вида программного обеспечения отсутствуют. Поэтому на ресурсных узлах GRID среды, как правило, можно ожидать наличия только библиотек стандарта MPI-1. Использование параллельных приложений в настоящее время в среде GRID крайне ограничено как возможностями брокера ресурсов, который не распознает тип параллельного задания, так и отсутствием предустановленных на кластерах необходимых Runtime библиотек, а также отсутствием стандартов на размещение таких библиотек. Поэтому запуск параллельного сложно сконфигурированного задания на узле распределенной среды обычно неэффективен, либо неудачен.

Проведенная работа была направлена на преодоление указанных недостатков и особенностей среды GRID для запуска в ней сложно сконфигурированных параллельных приложений.

Был проведен анализ предоставляемых псевдопользователям (так называемым <mapped users>) распределенных сред прав доступа к ОС расчетного узла, которые определяют, в свою очередь, возможность работы внешнего задания с локальной файловой системой, другими приложениями, исполняемыми модулями и утилитами. Для проведения такого анализа был самостоятельно разработан ряд оригинальных тестовых примеров, испытания которых позволили определить порядок запуска поступающих от брокера ресурсов на ресурсные узлы заданий, в том числе:

- присваиваемые заданиям имена пользователей и их права;
- создаваемые временные директории и требуемые файловые иерархии;
- доступность различных системных средств исполнения заданий (параллельные среды, доступ к очередям PBS и т.п.).

За **10** лет работы вычислительного центра ИПХФ РАН был накоплен большой опыт использования различных систем параллельного программирования - MPI, Linda, OpenMP и др. Наиболее широко использовались различные версии свободно распространяемого (Open Source) пакета Mpich, разработанного Argonne National Laboratories.

На основе версии Mpich-2 в вычислительном центре ИПХФ РАН в 2007-2008 гг. для кластера, работающего под ОС Scientific Linux, из исходных текстов была скомпилирована (после соответствующей доработки авторами) модифицированная библиотека MPI.

Основная причина использования менее распространенной версии стандарта MPI-2 заключается в том, что, как показали проведенные тесты, стандарт MPI-1 не позволяет создать полностью однородную среду исполнения задания на узлах кластера, несмотря на ряд дополнительно разработанных сторонних программных пакетов. Стандартом MPI-2 однородная среда исполнения задания на всех процессорах всегда рассматривается как базовая. Поэтому в качестве базового пакета для разработки виртуального образа среды исполнения параллельного приложения был выбран Mpich-2.

При создании виртуального образа среды исполнения параллельного приложения было принято решение ввести (на нынешней стадии разработки) ряд ограничений/допущений для ресурсных узлов:

- использована аппаратная архитектура x8б, как наиболее распространенная на ресурсных узлах российского сегмента GRID;
- на расчетных узлах GRID ресурса используется операционная система на базе клонов RedHat (Scientific Linux, Fedora, CentOS и т.п.);
- согласно стандарту настройки ресурсных узлов GRID *для* коммуникации между узлами используется интерфейс TCP/IP и беспарольный доступ по ssh;
- некоторые версии пакетов с целью повышения производительности вычислений имеют привязку к сетевым продуктам конкретных производителей и используют поставляемые этими производителями драйверы. На этом этапе работ такие версии, несмотря на их высокую эффективность, не были использованы.

Заметим, что принципиальных ограничений на работу в более <расширенных> условиях распределенных сред нет, включая переход на 64-битную среду.

В процессе тестирования было выбрано два потенциальных статических места инсталляция библиотек - по месту загрузки исполняемого приложения и использование общедоступной директории /tmp. Разработка системы динамического компилирования и инсталляции библиотек на данном этапе не рассматривалась.

В соответствии с указанными выше требованиями был сформирован перемещаемый программный пакет MPI-2, тестирование которого в локальных условиях на GRID кластере ИПХФ РАН с использованием тестовых примеров показало его работоспособность.

Полученный пакет в дальнейшем использовался в качестве базового прототипа для разработки виртуального образа среды исполнения конкретных параллельных приложений.

В качестве первичного тестового приложения был использована программа вычисления числа pi ('cpi.c') из пакета Mpich-2, правильность работы которой легко проверяется в параллельной среде с различным количеством узлов. Исходная тестовая программа была доработана с учетом особенностей запуска прикладных приложений, был получен ее исполняемый модуль и скрипты запуска с использованием библиотек Mpich-2.

Тестовый модуль и перемещаемый пакет Mpich-2 были собраны и упакованы в единый пакет, для запуска которого в среде GRID (gLite) были разработаны низкоуровневые скрипты. Отметим, что нет принципиальных препятствий для формирования такого же пакета

для работы в условиях других распределенных сред. Сейчас авторами тестируется среда Unicore.

С учетом возможного дальнейшего развития исследований была принята следующая схема запуска: на удаленный ресурсный узел сети GRID через брокер ресурсов передается главный скрипт и упакованный модуль, содержащий как исполняемые файлы, так и необходимые MPI библиотеки, т.е. динамически сформированный виртуальный <контейнер>. После доставки <контейнера>на ресурсный узел проводится следующая последовательность шагов:

1) сбор начальной информации о текущем ресурсном узле GRID (имена доступных хостов);

2) распаковка модуля в рабочей директории и перемещение библиотек в локальную директорию /tmp на текущем локальном узле (ТЛУ);

3) подготовка файла mpd.conf на ТЛУ для запуска MPI сервера mpd;

4) переопределение на ТЛУ текущего значения ряда переменных среды окружения для GRID пользователя;

5) запуск сервера mpd (с правами пользователя) на стартовом узле и проведение его runtime тестирования;

6) сбор информации о доступных узлах и их текущем состоянии. Список свободных узлов собирается в файл mpd.hosts, необходимый для запуска <кольца>серверов mpd;

7) распределение необходимых библиотек по списку свободных узлов (используется беспарольный доступ по протоколу ssh);

8) запуск <кольца>серверов mpd на ресурсном узле GRID и проведение его тестирования;

9) запуск параллельного приложения и его работа как обычного распределенного задания с последующей передачей результатов на брокер ресурсов и затем - пользователю;

10) удаление всех библиотек и созданных временных файлов со всех узлов.

Работа тестового варианта созданного <контейнера>была отлажена локально на ресурсном узле GRID ИПХФ РАН. Дальнейшее успешное тестирование было проведено на удаленных ресурсах полигона EGEE-RDIG (http://www.egee-rdig.ru) в рамках BO RGSTEST (узлы НИИЯФ МГУ и ИПХФ - используемый в роли удаленного узла) с запуском задания через брокер ресурсов **RDIG.**

1.3. Запуск реального параллельного приложения (GAMESS-US) в составе виртуального <контейнера>

Широко используемые в настоящее время в вычислительной химии пакеты прикладных программ (например, GAMESS, Gaussian, CPMD, NAMD и др.) требуют обязательной настройки большого количества переменных окружения операционной системы до запуска параллельного приложения на каждом из использующихся процессоров. Такая настройка обычно осуществляется в два этапа:

1) при установке прикладного ПО системным администратором на каждом расчетном узле ресурсного центра на уровне операционной системы;

2) при настройке соответствующих скриптов запуска задания для каждого пользователя согласно <Руководству пользователяжак приложения, так и системы параллельного программирования.

Такой традиционный подход со статическим линкованием необходимых библиотек к исполняемому модулю не способен создать полностью работоспособное перемещаемое сложно сконфигурированное параллельное задание на произвольном ресурсе среды GRID.

Выбранный выше подход по созданию виртуального образа среды исполнения на основе

перемещаемого пакета MPI-2 был применен для компиляции модифицированного пакета исходных кодов GAMESS и показал свою продуктивность.

GAMESS-US (http://www.msg.ameslab.gov/GAMESS) - одна из наиболее популярных программ для теоретического исследования свойств химических систем, уступает по известности лишь комплексу Gaussian, позволяет рассчитывать энергию, структуры молекул, частоты их колебаний, а также разнообразные свойства молекул в газовой фазе и в растворе, как в основном, так и в возбужденных состояниях. Основное направление - развитие методов расчета сверхбольших молекулярных систем.

Основные программные модули GAMESS поддерживают параллельный режим вычислений как на многопроцессорных компьютерах, так и на кластерах рабочих станций UNIX.

Работы по распараллеливанию GAMESS начались еще в 1991 году. Однако использование методов передачи сообщений MPI получило применение только с 1999 г., когда в пакете GAMESS была реализована модель интерфейса с распределенным размещением данных (DDI - Data Distributed Interface). Последняя версия интерфейса DDI, которая была оптимизирована для многопроцессорных SMP-архитектур общего вида, особенно работающих с памятью в стиле System V, была выпущена только в мае 2004 г.

В настоящее время практически все *ab initio* методы, включенные в пакет GAMESS, могут использовать параллельные вычисления.

Интерфейс DDI использует в качестве базовой сокетную TCP/IP модель межпроцессорных коммуникаций. Использование такого метода распараллеливания для работы на локальном кластере достаточно эффективно и довольно просто в конфигурации, но при работе в GRID средах возникает ряд принципиальных проблем: а) необходимо заранее явно указывать используемые расчетные узлы (что обычно нереально); б) неправильно оценивается загруженность расчетных узлов (учитывается только первый расчетный узел); в) отсутствует возможность контроля выполнения удаленной задачи средствами распределенного middleware.

Конфигурации же GAMESS с использованием библиотеки MPI авторами пакета разработаны только для ряда мейнфреймов известных производителей (Cray, IBM, SGI). В общем случае конфигурации с MPI не рекомендуются, и желающим предлагается экспериментировать с такими конфигурациями самостоятельно.

В ИПХФ РАН с целью расширения функциональности применения пакета GAMESS в сети GRID была поставлена задача разработки оригинальной конфигурации и сборки из исходных текстов исполняемого файла пакета GAMESS с использованием библиотеки MPI.

MPI (Message-Passing Interface, т.е. интерфейс для передачи сообщений) представляет стандартные спецификации для библиотек передачи сообщений. В ИПХФ РАН длительное время используется реализация Mpich, разработанная в Argonne National Laboratory (http://www.mcs.anl.gov/research/projects/mpich2).

Mpich представляет собой переносимую реализацию полных спецификаций MPI для широкого класса параллельных вычислительных средств, включающих кластеры рабочих станций и массивно-параллельных процессоров (MPPs). Mpich содержит, наряду с самими библиотеками MPI, программные средства для работы с программами MPI. Программные средства включают в себя переносимый стартовый механизм, несколько профилирующих библиотек для изучения производительности программ MPI.

Для работы с пакетом GAMESS в среде GRID на ресурсных узлах ИПХФ РАН первоначально была установлена последняя наиболее широко распространенная версия 1.2.7, которая является реализацией стандарта MPI-1. Достоинством данной версии является то, что она явно включает интерфейс Globus2, основанный на Globus Runtime System, что было бы эффективно *для* запуска Globus заданий. Однако получить работоспособную конфигурацию пакета GAMESS для MPI-1 не удалось по двум основным причинам:

- во-первых - особенности запуска исполняемого задания GAMESS, которая осуществляется скриптом, активизирующем более 150 переменных окружения. На главном узле среда создается правильно, но механизм передачи переменных окружения на подчиненные узлы в библиотеке Mpich стандартно отсутствует. В пакет Mpich был включен безопасный сервер (secure server), одной из задач которого являлась ликвидация этого недостатка. Но из-за неполной совместимости с операционной системой Scientific Linux эту функцию безопасного сервера использовать не удалось. При запуске задания на локальном узле пакет Mpich использует команды оболочки типа `.`, eval, exec и др., которые не наследуют среду окружения запускающего процесса.

- во-вторых - особенности реализации команды запуска параллельных заданий mpirun пакета Mpich-1. Запуск заданий на главном и подчиненном узлах существенно различаются - строки команды удаленного запуска (rsh или ssh) на подчиненных узлах дополняются служебными переменными. В результате стандартное расположение строчных аргументов задания GAMESS нарушается и не распознается, что ведет к краху запуска.

С 2002 года стандарт MPI-1 больше не поддерживается Argonne National Laboratory, и выпущена новая версия пакета Mpich, соответствующая стандарту MPI-2, в котором коренным образом изменена система запуска параллельных заданий, и устранены указанные недостатки. Перед запуском задания осуществляется запуск кольца серверов mpd (<<mpd ring>>), одна из задач которых состоит в выравнивании среды окружения на главном и подчиненных узлах. В более общем случае, кольцо серверов может запускаться пользователем root, а остальные пользователи могут использовать это глобальное кольцо. Обычно на узлах GRID пользователю root запрещен любой удаленный доступ между узлами, и, следовательно, каждый псевдопользователь GRID должен запускать собственное локальное кольцо серверов (основанное на использовании непривилегированных портов).

После установки библиотек MPI стандарта 2.0 (версия 1.0.3 пакета Mpich2) была проведена соответствующая модификация конфигурационных скриптов пакета GAMESS (compddi, comp, compall, Iked), а также программных модулей ddi_Jnit.c и ddi_base.h. Был полностью переписан соответствующий раздел в запускающем скрипте rungms, который сначала запускает кольцо серверов mpd, а затем уже и само задание. После сборки исполняемого файла было проведено его тестирование на включенных в пакет GAMESS примерах файлов данных, и получено совпадение результатов. Запуск параллельных заданий осуществляется командой mpiexec, которая не имеет указанных выше недостатков команды mpirun.

После компиляции GAMESS был сформирован виртуальный <<контейнер>>(аналогичный описанному выше) из модифицированного GAMESS, бинарных библиотек, исполняемых файлов Mpich-2, скриптов развертывания и запуска на создаваемой параллельной среде. Серия первичных запусков (с использованием собственных тестовых примеров пакета GAMESS) вплоть до получения положительного результата была проведена на ресурсном узле GRID ИПХФ РАН, использованном как удаленный через брокер ресурсов RDIG . Дальнейшее успешное тестирование было проведено на удаленном ресурсном узле НИИЯФ МГУ. Были проведены успешные запуски пакета GAMESS с применением данной технологии. Рассчитаны тестовые примеры различного уровня сложности из дистрибутива GAMESS, подтвердившие полную работоспособность разработанной технологии.

В перспективе более общим вариантом данной технологии является использование (по аналогии с описанным <<контейнером>>) виртуальных машин как исходящих распределенных заданий, что позволит гарантировать пользователю необходимое качество обслуживания, не затрагивающее при этом работу основных служб ресурсных узлов. Таким образом, пользо-

вателю распределенной среды может быть предоставлена полностью изолированная виртуальная вычислительная среда, по своим свойствам не уступающая физическому серверу, в которой может быть предоставлен любой его собственный вычислительный сервис. Приложения, реализованные в ВМ, в этом случае абсолютно не зависят от операционной системы и окружения, в котором ВМ выполняется. Еще одним преимуществом такой виртуализации является полное отделение как конкретной службы или программы от внешней среды, так и внешней среды от нее. Это возможно благодаря использованию дополнительного слоя программного обеспечения - виртуального оборудования, позволяющего выполнять обычное приложение, как если бы оно было запущено на отдельном компьютере. В чем смысл данной технологии в условиях GRID среды? Пользователь получает возможность создать образ виртуальной машины с предустановленной операционной системой и полностью сконфигурированными приложениями, нацеленной на решение конкретной задачи. Этот образ затем передается на распределенный ресурс и исполняется там как GRID приложение, не требуя настройки данного узла под конкретные задачи. Это существенно облегчает адаптацию прикладного ПО для работы в распределенных средах. Дополнительным плюсом служит то, что данные технологии в принципе позволяют запускать образы виртуальных машин с операционными системами, отличными от установленных на ресурсах (например, Windows ВМ на Linux кластере). Следует отметить потенциальные недостатки данного метода/Прежде всего - это размер передаваемых заданий, задание может достигать нескольких гигабайтов. <<Накладные расходы>> на виртуализацию могут забирать до 15-20% от мощности ресурса (при неудачном конфигурировании).

2. Заключение

Таким образом, на реальных примерах показана широкая степень применимости различных методов виртуализации для работы с приложениями вычислительной химии в условиях распределенных и параллельных сред. Авторами были сформированы несколько виртуальных вычислительных ресурсов и создана методика формирования виртуальных программных <контейнеров>для запуска сложно сконфигурированных и требующих особых настроек параллельной среды выполнения прикладных пакетов вычислительной химии.

Применение указанных методов виртуализации позволит существенно облегчить создание проблемно-ориентированных виртуальных вычислительных организаций, поскольку позволит объединять в их составе весьма разнородные вычислительные ресурсы, избегая при этом значительной реконфигурации этих ресурсов под нужды распределенной среды.

Работа проводилась при финансовой поддержке программы Союзного Государства <<СКИФ-ГРИД>>и Программы фундаментальных исследований Президиума РАН №15 на 2007-2008 годы ^Разработка фундаментальных основ создания научной распределенной информационно-вычислительной среды на основе технологий GRIIh>. Статья рекомендована к печати программным комитетом международной научной конференции ^.Параллельные вычислительные технологии 2009>, http://agora.guru.ru/pavt.

Литература

1. Вычислительная химия в среде GRID: параллельные и распределенные вычисления / СМ. Алдошин, В.М. Волохов, Д.А. Варламов, А.В. Пивушков // Труды Второй международной конференции «Суперкомпьютерные системы и их применение> SSA'2008, Минск, октябрь 2008. - Минск, 2008. - С. 114 - 118.

2. Волохов, В.М. Крупномасштабные задачи химии на параллельных и распределенных

вычислительных полигонах: современное состояние и перспективы / В.М. Волохов, Д.А. Варламов, А.В. Пивушков, // Научный сервис в сети Интернет: решение больших задач: тр. Всерос. науч. конф. (22 - 27 сент. 2008 г., г. Новороссийск). - М., 2008. - С. 210 - 212.

3. Варламов, Д.А. Распределенные и параллельные вычисления в области химии на ресурсном узле ГРИД ИПХФ РАН / Д.А. Варламов, В.М. Волохов, А.В. Пивушков, Н.Ф. Сурков, Г.А. Покатович // Distributed Computing and Grid-Technologies in Science and Education: Extended Proceedings of the III Intern.Conf.: сб. науч. тр. - Дубна, 2008. - С. 127 - 130.

4. Создание ресурсного узла вычислительного полигона СКИФ-ГРИД на базе middleware UNICORE / Д.Е. Баргатин, А.В. Пивушков, Д.А. Варламов, В.М. Волохов, // Современные информационные технологии в науке, образовании и практике: материалы VII Всерос. науч.-практ. конф. (27 - 28 нояб. 2008 г., г. Оренбург). - Оренбург, 2008. - С. 102 - 104.

Отдел вычислительных и информационных ресурсов
Институт проблем химической физик РАН
dima@iem.ac.ru

Поступила в редакцию 2 марта 2009 г.