

Увеличение вычислительной мощности распределенных систем с помощью грид-систем из персональных компьютеров*

R. Lovas¹, А.П. Афанасьев², В.В. Волошинов²,
М.А. Посыпкин², О.В. Сухорослов², Н.П. Храпов²

¹МТА SZTAKI,
²ИСА РАН

Рассматриваются технологии грид-систем из персональных компьютеров, получившие широкое распространение в последнее время. Эти технологии позволяют задействовать свободные от полезной работы ресурсы рабочих станций, настольных ПК, ноутбуков для проведения ресурсоемких расчетов. Приводится обзор программного обеспечения, используемого для создания гридов из персональных компьютеров. Рассматриваются средства разработки приложений для таких систем. Также в статье уделяется внимание вопросам интеграции различных типов грид-систем.

1. Введение

Современная наука и производство немислимы без масштабных расчетов, требующих колоссальной вычислительной мощности, которую можно обеспечить только в рамках распределенных и параллельных систем. В настоящее время существует несколько подходов к организации распределенных вычислений. Наибольшую производительность обеспечивают специализированные суперкомпьютеры большой мощности, которых насчитывается несколько сотен по всему миру [1]. Они могут решать широкий спектр вычислительных задач, но требуют больших расходов на создание и эксплуатацию.

Несколько меньшей производительностью обладают так называемые сервисные гриды, которые соединяют кластеры, установленные в различных организациях. Это решение дешевле по отношению к суперкомпьютерам, но также требует выделенных ресурсов и значительных усилий, связанных с эксплуатацией. Инфраструктура сервисных гридов состоит из набора сервисов, обеспечивающих доступ к брокерам ресурсов, информационным службам, хранилищам данных, вычислительным ресурсам. Пользователи сервисных гридов имеют соответствующие права доступа к предоставляемым сервисам. Контроль доступа к ресурсам осуществляется посредством сертификатов безопасности. Хорошо известны следующие технологии создания сервисных гридов, как Globus [2], LCG-2/gLite (EGEE) [3], ARC [4] и Unicore [5].

На нижнем уровне рассматриваемой иерархии находятся грид-системы персональных компьютеров (ГПК). ГПК основаны на наблюдении, что большую часть времени персональные компьютеры либо простаивают, либо загружены лишь на некоторую небольшую долю своей мощности. ГПК обеспечивают возможность объединения свободных вычислительных мощностей домашних компьютеров и персональных компьютеров учреждений в единую распределенную систему для решения сложных вычислительных задач. В отличие от сервисных гридов, грид-системы из персональных компьютеров легко установить и поддерживать. Как правило, требуется одна рабочая станция, на которой запускается серверная часть инфраструктуры. Пользователи со всего мира имеют возможность подключать свои персональные компьютеры к этому ресурсу, предоставляя тем самым свободные ресурсы своих компьютеров для работы приложений, размещенных на центральном сервере. Грид-системы из персональных компьютеров являются наиболее дешевым решением, обеспечивающим большую вычислительную мощность. ГПК обладают огромным потенциалом роста – в настоящее время в мире насчитывается более одного миллиарда персональных компьютеров и их число стремительно увеличивается. К сожалению, далеко не все распределенные приложения могут эффективно выполняться на подобных системах из-за серьезных ограничений, накладываемых возможностями по передаче

* Работа выполнена при поддержке Седьмой Рамочной программы Европейского Союза (FP7/2007-2013), грант № 261561 (DEGISCO).

данных и высокой вероятностью отказа узлов, участвующих в вычислениях. Вместе с тем, достаточно широкий класс практических задач укладывается в модель управляющий-рабочие, которая является основной моделью приложения в ГПК. К этому классу относятся многие переборные и комбинаторные задачи, моделирование методом Монте-Карло, задачи идентификации и многие другие. Для таких задач использование ГПК оправдано и позволяет разгрузить суперкомпьютеры и сервисные гриды. Резюмируя можно сказать, что грид-системы персональных компьютеров являются дешевой альтернативой суперкомпьютерам и сервисным гридам и для ряда задач могут их успешно заменять.

В настоящее время широко известны следующие системы ГПК: Condor[6], BOINC [7], XtremWeb [8], OurGrid[9]. Эти системы могут быть использованы для создания распределенных систем масштаба лаборатории, предприятия, города или всего мира. Несмотря на различия между сервисными гридами и грид-системами персональных компьютеров между ними много общего. В частности, модель управляющий-рабочие используется при разработке приложений для систем обоих типов. Поэтому, важнейшей задачей является обеспечение интероперабельности между сервисными гридами и ГПК. Различают два вида интероперабельности: на программном и на системном уровне. Интероперабельность на программном уровне означает наличие универсального интерфейса для подключения вычислительного ресурса, позволяющего объединять разнородные ресурсы в единую распределенную систему. На этом принципе основаны системы SAGA[10], P-GRADE[11], MathCloud[12], BNB-Grid[13]. Интероперабельность на системном уровне предполагает наличие прозрачных для приложения механизмов, обеспечивающих «бесшовную» интеграцию вычислительных ресурсов. Примером такого подхода является механизм мостов[14], применяемый для интеграции сервисных гридов и ГПК.

В данной работе рассматриваются основные технологии, применяемые для организации грид-систем персональных компьютеров (раздел 2), методы и технологии разработки приложений для таких систем (раздел 3), средства обеспечения интероперабельности с сервисными гридами (раздел 4). В заключении (раздел 5) приводятся примеры использования данной технологии для решения научных задач и рассматриваются организационные аспекты создания и развития ГПК.

2. Технологии гридов из персональных компьютеров

2.1 Condor

Система Condor [6] разрабатывается с 1988 года в рамках одноименного исследовательского проекта в University of Wisconsin-Madison (США). По предоставляемой пользователю функциональности Condor близок к традиционным системам управления пакетной обработкой, используемых на вычислительных кластерах. В то же время, оригинальная архитектура Condor поддерживает режимы вычислений, недоступные в традиционных системах. Режим вычислений высокой пропускной способности (High-Throughput Computing) позволяет надежно поддерживать высокие объемы вычислительных ресурсов в течение длительных периодов времени путем эффективного использования всех доступных в сети вычислительных ресурсов. Режим оппортунистических вычислений (opportunistic computing) позволяет использовать ресурсы в любой момент времени, когда они доступны, не требуя постоянного монопольного доступа.

Помимо управления кластерами из выделенных узлов Condor может эффективно использовать ресурсы простаивающих персональных компьютеров, объединяя все доступные ресурсы организации в единый «виртуальный» кластер или пул. Например, машина может быть автоматически добавлена в пул в те моменты, когда отсутствует активность клавиатуры или мыши. В случае, если машина вновь задействована ее владельцем, Condor выводит машину из пула, осуществляя миграцию выполнявшихся на ней заданий на другие доступные узлы. В отличие от традиционных систем управления кластером, Condor не требует наличия разделяемой между узлами файловой системы. Система поддерживает передачу файлов задания между узлами и прозрачное перенаправление потоков ввода/вывода задания на машину, запустившую данное задание.

Пул под управлением Condor состоит из выделенного узла, играющего роль центрального менеджера, и произвольного количества других машин. Любая машина в пуле может играть как

роль узла, запускающего задания, так и роль узла, выполняющего задания. На концептуальном уровне пул представляет собой совокупность ресурсов (машин) и запросов (заданий). Центральный менеджер пула играет роль посредника, сопоставляя поступившие запросы с доступными ресурсами. Каждая машина в пуле периодически отправляет свое состояние центральному менеджеру, который использует данную информацию при выборе узлов для выполнения заданий.

Для описания заданий и ресурсов в Condor используется единый механизм ClassAd, обладающий высокой гибкостью и выразительностью. В описании задания могут быть указаны как обязательные требования, так и дополнительные предпочтения по выбору машин для выполнения задания. Аналогично, в описании ресурса (машины) могут быть указаны требования и предпочтения относительно заданий, которые могут быть запущены на данной машине. Важно отметить, что данные условия могут определяться индивидуально владельцем каждой машины. Гибкость и выразительность языка ClassAd позволяет описать практически любые политики использования ресурсов. Использование единого механизма описания для заданий и ресурсов обеспечивает эффективный выбор ресурсов для выполнения заданий с учетом взаимных требований.

Несмотря на то, что основной областью применения Condor является интеграция ресурсов внутри организации, благодаря реализованному механизму «flocking» система также поддерживает объединение ресурсов нескольких независимых пулов, принадлежащих различным организациям.

2.2 VOINC

VOINC (Berkeley Open Infrastructure for Network Computing) [7] представляет собой платформу с открытым кодом для организации проектов добровольных вычислений. Разработка системы ведется в U.C. Berkeley Spaces Sciences Laboratory (США) исследовательской группой, которая также разрабатывала проект SETI@home. Работа над VOINC была начата в 2002 году с целью создания универсальной программной платформы для проектов подобного рода, которая бы упростила процесс развертывания необходимой инфраструктуры и разработки приложений. Первый проект добровольных вычислений на основе VOINC был запущен в 2004 году. В настоящее время насчитывается около 60 публичных проектов на основе VOINC, делая платформу стандартом де-факто в данной области. Система также часто используется для организации внутренних гридов из персональных компьютеров.

Программное обеспечение VOINC состоит из двух основных частей: серверной, которая обеспечивает функционирование проекта, и клиентской, размещаемой на машинах добровольцев. Каждый проект на основе VOINC создается и функционирует независимо от других проектов, поддерживая собственный центральный сервер и веб-сайт. Для участия в проекте добровольцы устанавливают на своих компьютерах универсальный клиент VOINC, распространяемый с сайта платформы и доступный для всех основных операционных систем.

Проект на основе VOINC идентифицируется с помощью уникального адреса (URL), являющегося одновременно главной страницей веб-сайта проекта и точкой входа для клиентов. В рамках проекта могут выполняться несколько приложений, состав которых может со временем изменяться. В рамках VOINC предусмотрена поддержка широкого класса вычислительных приложений, которые могут быть сформулированы в виде совокупности из большого количества независимых заданий. Платформа обеспечивает надежное и эффективное выполнение приложений в динамичной распределенной среде, реализуя механизмы планирования заданий, передачи данных и обработки отказов. Существующие приложения на таких языках, как C, C++ и FORTRAN, могут быть использованы в рамках VOINC без или с минимальной модификацией их кода.

После установки клиента VOINC, добровольцы могут подключить клиент к одному или нескольким проектам. При этом пользователь может указать то, каким образом в процентном отношении ресурсы его компьютера будут распределены между данными проектами. При подключении к проекту пользователь фактически дает разрешение на выполнение на своей машине любых исполняемых файлов, загруженных с сервера проекта. Несмотря на то, что VOINC обеспечивает определенную изоляцию выполняемого на клиентской стороне кода (sandboxing),

в общем случае пользователь самостоятельно должен убедиться в подлинности, защищенности и научной значимости проекта.

Для учета индивидуального вклада каждого из добровольцев в проект в BOINC реализован механизм учета кредитов, которые вычисляются пропорционально процессорному времени, использованному для выполнения заданий проекта. В BOINC также предусмотрена возможность экспорта информации о кредитах пользователя на уровне отдельного проекта. Это, а также поддержка глобальной идентификации пользователя по его адресу электронной почты, позволяет агрегировать и делать доступной сводную статистику кредитов пользователя по всем проектам.

Механизм менеджеров учетных записей (account manager) позволяет централизованно управлять подключением клиентов к тем или иным проектам. В данном случае, вместо прямого подключения к проекту, клиент подключается к веб-сервису менеджера учетных записей. Во время работы клиент периодически связывается с менеджером, получая от него список проектов, к которым необходимо подключиться. Данный механизм может быть использован для делегации пользователем решений по выбору поддерживаемых проектов некоторому доверенному комитету.

2.3 XtremWeb и XWHEP

XtremWeb [8] представляет собой ПО с открытым кодом для создания грид-систем из персональных компьютеров, разработанное в рамках исследовательского проекта по глобальным вычислениям в INRIA/IN2P3 (Франция). В настоящее время дальнейшее развитие системы под именем XWHEP (XtremWeb-HEP) ведется в LAL (Laboratoire de l'Accélérateur Linéaire), IN2P3-CNRS (Франция). Система реализована на языке Java и доступна на условиях лицензии GPLv2.

Архитектура XWHEP состоит из трех основных типов компонентов: серверы, рабочие и клиенты. Серверы обеспечивают функционирование центральных сервисов системы, таких как планировщица и агрегатор результатов заданий. Рабочие (workers) устанавливаются владельцами ресурсов на своих компьютерах для предоставления данных ресурсов в рамках системы. Клиенты устанавливаются пользователями ресурсов на своих компьютерах для взаимодействия с системой и использования агрегированных ею ресурсов. Клиент позволяет пользователям управлять системой, размещать приложения, запускать их в виде заданий и загружать полученные результаты. Запущенные клиентом задания регистрируются на сервере и назначаются рабочим. Аналогично системе Condor, в рамках XWHEP любой рабочий может одновременно играть роль клиента и наоборот.

XWHEP значительно расширяет изначальные возможности XtremWeb в плане обеспечения безопасности. Так, в системе реализованы механизмы прав доступа как на уровне пользователей, так и групп пользователей. Данные механизмы однородным образом могут быть применены для ограничения доступа пользователей к ресурсам, приложениям и данным.

2.4 OurGrid

Система OurGrid [9], разрабатываемая в Federal University of Campina Grande (Бразилия), реализует оригинальную концепцию кооперативных грид-систем на основе модели peer-to-peer. В рамках данной концепции простаивающие вычислительные ресурсы участников системы объединяются и разделяются между ними таким образом, что размер доступных участнику ресурсов пропорционален количеству ресурсов, предоставленных им сообществу. Система реализована на языке Java и доступна в исходных кодах на условиях лицензии GPL.

OurGrid реализует масштабируемую и защищенную распределенную платформу для выполнения приложений класса Bag-of-Tasks (BoT), к которому относится широкий спектр прикладных задач. Данные параллельные приложения, состоящие из множества независимых заданий, идеально подходят для запуска в грид-системах из персональных компьютеров.

Архитектура OurGrid основана на симметричной peer-to-peer модели, где каждый грид-сайт представлен в системе равноправным узлом (peer). Ключевой проблемой, связанной с применением подобной модели, является возможность появления в системе узлов-паразитов (freeriders),

которые потребляют ресурсы других узлов, никогда не давая взамен своих ресурсов сообществу. Для решения данной проблемы в OurGrid используется оригинальный подход «Сеть услуг» (Network-of-Favors), мотивирующий кооперацию между узлами. Под услугой подразумевается выделение ресурса узлу, который запросил данный ресурс. Ценность (количественная мера) услуги определяется ценностью вычислений, выполненных по запросу узла. Каждый узел локально хранит суммарное количество предоставленных и полученных им услуг. При выделении своих ресурсов узлы предпочитают те узлы, которым они «должны» больше всего услуг.

С 2004 года разработчики OurGrid поддерживают одноименную открытую кооперативную грид-систему, участники обмениваются свободными вычислительными ресурсами. К данной системе может присоединиться любой желающий, загрузив и установив на своих ресурсах ПО OurGrid. При этом, в отличие от традиционных грид-систем, не требуется никаких контактов и согласований.

3. Разработка приложений

Несмотря на широту класса приложений, пригодных для запуска в гридах из персональных компьютеров, данный класс предъявляет определенные требования к приложениям. Перечислим основные из них:

- Наличие большого количества подзадач (заданий), укладываемых в модель «управляющий-рабочие»;
- Типичный размер заданий соответствует нескольким часам вычислений на типичном персональном компьютере и не более десятков мегабайт входных и выходных данных;
- Отсутствие необходимости в общем доступе к данным и взаимодействии между рабочими узлами, выполняющими задания. Синхронизация между рабочими узлами возможна только через центральный сервер.

Приложения, запускаемые в гридах из персональных компьютеров, как правило, состоят из двух распределенных частей: серверной (управляющей) и клиентской (рабочей). Серверная часть приложения отвечает за формирование заданий, проверку результатов заданий и последующую их обработку. Клиентская часть приложения выполняется на рабочих узлах и отвечает за выполнение заданий. Данная часть приложения должна поддерживать работу в фоновом режиме, без интерактивного ввода или графического интерфейса. Гетерогенность рабочих узлов обуславливает необходимость компиляции клиентской части приложения для всех используемых вычислительных платформ.

Технологии гридов из персональных компьютеров предоставляют функционирующее между двумя указанными частями приложения промежуточное ПО, которое прозрачным образом осуществляет распределение клиентского кода и заданий между доступными машинами, мониторинг выполнения заданий, обработку отказов, сбор результатов заданий и передачу их серверной части приложения. Для разработки, развертывания и управления приложениям в каждой из технологий, как правило, предусмотрены универсальные интерфейсы командной строки и/или прикладного программирования.

Например, BOINC предоставляет подобного рода интерфейсы как для серверной (генерация заданий, валидация результатов и их обработка), так и для клиентской части (управление выполнением приложения, сохранение промежуточного состояния и учет используемых ресурсов) приложения. Данные интерфейсы подразумевают модификацию исходного кода приложения. Поскольку существует много «унаследованных» приложений, код которых недоступен или требует значительных усилий по модификации, в BOINC предусмотрен готовый конфигурируемый адаптер (`wrapper`), позволяющий запускать на клиентской стороне любое приложение без модификации его исходного кода.

Технологии XtremWeb/XWNER и OurGrid поддерживают запуск произвольных исполняемых файлов, что упрощает перенос приложений.

В случае с XtremWeb/XWNER разработчик приложения в первую очередь должен зарегистрировать приложение, загрузив исполняемый файл на центральный сервер с помощью клиента. Поддерживается регистрация сразу нескольких версий исполняемого файла для разных платформ. Любой пользователь может зарегистрировать приложение, при этом права доступа

приложения будут определяться правами доступа пользователя. Для регистрации публично доступного приложения требуются права администратора. После того, как приложение зарегистрировано, пользователь может запускать задания, указывая идентификатор приложения, аргументы командной строки и ссылки на входные данные. Задания могут быть объединены в группы, что упрощает реализацию приложений с большим количеством заданий.

В случае с OurGrid приложение описывается с помощью дескриптора задачи (job description file). Задача (job) состоит из нескольких заданий (tasks). Каждое задание формируется из трех подзаданий: начального (init), удаленного (remote) и финального (final), которые выполняются последовательно. В качестве подзаданий указываются внешние команды, вызываемые OurGrid. Таким образом, допускается использование произвольных исполняемых файлов и скриптов. Начальное и финальное подзадания выполняются локально, на машине с которой производится запуск задачи. Начальное подзадание предназначено для подготовки окружения для выполнения задания, например — передачи входных данных задания на машину, которая выбрана для его выполнения. Финальное подзадание обычно используется для загрузки результатов задания на машину пользователя. Удаленное подзадание запускается на машине в гриде и выполняет основные вычисления, связанные с заданием. Помимо описания подзаданий, в описании задания также входят требования к ресурсам. OurGrid позволяет описывать подзадания, абстрагируясь от специфики конкретных машин, на которых будут запущены данные подзадания, например - организации файловой системы. Для этого применяются абстракции, представляющие хранилище данных на удаленной машине в гриде, и набор команд для отправки и получения файлов из хранилища.

Система Condor поддерживает запуск как немодифицированных исполняемых файлов, так и приложений, собранных с использованием библиотеки Condor. Во втором случае система берет на себя обязанности по автоматическому сохранению контрольных точек приложения (checkpointing) и перенаправлению системных вызовов с исполняемой на запускаемую машину. Задание описывается с помощью текстового дескриптора (submit description file), содержащего информацию об исполняемом файле, аргументах запуска, входных и выходных файлах, требованиях к ресурсам и т. п. Дескриптор заданий поддерживает возможность описания задания-кластера, состоящего из набора параметризованных подзаданий.

4. Технологии интеграции сервисных гридов и гридов из персональных компьютеров

Механизм мостов, предложенный в работе [14], позволяет осуществлять интеграцию сервисных гридов и ГПК на системном уровне, т.е. прозрачным для пользователя образом. На данный момент этот подход реализован для связи грид-инфраструктуры EGEE/EGI с несколькими ГПК (Рис. 1). Суть подхода состоит в специальном программном компоненте 3G-Bridge (a Generic Grid to Grid bridge) который, опираясь на абстрактное понятие задания, может быть использован для интеграции двух грид-систем. По выполняемым функциям интегрирующее программное обеспечение можно подразделить на два типа:

- Мост EGEE \Rightarrow DG, обеспечивающий запуск заданий сервисного грида в инфраструктуре ГПК.
- Мост DG \Rightarrow EGEE, позволяющий, наоборот, запускать задания ГПК в инфраструктуре EGEE.

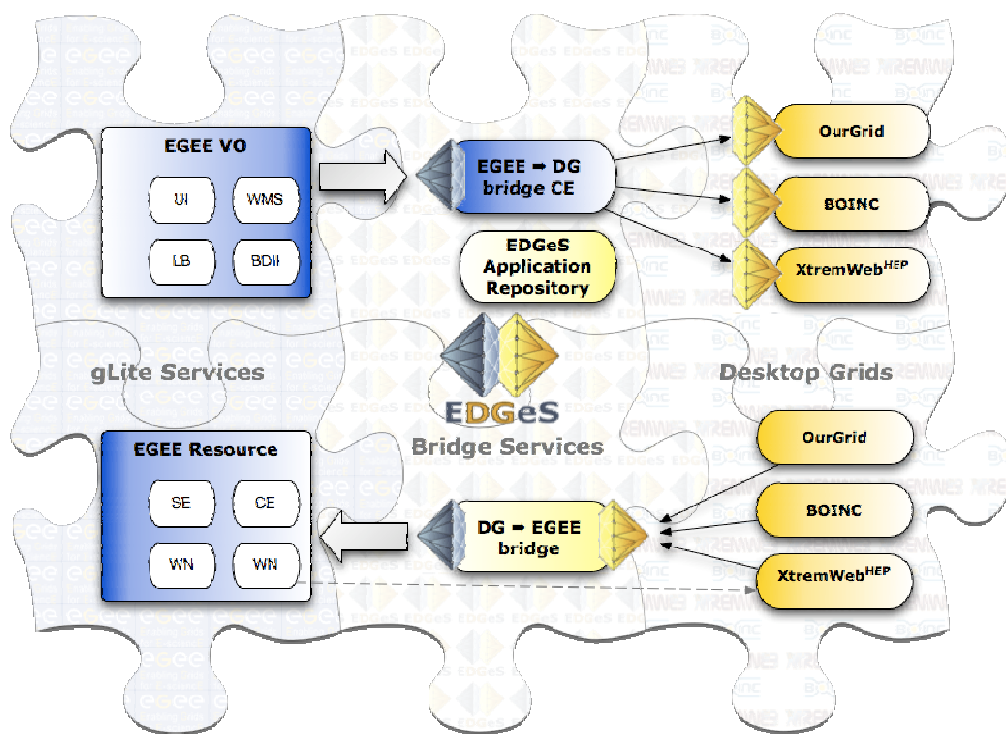


Рис. 1. Основные элементы инфраструктуры, обеспечивающие взаимодействие разнородных грид-систем.

5.1 Мост EGEE ⇒ DG

Данное соединение функционирует как Computing Element (CE) сервисного грида, где задания вместо вычислительных узлов направляются в инфраструктуру грида из персональных компьютеров (BOINC, XWHEP, OurGrid). Взаимодействие различных типов грид-систем обеспечивается тремя основными программными компонентами:

1. Функционирующий на стороне gLite модифицированный Computing Element, который отправляет принятые из инфраструктуры сервисного грида задания на удалённый мост. Данный CE поставляется в качестве модуля YAIM, и может быть установлен и настроен вместе с другими компонентами gLite.
2. На стороне сервера ГПК функционирует специальный адаптер, отвечающий за получение заданий, их преобразование для новой инфраструктуры, и выполнение.
3. Репозиторий приложений (Application Repository — AP), содержащий информацию о всех приложениях, проходящих через данный мост.

5.2 Мост DG ⇒ EGEE

Поскольку принцип работы и основное программное обеспечение зависит от типа подключаемой инфраструктуры ГПК, для каждой из них создана отдельная реализация моста:

1. **Мост BOINC ⇒ EGEE**, который функционирует как клиент BOINC, отправляя скачанные задания в виртуальную организацию EGEE. В инфраструктуре сервисного грида задание запускается специальной программой (jobwrapper), которая запускает приложение BOINC, и эмулирует для него окружение клиента BOINC.
2. **Мост XWHEP ⇒ EGEE**, подключающий рабочие узлы EGEE к гриду XWHEP путём запуска рабочих компонентов инфраструктуры в виде заданий EGEE.
3. **Мост OurGrid ⇒ EGEE**, который непосредственно запускает задания на вычислительных узлах виртуальной организации EGEE.

Для организации взаимодействия с инфраструктурой EGEE в двух последних реализациях используется библиотека jLite[15].

В рамках EGEE создана виртуальная организация desktopgrid.vo.edges-grid.eu, предназначенная для выполнения заданий ГПК. Гриды из персональных компьютеров (включая три основных варианта: BOINC, XWHEP, OurGrid), подключённые к EDGeS, и поддерживающие запуск EGEE приложений, позволяют добровольцам внести свой собственный вклад в проект EGEE и инфраструктуру сервисного грида EGEE. Инфраструктура EDGeS, содержащая более 100 тыс. процессорных ядер создавалась на основе нескольких типов ГПК и европейского грида EGEE/EGI. Использование гибридной вычислительной инфраструктуры в научных целях даёт следующие преимущества:

1. Операторы гридов из персональных компьютеров и сервисных гридов могут объединить свои системы посредством инфраструктуры EDGeS. Таким образом можно увеличить вычислительный ресурс, доступный для пользователей.
2. Менеджеры виртуальных организаций (ВО) EGEE или другого сервисного грида могут подключиться к виртуальной организации EDGeS, и таким образом добавить вычислительный ресурс в инфраструктуру, и запускать в ней свои приложения.
3. Потребители ресурсов, подключённые к EDGeS имеют в распоряжении больший вычислительный ресурс, чем на локальных гридах. Если используется приложение, портированное на несколько типов грид-систем, то соответствующие задания будут автоматически запускаться в нужных частях инфраструктуры.
4. Любому желающему предоставить свой вычислительный ресурс для нужд науки достаточно подключиться к одной из поддерживаемых EDGeS грид-систем, и таким образом стать участником этой вычислительной инфраструктуры.

5. Заключение

В настоящее время грид-системы персональных компьютеров стали серьезной альтернативой традиционным высокопроизводительным вычислительным системам и получили признание международного научного сообщества. Созданы инфраструктуры ГПК, в рамках которых ведутся расчеты различных научных приложений. Примерами таких инфраструктур являются локальная инфраструктура университета Вестминстера[16], открытый проект EDGeS@home[17], российский проект центра Грид-технологий и распределенных вычислений[18].

Грид-системам персональных компьютеров посвящено несколько проектов европейской рамочной программы (FP7). Проект DEGISCO[19], направлен на поддержку и развитие международной кооперации в области создания и расширения ГПК, всемерную популяризацию этой технологии, поддержку разработчиков распределенных приложений. Участниками проекта DEGISCO и родственного проекта EDGI [20] основана Международная федерация грид-систем из персональных компьютеров (IDGF)[21]. Международная федерация грид-систем из персональных компьютеров является организацией, объединяющей людей из различных компаний, университетов и исследовательских институтов, заинтересованных в использовании вычислительной мощности такого типа и желающих обменяться опытом друг с другом. Своим участникам федерация предоставляет несколько услуг: личные встречи на семинарах, доступ к документации, форум, веб-портал и консультации экспертов.

Учитывая темпы роста числа персональных компьютеров в мире и развития Интернет можно с уверенностью сказать, что грид-системы персональных компьютеров будут активно развиваться и в ближайшем будущем интерес к этой технологии будет возрастать. Технологии интеграции различных типов грид-систем будут способствовать популяризации ГПК в научной среде.

Литература

1. Top 500 Supercomputers. URL: <http://www.top500.org> (дата обращения: 13.02.2011).

2. I. Foster, C. Kesselman: The Globus Project: A Status Report, Proc. IPPS/SPDP '98 Heterogeneous Computing Workshop, pp. 4-18, 1998.
3. The gLite webpage, <http://glite.web.cern.ch/glite>.
4. M.Ellert et al.: Advanced Resource Connector middleware for lightweight computational Grids, Future Generation Computer Systems 23 219-240, 2007.
5. A. Streit, D. Erwin, Th. Lippert, D. Mallmann, R. Menday, M. Rambadt, M. Riedel, M. Romberg, B. Schuller, and Ph. Wieder L. Grandinetti (Edt.): UNICORE - From Project Results to Production Grids, Grid Computing: The New Frontiers of High Performance Processing, Advances in Parallel Computing 14, Elsevier, 2005, pages 357-376
6. Michael Litzkow, Miron Livny, and Matt Mutka, Condor-A Hunter of Idle Workstations. In Proc. The 8th International Conference of Distributed Computing Systems, San Jose, California, June, 1988, pp.204-111.
7. D. P. Anderson. Boinc: A system for public-resource computing and storage. In R. Buyya, editor, Fifth IEEE/ACM International Workshop on Grid Computing, pages 4-10, 2004.
8. F. Cappello, S. Djilali, G. Fedak, T. Herault, F. Magniette, V. Neri and O. Lodygensky: Computing on Large Scale Distributed Systems: XtremWeb Architecture, Programming Models, Security, Tests and Convergence with Grid FGCS Future Generation Computer Science, 2004.
9. Walfredo Cirne, Francisco Brasileiro, Nazareno Andrade, Lauro B. Costa, Alisson Andrade, Reynaldo Novaes and Miranda Mowbray. Labs of the World, Unite!!! J. Grid Computing 4(3), 2006, pp. 225-246.
10. Shantenu Jha, Hartmut Kaiser, André Merzky, Ole Weidner: Grid Interoperability at the Application Level Using SAGA. eScience 2007: 584-591
11. P. Kacsuk and G. Sipos: Multi-Grid, Multi-User Workflows in the P-GRADE Portal. Journal of Grid Computing, Vol. 3, No. 3-4, Springer Publishers, pp. 221-238, 2005
12. Астафьев А.С., Афанасьев А.П., Лазарев И.В., Сухорослов О.В., Тарасов А.С. Научная сервис-ориентированная среда на основе технологий Web и распределенных вычислений. // Научный сервис в сети Интернет: масштабируемость, параллельность, эффективность: Труды Всероссийской суперкомпьютерной конференции (21-26 сентября 2009 г., г. Новороссийск). – М.: Изд-во МГУ, 2009. – 524 с. (с. 463-467)
13. Посыпкин М.А. Решение задач глобальной оптимизации в среде распределенных вычислений // Программные продукты и системы. № 1. 2010. С. 23-29.
14. E. Urbach, P. Kacsuk, Z. Farkas, G. Fedak, G. Kecskeméti, O.Lodygensky, A. Cs. Marosi, Z. Balaton, Zoltán; G. Caillat, G. Gombás, A.Kornafeld, J. Kovács, H. He, R. Lovas: EDGeS: Bridging EGEE to BOINC and Xtrem Web, Journal of Grid Computing, 2009, Vol 7, No. 3, pages 335 -354
15. O.V. Sukhoroslov. jLite: A Lightweight Java API for gLite. // Distributed Computing and Grid-Technologies in Science and Education: Proceedings of the 3rd Intern. Conf. (Dubna, June 30 - July 4, 2008). - Dubna: JINR, 2008. - 401 p. , pp. 201-204
16. The University of Westminster Local DesktopGrid. URL: http://wgrass.wmin.ac.uk/index.php/Desktop_Grid:Westminster_Local_DG (дата обращения: 13.02.2011).
17. EDGeS@Home Desktop Grid. URL: <http://home.edges-grid.eu/home/> (дата обращения: 13.02.2011)
18. Проекты Центра грид-технологий и распределенных вычислений. URL: <http://boinc.isa.ru/dcsdg/> (дата обращения: 13.02.2011)
19. The DEGISCO project. URL: <http://degisco.eu> (дата обращения: 13.02.2011)
20. The EDGI project. URL: <http://edgi-project.eu> (дата обращения: 13.02.2011)
21. Международная федерация грид-систем персональных компьютеров. URL: <http://desktopgridfederation.org> (дата обращения: 13.02.2011)