Эффективный запуск гибридных параллельных задач в гриде*

А.П. Крюков[1,2,+], М.М. Степанова[3], Н.В.Приходько[4],
Л.В. Шамардин[1], А.П. Демичев[1,2]

[1]Московский государственный университет имени М.В. Ломоносова,
[2]Национальный исследовательский центр «Курчатовский институт»,
[3]Санк-Петербургский государственный университет,
[4]Новгородский государственный университет имени Ярослава Мудрого

В работе рассматривается способ эффективного запуска в гриде гибридных задач, совместно использующих технологии MPI и OpenMP. Для гибкого управления параметрами запуска параллельных задач на суперкомпьютерных (СК) ресурсах была расширена спецификация языка описания задач. Поддержка новых атрибутов реализована для всех ключевых компонентов инфраструктуры. Взаимодействие вебсервиса запуска с локальным менеджером ресурсов организовано через специальные обработчики разных типов заданий (single/openmp/mpi/hybrid), что обеспечивает передачу локальному менеджеру СК правильных параметров для резервирования ресурсов и запуска задачи. Представленное решение было опробовано на грид-полигоне, развернутом на базе промежуточного ПО ГридННС.

1. Введение

Современные прикладные и фундаментальные исследования требуют выполнения высокопроизводительных расчетов, использующих параллельные вычисления, что подразумевает применение вычислительных кластеров, которые поддерживают параллельный режим выполнения программ. Для получения максимальной производительности и скорости исполнения программ в режиме пакетной обработки необходимо запускать такие программы оптимальным образом с учетом особенностей конкретного кластера. При непосредственной работе на кластере пользователю хорошо известна его архитектура и конфигурация, поэтому он может точно указать все параметры ресурсов и запуска задачи. В гриде все сложнее – среда является гетерогенной, а ресурс, на котором будет выполняться задание, в общем случае, заранее неизвестен. Кроме того, сам процесс запуска в гриде существенно сложнее – это многоуровневая процедура, в которой участвует большое количество сервисов и компонентов промежуточного ПО. Выполнение задания в гриде включает:

- определение характеристик задания в описании;
- отправка пользователем задания в грид на сервис распределения нагрузки;
- поиск и выбор подходящего ресурса для выполнения конкретных задач композитного задания;
- передача задания на выбранный сайт;
- резервирование ресурса для задания;
- запуск и выполнение на конкретном ресурсе.

В общем случае, описание задания для запуска в гриде имеет довольно абстрактный вид и не зависит от типа ресурса. Конечным итогом прохождения заданием всех грид-уровней является формирование скрипта задачи, который будет запущен конкретным локальным менедже-

* Работа поддержана грантом РФФИ (№ 11-07-00434-а) и грантом Президента РФ (НШ-3920.2012.2)
+ E-mail: kryukov@theory.sinp.msu.ru

ром (ЛРМ) на конкретном ресурсе. Чтобы ЛРМ мог выполнить корректный запуск, ожидаемый пользователем, описание задачи должно быть исчерпывающим, а алгоритмы его обработки на всех уровнях очень тщательно проработаны.

На сегодняшний день многие грид-инфраструктуры позволяет проводить параллельные расчеты, однако, в каждой из них имеются свои ограничения по запуску параллельных задач по сравнению с обычным кластером. Сравнение возможностей и тестирование ППО современных грид-проектов показывает, что очень трудно обеспечить максимально простой и универсальный механизм, который был бы эффективен для оптимального резервирования ресурсов и обработки разных типов задач в гетерогенной среде [1]. Можно выделить следующие основные моменты, которыми определяются присущие всем гридам ограничения по сравнению с обычным кластером:

- возможности формата описания заданий;
- полнота публикуемой информации и гибкостью алгоритма поиска ресурса;
- различия в типах и конфигурации ЛРМ на сайте;
- реализация интерфейса к ЛРМ;
- специфичность требований к способу запуска и установке окружения для разных типов задач.

Общей проблемой в гриде при выполнении параллельных задач является генерация набора параметров для $MPIEXEC. С одной стороны, набор должен быть согласован с запрошенными ресурсами, с другой – весьма гибким для учета особенностей ПО и правильного запуска. Все существующие реализации с фиксированным набором типов заданий, где параметры запуска автоматически рассчитываются из требований к ресурсам, привлекают простотой с точки зрения описания заданий, но подходят лишь под ограниченный круг задач. В частности, это относится к разработкам, в основе которых лежит ПО GlobusToolkit, в том числе и к российскому проекту ГридННС [2].

Другие известные проекты без дополнительной конфигурации также пока обеспечивают лишь базовую функциональность для простейших MPI- и OpenMP-задач.

Так, в gLite в качестве языка описания заданий используется JDL. На грид-шлюзе (CreamCE), начиная с версии glite-ce-cream v.1.13, для резервирования ресурсов введен новый расширенный набор атрибутов: CPUNumber (полное число запрашиваемых cpu), SMPGranularity (минимальное число ядер на узел), HostNumber (полное число запрашиваемых узлов), WholeNodes (полное резервирование узла, т.е. все ядра). Для запуска параллельных заданий предназначен набор скриптов MPI-Start [3] – пользователь должен передать скрипт с явно установленными параметрами и вызовом MPI-Start. Итерфейс MPI-Start в сочетании с новым набором атрибутов, по сути, дает унифицированную связку с любым ЛРМ. Теоретически можно запустить все, но требуется очень аккуратная настройка как сайта, так и параметров запуска пользователем.

В проекте NorduGrid ARC описание заданий выполняется на языке xRSL. Для резервирования cpu в описании имеется единственный атрибут count, который в зависимости от настройки сайта может интерпретироваться как число узлов или число ядер. Более гибкий вариант резервирования на сайте может быть доступен путем настройки runTimeEnvironment (RTE). RunTimeEnvironment – это механизм установки среды на основе shell-скриптов. Три вызова (до создания PBS submit-скрипта, перед запуском задачи и после завершения) позволяют очень гибко настроить исполняющую среду для любых приложений. Однако, использование такого способа для резервирования ресурсов не является лучшим вариантом, поскольку требует либо унифицированной поддержки в рамках VO, либо знания пользователем деталей конфигурации конкретных сайтов

Самым законченным решением на сегодняшний день следует признать проект UNICORE, который имеет наиболее продуманную спецификацию описания заданий и ее полную поддержку на грид-шлюзе. Характерная особенность – четкое разделение функциональности, а именно: параметры описания из раздела Resources полностью определяют резервирование, а через механизм программных окружений ExecutionEnv можно точно задать параметры и опции запуска

задачи. Это позволяет обеспечить поддержку корректного запуска практически любых параллельных задач и специализированных программных пакетов.

Данное исследование направлено на разработку для инфраструктуры ГридННС универсального способа запуска сложных параллельных задач. Предлагаемое решение позволяет использовать грид-среду для эффективного запуска не только распространенных вариантов на базе MPI- и OpenMP-технологий, но и наиболее сложного комбинированного типа - гибридных (MPI+OpenMP)-задач.

Структура работы следующая. Во втором разделе описаны особенности гибридных задач и используемые методы их запуска. Третий раздел посвящен описанию методов запуска разных типов параллельных задач в среде ГридННС. Четвертый и пятый разделы описывают некоторые аспекты реализации предложенных методов. В шестом разделе представлены результаты тестов. В заключении перечислены основные результаты работы и возможные направления их развития.

2. Особенности гибридных задач и методы их запуска

Классические параллельные приложения реализуются на основе технологий MPI [4] или OpenMP [5]. Вариант на базе MPI хорошо зарекомендовал себя для работы на традиционных кластерных системах. OpenMP предназначен для распараллеливания на многоядерных узлах с общей памятью. Наиболее типичные варианты запуска параллельных задач на кластере из многоядерных узлов обсуждаются в [1].

Самым сложным из параллельных типов являются гибридные задачи. Особый интерес к ним вызван тенденцией к использованию для высокопроизводительных вычислений многоядерных архитектур и SMP-кластеров. Одним из наиболее эффективных подходов программирования для таких кластеров является гибридный, основанный на комбинированном использовании MPI и OpenMP. Гибридный подход предполагает, что алгоритм разбивается на параллельные процессы, каждый из которых сам является многопоточным. Таким образом, имеется два уровня параллелизма: параллелизм между MPI процессами и параллелизм внутри MPI процесса на уровне потоков. В такой модели программирования MPI используется для организации обмена между процессами, а OpenMP для многопоточного взаимодействия внутри процесса (в рамках одного узла). Как показывает практика [6-10], за счет укрупнения MPI-процессов и уменьшения их числа гибридная модель может устранить ряд недостатков MPI, таких как большие накладные расходы на передачу сообщений и слабая масштабируемость при увеличении числа процессов. Однако, производительность гибридной задачи очень сильно зависит от режима ее запуска и выполнения, который определяет соотношение числа MPI-процессов и OpenMP-потоков на одном вычислительном узле, а также способа привязки MPI-процессов к физическим процессорам системы. В случае неправильного запуска или некорректного выделения ресурсов производительность таких программ может существенно снижаться.

Стандартный способ организации многопользовательского доступа к вычислительным кластерам – использование системы управления пакетной обработкой заданий (например, Torque, SLURM, PBSPro и др.) Важнейшим аспектом для запуска параллельных и особенно гибридных задач является поддержка локальным менеджером корректного выделения и резервирования вычислительных ресурсов на время выполнения задачи. В частности, должно обеспечиваться управление динамическим распределением задач в больших системах и строгое ограничение процессов на подмножестве процессоров и памяти узла. Один из возможных механизмов такого типа – cpuset, реализованный на уровне ядра ОС Linux и доступный в Torque. Данный механизм позволяет "на лету" распределять вычислительные ресурсы между различными задачами, ограничивая использование оперативной памяти и процессоров/ядер "вычислительным доменом", которому присвоена задача.

Гибридные задачи являются разновидностью MPI-задач, поэтому их запуск выполняется с помощью утилиты mpirun (или mpexec):

```
mpirun [ options ] <program> [ <args> ]
```

Предварительно должно быть определено необходимое окружение (пути к библиотекам, переменные окружения и т.п.), а так же обеспечено количество свободных ресурсов в соответствии с указанными при запуске опциями [options].

Требования к ресурсам и, соответственно, оптимальный способ запуска гибридной задачи в значительной степени определяется ее кодом. Большинство программ разрабатываются без привязки к архитектуре кластера, на котором она будет компилироваться и выполняться. Для их нормального выполнения достаточно задать полное количество MPI-процессов (M) и количество потоков на один процесс (K), а также обеспечить монопольное резервирование на время выполнения MxK процессоров/ядер.

В Таблице 1 приведены различные варианты запуска гибридных задач на кластере. Наиболее универсальный метод состоит в том, чтобы обеспечить распределение одного MPI-процесса на один узел, при этом число OpenMP-потоков внутри процесса не должно превышать количество ядер на этом узле (строка 1 в Таблице 1). Такой способ приемлем на любых ЛРМ, в том числе, без продвинутой поддержки управления динамическим распределением. Однако, его нельзя считать эффективным на кластерах, состоящих из узлов с большим количеством cpu, поскольку в этом случае задача будет потреблять неоправданно большие ресурсы (занимает узлы полностью). Строка 2 из Таблицы 1 содержит другой вариант запуска, когда на одном узле размещается несколько многопоточных процессов. Такой способ повышает полезную загрузку кластера, но уже предъявляет высокие требования к ЛРМ. В третьей строке Таблицы 1 приведен общий формат запуска задачи со строгой привязкой к ресурсам. Это необходимо для программ, код которых оптимизирован под конкретную архитектуру вплоть до точной привязки распределения процессов к топологии сокетов и ядер на узлах. Такой вариант требует, во-первых, знания архитектуры на которой код компилируется и запускается, во вторых, также предъявляет высокие требования к ЛРМ.

Таблица 1. Методы запуска гибридных (MPI+OpenMP) задач на вычислительном ресурсе

	Метод запуска	Резервирование ресурсов (cpu) и параметры запуска
1	Запуск в режиме один MPI-процесс на узел при полном резервировании узла	`#PBS -l nodes=N:ppn=K, где K=SMPSize;` `$MPIEXEC -pernode ./hyb_task`

2	Запуск в режиме на один узел M MPI-процессов, каждый с числом потоков K	`#PBS -l nodes=N:ppn=K,`
		`export OMP_NUM_THREADS=L, где M*L=K`
		`$MPIEXEC -npernode M ./hyb_task`

3	Запуск в режиме отображения на ресурсы, задаваемого строкой опций	`#PBS -l nodes=N:ppn=K,`
		`export OMP_NUM_THREADS=L, где M*L=K`
		`$MPIEXEC[special_option_string] ./hyb_task`

3. Запуск параллельных задач в ГридННС

3.1 Общая архитектура ГридННС с точки зрения прохождения задания

ГридННС включает ряд базовых компонентов ПО GlobusToolKit-4, а также набор ключевых инфраструктурных RESTful-грид-сервисов оригинальной разработки. Подробности архитектуры среды ГридННС и детали реализации отдельных сервисов представлены в [2, 11].

Рассмотрим ее с точки зрения функционирования компонент, определяющих запуск задания. Общая схема прохождения задания представлена на Рисунке 1.

1. Пользователь составляет описание задания в формате JSON. Синтаксис описания задания является унифицированным и не зависит от типа локального менеджера ресурсов.
2. С помощью интерфейса управления заданиями pilot-cli (или веб-интефейса ВИГ) передает его системе распределения и контроля заданий – "Пилот", который выполняет все

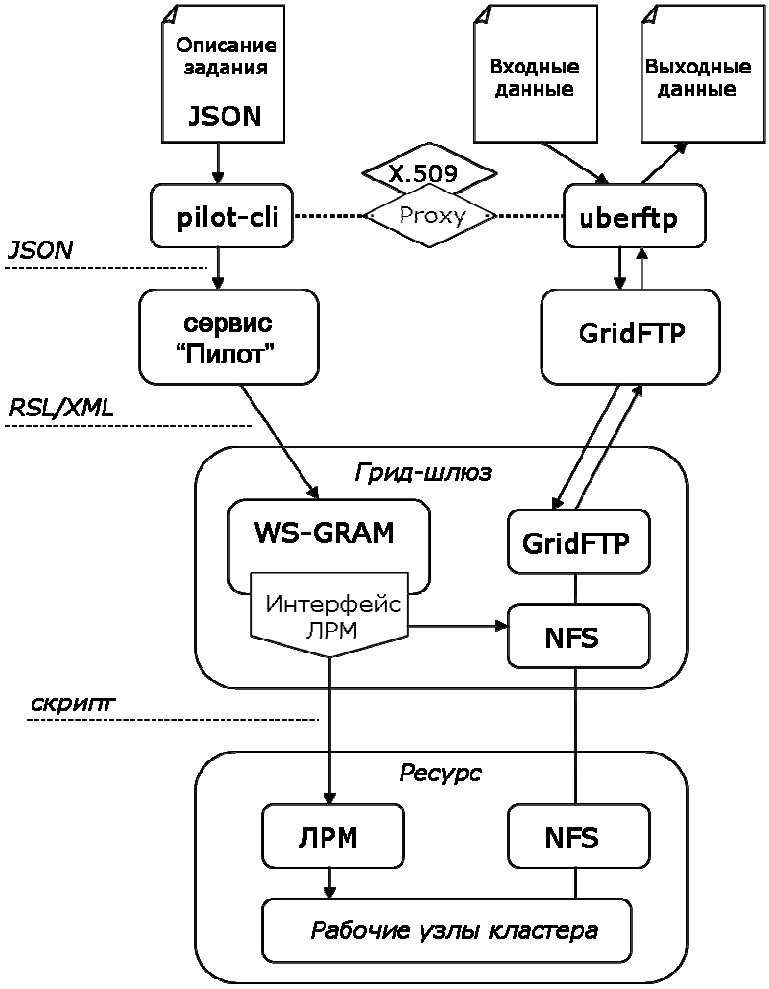

Рис. 1. Схема прохождения задания в ГридННС

функции брокера. Проверка корректности синтаксиса описания выполняется на уровне pilot-cli.

3. "Пилот" [12] реализован в виде комплекса RESTful-грид-сервисов, который обеспечивает запуск и координацию выполнения задач вычислительных ресурсах конкретного сайта. В частности, выполняются следующие действия:

   - аутентификация и авторизация на основе сертификата пользователя и его членства в BO;
   - разбор JSON-описания задания;
   - запрос к информационному сервису о наличии ресурсов-кандидатов, удовлетворяющих критериям из описания задания;
   - выбор конкретного ресурса и подбор необходимых параметров запуска;
   - преобразование JSON-описания с учетом выбранных дополнительных параметров задачи в формат RSL/XML, используемый грид-менеджером ресурсов на грид-шлюзе;
   - передача на грид-шлюз для запуска;
   - отслеживание статуса состояния задач.

4. Грид-менеджер на шлюзе (построен на основе WS-GRAM из GlobusToolkit) принимает RSL-описание задачи, по которому интерфейс ЛРМ формирует скрип запуска для локального менеджера и передает его на ЛРМ.
5. ЛРМ запускает задачу на выполнение на вычислительных узлах кластера.

Исходная реализация ГридННС обеспечивает запуск задач только двух типов:

- "single": простая задача с последовательным кодом; запускается на одном ядре;
- "mpi": параллельная MPI-задача; запускается в режиме один MPI-процесс на одно ядро; для резервирования ресурса и запуска пользователю доступен один параметр описания – count, который определяет количество запрашиваемых ядер и задает число запускаемых процессов.

Отметим, что тип задачи пользователь не задавал явным образом в описании, он присваивался на стадии ее обработки сервисом "Пилот" в зависимости от значения параметра `count`: если `count` >1, то тип "mpi", если `count=1` или не задан, то "single". При формировании скрипта запуска для ЛРМ на грид-шлюзе выполнялась автоматическая подстановка способа запуска. Так, в случае "mpi" - посредством "`$MPIEXEC –n count <executable>`".

Очевидно, что поддержка более сложных вариантов запуска параллельных задач требует расширения возможностей языка описания заданий и новых алгоритмов их обработки для обеспечения резервирования ресурсов и запуска задачи. Наиболее простой путь состоит в расширении функциональности грид-сервисов для поддержки ряда новых типов заданий. Конкретная реализация необходимых новых компонентов определяется как текущим программным решением, так и используемыми внешними стандартами и спецификациями.

Синтаксис языка описания задания с представлением в формате JSON достаточно гибкий, и его расширение новыми атрибутами не вызывает затруднений. Однако, следует учитывать, что в дальнейшем это описание транслируется в RSL для обработки на грид-шлюзе сервисом WS-GRAM, а спецификация RSL допускает это только через механизм extensions.

Это потребовало существенной доработки в части алгоритмов обработки JSON-описания, поиска ресурса, подбора параметров запуска и формирования RSL важнейшего инфраструктурного сервиса "Пилот".

При выборе ресурса необходима исчерпывающая информация о текущем состоянии сайтов, которая предоставляется посредством информационного сервиса, использующего модифицированный вариант XML-реализации схемы GLUE 1.3. Добавление, при необходимости, новых элементов в публикуемую сайтами информацию о ресурсах не представляет больших сложностей.

Пользовательский интерфейс требует изменений в части проверки корректности синтаксиса JSON-описания задания.

Наконец, существенная модификация необходима для интерфейсов ЛРМ. Во-первых, нужен алгоритм для обработчика нового типа заданий. Во-вторых, расширение набора атрибутов описания задачи, усложнит и потребует модификации кода существующих обработчиков.

Принципы, положенные в процессе проектирования ГридННС позволили достаточно просто провести доработку ПО, чтобы обеспечить запуск дополнительных типов задач. В следующих разделах будут определены базовые типы задач и спецификации для их поддержки.

3.2 Типы задач

Для адекватной работы большинства пользовательских приложений и специализированных пакетов можно выделить следующий набор базовых типов задач, которые должна обеспечивать современная грид-инфраструктура:

- "single" – запуск задачи с последовательным кодом на одном ядре узла вычислительного кластера.
- "openmp" – запуск многопоточной (например, использующей технологию OpenMP) задачи на одном узле с возможностью резервирования под задачу узла либо целиком,

либо частично – резервирование запрошенного числа ядер на узле для монопольного использования задачей.

- "mpi" – запуск MPI-задачи с одним процессом на ядро без каких-либо требований к распределению ядер по узлам кластера.
- "hybrid" – запуск гибридной (MPI+OpenMP) задачи в режиме один многопоточный MPI-процесс на один узел.

3.3 Расширение языка описания задач и общие спецификации на реализацию сервисов

Спецификация описания задач была расширена следующим образом:

```
Секция definition:
   jobtype [single|openmp|mpi|hybrid]
   nodes [int]
   ppn [int]

Секция extensions:
   mpi_extra_arg [string]
```

Рис. 2. Расширение набора атрибутов описания задач

В секцию definition описания задачи вводится атрибут jobtype. Разрешенными типами заданий являются "single", "mpi", "openmp", "hybrid". В случае указания параметра jobtype сервис "Пилот" передает этот параметр на шлюз без изменений. Фактическая обработка параметра jobtype осуществляется на стороне шлюза в модуле стыка шлюза и ЛРМ. В этой же секции definitions добавляется возможность указания атрибутов для описания требований к ресурсам: nodes – число запрашиваемых задачей узлов и ppn – необходимое количество ядер на узле.

В случае, когда значения этих параметров заданы пользователем в описании задачи, они учитываются сервисом "Пилот" и шлюзом на разных стадиях жизни задания.

Так сервис "Пилот" использует параметры nodes и ppn на стадии выбора ресурса и, в общем случае, они должны учитываться совместно с count (общее число ядер, запрашиваемое для задачи). Для каждого из заданных параметров в описании задачи "Пилот" подходящими считаются только ресурсы со значением этого параметра >= "заданного". Если запрошенный набор не может быть удовлетворен хотя бы по одному из параметров, то задание не будет запущено, а пользователь получит сообщение - "нет подходящих ресурсов".

При поступлении задачи на грид-шлюз эти параметры учитываются им на стадии обработки модулем стыка шлюза с ЛРМ. Способ их обработки при формировании скрипта запуска на ЛРМ определяется атрибутом jobtype.

На стороне шлюза также реализуется поддержка двух вариантов обработки параметра ppn для целей резервирования – либо в точном соответствии с указанным ppn, либо резервирование узлов целиком, в зависимости от наличия поддержки cpuset в ЛРМ.

Также в секцию extensions добавляется поддержка (в настоящее время в статусе "experimental") расширения mpi_extra_args, в которой может передаваться строка альтернативных аргументов для $MPIEXEC. Для включения поддержки на шлюзе в jobmanager-pbs.conf указывается параметр use_mpi_extra_args (значения "no" или "yes", по умолчанию "no"). Использование значения "yes" для этой опции не рекомендуется для сайтов без поддержки cpuset, так как это может привести к потере контроля над ресурсами. Данный параметр предназначен для квалифицированных пользователей, которые хорошо представляют последствия своих действий.

В Таблице 2 приведены наборы атрибутов, допустимые к использованию для разных типов задач. Если в описании задания есть несоответствие в наборе обязательных параметров, то задание не будет отправлено в грид, а пользователь получит сообщение об ошибке на стадии валидации JSON-описания задания.

Таблица 2. Типы задач, поддерживаемые в ГридННС

Тип задач	Обязательные параметры в описании задачи	Дополнительные (необязательные) параметры в описании задачи
single	Нет	Нет
openmp	jobtype="openmp" ppn или count	environment: {"OMP_NUM_THREADS": <number>}
mpi	jobtype="mpi" count или пара {ppn,nodes}	Нет
hybrid	jobtype="hybrid" любая пара из {ppn,nodes,count} */ рекомендуется {ppn,nodes}	mpi_extra_args: <string> environment: {"OMP_NUM_THREADS": <number>}

4. Адаптация ПО "Пилота" для запуска гибридных задач

Базовая версия грид-сервиса "Пилот", распространяемая в составе комплекса программного обеспечения ГридННС, поддерживает две разновидности задач: однопроцессорные ("single") и обычные MPI-задачи ("mpi"). Добавление новых типов задач "openmp" и "hybrid" потребовало изменение алгоритма работы сервиса. Также был модифицирован алгоритм обработки задач типа "mpi" для учета дополнительных атрибутов ppn и nodes в процессе подбора ресурсов, удовлетворяющих требованиям задачи.

В схемы описания задач JSON Schema были добавлены определения новых полей ppn и nodes. Поскольку JSON Schema не позволяет задать сложные отношения между атрибутами, были также расширены правила верификации задач используемые загрузчиком описаний задач "Пилот". Проверяется выполнение следующих правил:

- для задач типа "single" проверяется отсутствие атрибутов ppn, nodes и count;
- для задач типа "mpi" проверятся наличие атрибута count, либо пары атрибутов nodes и ppn;
- для задач типов "openmp" и "hybrid" проверяется наличие атрибутов nodes и ppn.

Данная проверка правил выполняется как на стороне клиента при запуске задачи, так и на стороне сервера.

Также модификации потребовали модули генерации описания задач и выбора подходящих ресурсов на стороне сервера "Пилот". Поскольку язык промежуточного уровня RSL, используемый Globus Toolkit 4 не поддерживает атрибутов задач, логически соответствующих параметрам ppn и nodes, все атрибуты, связанные с запуском гибридных параллельных задач, передаются через расширения в описании задачи, что делает их доступными для интерпретации ПО грид-шлюза.

В модуле выбора подходящих ресурсов были внесены изменения, накладывающие следующие ограничения на выбор подходящих ресурсов. Во-первых, для всех задач рассчитываются значения недостающих атрибутов, если один из атрибутов count, ppn или nodes не указан. Во-вторых, выполняется проверка, что:

- количество процессов, запускаемых на одном узле (ppn) не превышает количества логических процессоров на узле ("logical slots" в информационной системе);
- суммарное количество узлов, необходимое для запуска задачи не превышает общее количество узлов в кластере ("physical slots").

Для задач типа "mpi", в которых указание ppn и nodes является опциональным, предполагается максимально плотная упаковка задач на узлах.

5. Взаимодействие грид-шлюза с менеджером локальных ресурсов

Взаимодействия веб-сервиса запуска с локальным менеджером ресурсов организовано через грид шлюз, который использует GRAM ПО Globus Toolkit. Из требуемых для запуска гибридных задач параметров count, ppn и nodes Globus Toolkit непосредственно поддерживает только передачу параметра count. Поэтому в спецификацию (см. п.3.3) введены атрибуты nodes и ppn, по которым "Пилот" генерирует расширение в RSL-описании, а на грид-шлюзе реализована его обработка в соответствии с типом заданий.

При формировании задачи для ЛМР на стороне шлюза реализована поддержка двух вариантов обработки параметра ppn для целей резервирования:

- обработка заданного значения ppn, то есть резервирование части узла; данный вариант может использоваться только в случае ЛРМ с поддержкой cpuset;
- замена заданного ppn на ppn_max, то есть резервирование узлов полностью; данный вариант должен использоваться в случае ЛРМ без поддержки cpuset.

Способ обработки ppn задается в конфигурационном файле jobmanager-pbs.conf через параметр use_full_node (значения "no" или "yes"; по умолчанию "yes").

С целью поддержки дополнительных опций $MPIEXEC также добавлена поддержка дополнительного параметра mpi_extra_args, в которой передается строка альтернативных аргументов. Возможность использования данного параметра определяется через параметр use_mpi_extra_args в конфиграционном файле jobmanager-<lrm>.conf.

Переданные параметры count, ppn, nodes, mpi_extra_args, учитываются шлюзом на стадии обработки модулем стыка шлюза с локальным менеджером ресурсов. Способ их обработки при формировании скрипта запуска на ЛМР определяется заданным jobtype. Рассмотрим различные значения параметра jobtype.

- **jobtype=single**

Запуск задания предполагает запуск одного однопоточного процесса на одном ядре рабочего узла кластера. Схема запуска повторяет стандартную схему запуска Globus Toolkit.

- **jobtype=mpi**

В этом случае выполняется запуск MPI-задачи с одним процессом на ядро без каких-либо требований к распределению ядер по узлам кластера. При генерации задания учитываются параметры ppn, nodes, count. Схема резервирования ресурсов ЛМР и запуска определяется следующим образом:

- если задан только count или только nodes, то резервирование и запуск осуществляется по стандартному алгоритму Globus Toolkit;
- если задан только ppn, то производится резервирование "1:<ppn_max>" и запуск "$MPIEXEC –n <ppn>";
- если заданы count+ppn, то производится резервирование "1:<ppn>" и запуск "$MPIEXEC –n <count>";
- если заданы count+nodes+ppn, count+nodes, nodes+ppn, то производится резервирование "<nodes>:<ppn>" и запуск "$MPIEXEC –n <count>". При этом неопределенный параметр определяется на основании соотношения count=nodes*ppn.

- **jobtype=openmp**

Запуск задания предполагает запуск одного многопоточного процесса на одном рабочем узле кластера. При генерации задания учитываются параметры ppn и count. Последний используется, если параметр ppn не указан. Дополнительно учитывается параметр

OMP_NUM_THREADS, переданный в разделе "extension.environment". Задание Torque/PBS запускается с резервированием nodes=1:<ppn>. В задание передается переменная среды OMP_NUM_THREADS со значением ppn или значением, переданным в "extension.environment", если оно указано.

- **jobtype=hybrid**

В этом случае предполагается, что будет запущена гибридная (MPI+OpenMP) задача в режиме один многопоточный mpi-процесс на один рабочий узел кластера. При генерации задания учитываются параметры ppn, nodes, count. При этом необходимо указание двух из трех параметров, оставшийся параметр автоматически рассчитывается из соотношения count=nodes*ppn. Задание Torque/PBS запускается с резервированием nodes=<nodes>:<ppn>. Дополнительно в задание передается переменная окружения OMP_NUM_THREADS со значением ppn или значением, переданным в "extension.environment", если оно указано. Если в задании указан параметр mpi_extra_args и его поддержка доступна на ресурсе, то запуск производится командой "$MPIEXEC $mpi_extra_args", в противном случае выполняется "$MPIEXEC -pernode".

6. Проверка методов запуска гибридных задач на полигоне ГридННС

Разработанное программное решение прошло тестирование на полигоне ГридННС, включающем:

- компьютер с пользовательским интерфейсом командной строки (pilot-cli);
- сервер с сервисом "Пилот";
- два сайта в конфигурации: шлюз ГридННС + ЛРМ Torque +кластер из 8-ядерных узлов. На одном сайте использовалась сборка Torque 3.0.4 с поддержкой cpuset, на другом – без такой поддержки.

В состав тестового набора входили исходные коды типичных параллельных программ (OpenMP-, MPI- и гибридных(MPI+OpenMP)), разработанных таким образом, чтобы выходные данные содержали подробную диагностику о количестве запущенных процессов/потоков и используемых ресурсах.

Все тестовые задания оформлялись в виде группы из двух задач – компиляции и параллельного запуска программы. Такое объединение задач компиляции и запуска в группу, означает, что они будут выполнены на одном и том же сайте, причем вторая задача является потомком первой. Резервирование ресурсов для этих задач происходит независимо. Задача компиляции запускалась как "single", а вторая - в соответствие с типом, указанным в описании ("openmp/mpi/hybrid").

Пример описания теста гибридной задачи представлен на рисунке 2. В данном примере задача COMPILE будет обработана по типу "single", а задача RUN в соответствие с типом "hybrid". Критерий корректного резервирования ресурсов и правильного запуска для этого теста следующий: количество запущенных mpi-процессов совпадает с числом выделенных узлов кластера (nodes), причем на каждом узле исполняется ровно один процесс, а количество omp-потоков, которое порождает каждый mpi-процесс, не превышает полного числа ядер узла.

Независимо от особенностей конфигурации ЛРМ сайтов (см. выше) разработанные алгоритмы обеспечивают правильный запуск. При этом сайт с поддержкой cpuset может более рационально использовать узлы кластера, а именно задействовать их не целиком, а в точном соответствии с заданными требованиями задачи.

```
{ "version": 2,
  "default_storage_base": "gsiftp://phys27.gridzone.ru/tmp/",
  "tasks": [
      { "id": "COMPILE",
        "children": [ "RUN" ],
        "definition":
        { "version": 2,
          "requirements": { "software": "mpich2" },
          "executable": "/bin/bash",
          "arguments":["-c",
             "mpicc -fopenmp -o test-hybrid test-hybrid.c"
                      ],
          "input_files": { "test-hybrid.c": "test-hybrid.c" },
          "output_files": { "test-hybrid": "test-hybrid" }
        },
       },
       { "id": "RUN",
         "definition":
         { "version": 2,
           "requirements": { "software": "mpich2" },
           "jobtype": "hybrid",
           "nodes": 4,
           "ppn": 3,
           "executable": "/bin/bash",
           "arguments": [ "-c", "./test-hybrid" ],
           "input_files": { "test-hybrid": "test-hybrid" },
           "stdout": "test-hybrid.out"
         }
       }
          ],
  "groups": [ [ "COMPILE", "RUN" ] ] }
}
```

Рис. 3. Пример описания тестового задания, включающий группу из двух задач – компиляция и запуск гибридной программы

Результаты проведенных тестов вместе с мониторингом состояния очередей и реальной загрузки кластерных узлов показали, что запуски всех типов задач приводили к корректному резервированию ресурсов кластера в соответствии с указанными в описании параметрами.

7. Заключение

Целью выполнения данной работы являлось повышение эффективности использования вычислительных ресурсов суперкомпьютерных центров, объединенных в грид-инфраструктуру с использованием ГридННС.

В ходе выполнения работы были систематизированы и формализованы методы запуска гибридных задач на вычислительном ресурсе, определены расширения спецификации языка описания заданий набором атрибутов, необходимых и достаточных для определения требований к выделяемым ресурсам, программной среде и параметрам запуска.

На основании этих результатов были разработаны спецификации и алгоритмы обработки новых атрибутов компонентами грид-сервисов на всех уровнях и выполнена программная реализация этих алгоритмов для грид-сервисов в среде ГридННС.

Проведенные тестовые испытания на полигоне ГридННС показали, что предложенные методы обеспечивают корректное резервирование ресурсов на вычислительном кластере в соответствии с типом запускаемой задачи.

Предложенное решение является достаточно общим подходом и может быть перенесено в другие грид-инфраструктуры. Разработанные методы позволяют существенно расширить спектр пользовательских грид-приложений, а также повысить качество обработки заданий и эффективность использования СК ресурсов.

Таким образом, полученные результаты позволяют повысить эффективность использования суперкомпьютеров, подключенных к гриду, за счет учета специфики параллельных задач и, как следствие, позволят пользователям более быстро выполнять инженерные и научные исследования.

В дальнейшем предполагается, что данная методика будет распространена на задачи с использованием графических процессоров, которые получают широкое распространение на современных вычислительных установках.

Литература

1. M.M. Stepanova, O.L. Stesik. Running Parallel Jobs on the Grid // Distributed Computing and Grid-Technologies in Science and Education: Proceedings of the 5rd Intern.Conf. (Dubna, 16-21 July, 2012).– Dubna: JINR, 2012, pp.383-387, ISBN -5-9530-0345-2
2. В.А. Ильин, В.В. Кореньков, А.П. Крюков. ГридННС: состояние и перспективы // Труды 5-й международной конференции "Распределенные вычисления и Грид-технологии в науке и образовании" (Дубна, 16-21 июля, 2012 г.).-Дубна: ОИЯИ, с. 332-336
3. MPI-Start // URL: http://grid.ifca.es/wiki/Middleware/MpiStart/
4. OpenMP Application Program Interface Version 3.1, July 2012 // URL: http://www.openmp.org/mp-documents/OpenMP3.1.pdf
5. MPI: A Message-Passing Interface Standard Version 2.2, September 2009 // URL: http://www.mpi-forum.org/docs/mpi-2.2/mpi22-report.pdf
6. Makris I. Mixed Mode Programming on Clustered SMP Systems // MSc in. High Performance Computing. The University Of Edinburgh, 2005.
7. R. Rabenseifner, G. Hager, and G. Jost. Hybrid MPI/OpenMP parallel programming on clusters of multi-core SMP nodes // In Proc. of 17th Euromicro Int'l Conference on Parallel, Distributed, and Network-Based Processing (PDP 2009), pages 427–236, 2009.
8. A. Rane and D. Stanzione. Experiences in tuning performance of hybrid MPI/OpenMP applications on quad-core systems // In Proc. of 10th LCI Int'l Conference on High-Performance Clustered Computing, 2009.
9. Глазкова Е.А., Попова Н.Н.Анализ эффективности гибридного параллельгого программированиия на примере системы BLUE GENE/P 2009 // URL: http://agora.guru.ru/abrau2009/pdf/36_NSSI_2009_Abrau-2009.pdf
10. MPI Forum Hybrid Programming Working Group (Lead: Pavan Balaji) // URL: http://meetings.mpi-forum.org/mpi3.0 hybrid.php (July 2010).
11. Крюков А.П., Демичев А.П., Ильин В.А., Шамардин Л.В., Основные подходы к построению грид-инфраструктуры национально нанотехнологической сети // В сборнике Вычислительные технологии в естественных науках. Перспективные компьютерные системы: устройства, методы и концепции. Труды семинара, Таруса 24 марта 2011 г., Под ред. Р.Р.Назирова, Л.Н.Щура, ИКИ РАН, серия Механика, управление и информатика, с. 51-68
12. Демичев А.П., Ильин В.А., Крюков А.П., Шамардин Л.В., Реализация программного интерфейса грид-сервиса Pilot на основе архитектурного стиля REST // Вычислительные методы и программирование, том 11, с. 62-65