Объединение вычислительных кластеров для крупномасштабного численного моделирования в проекте NumGRID*

М.А. Городничев

Институт вычислительной математики и математической геофизики СО РАН

В работе представлен программный комплекс NumGRID для организации вычислений на объединении высокопроизводительных вычислительных кластеров в целях крупномасштабного численного моделирования. Дан анализ проблем организации распределенных вычислений на кластерах и обзор родственных проектов. Продемонстрированы результаты экспериментального исследования системы NumGRID.

1. Введение

Программный комплекс NumGRID[1] предназначен для объединения разнородных вычислительных кластеров в единый вычислительный ресурс на основе частичной реализации стандарта MPI-2.2 [2]. NumGRID обеспечивает возможность запускать MPI-приложение так, чтобы процессы приложения были распределены по рабочим узлам нескольких кластеров. При этом процессы составляют один коммуникатор MPI и имеют возможность обмениваться между собой средствами MPI.

NumGRID позволяет

- решать задачи, для которых недостаточно ресурсов отдельных кластеров,
- продлять жизнь устаревающего оборудования за счет объединения с новым,
- распределять части комплексных задач между специализированными кластерами в соответствии с индивидуальными требованиями частей к оборудованию и программному обеспечению,
- повысить гибкость при планировании распределения задач между кластерами в грид,
- планировать постепенное наращивание мощностей распределенной системы.

Схема устройства объединенного вычислительного ресурса NumGRID представлена на рис. 1.

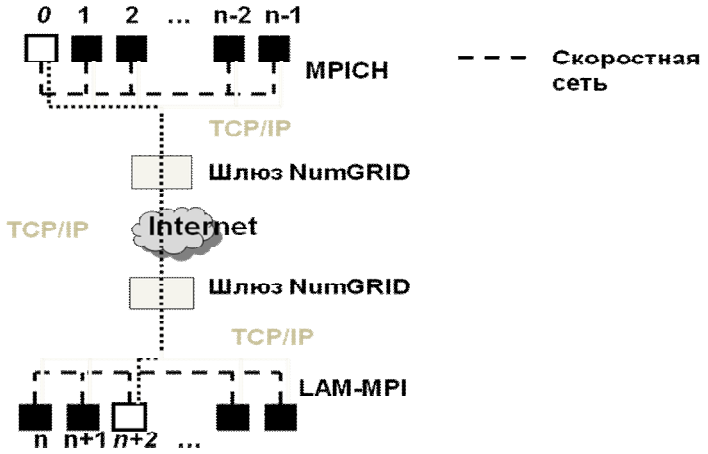

Рис. 1. Устройство NumGRID

В примере объединяются два кластера через Internet. Каждый кластер имеет рабочие узлы, объединенные высокоскоростной сетью (Myrinet, Infiniband, др.). Также узлы связаны с голов-

* Проект поддержан грантом РФФИ №10-07-00454а

ным узлом сетью с меньшей пропускной способностью, поддерживающей протокол TCP. Процессы приложения MPI, находящиеся на рабочих узлах одного кластера обмениваются между собой через высокоскоростную сеть средствами специфичной для кластера библиотеки MPI, позволяющей использовать возможности высокоскоростной сети. В примере это библиотеки MPICH и LAM-MPI. Для процессов, распределенных по рабочим узлам двух кластеров, поддерживается глобальная адресация в рамках одного коммуникатора MPI. Сообщения между процессами, расположенными на разных кластерах, проходят путь через головные узлы кластеров, где для этих целей перед стартом приложения запускаются шлюзы.

Пользователь вначале запускает шлюзы на каждом кластере. Шлюзы устанавливают между собой связь. Пользователь загружает на головные узлы кластеров исходный код своего приложения и исходные данные, собирает приложение на каждом кластере с библиотекой NumGRID-MPI, запускает необходимое число процессов на каждом кластере как локальные задачи MPI. Локальная задача получает в параметрах информацию о месте данной группы процессов в общей распределенной задаче, что позволяет обеспечить глобальную адресацию процессов. В конце работы пользователь забирает с каждого кластера результаты работы программы.

В разделе 2 ставится задача объединения вычислительных кластеров, формулируются требования к программному комплексу NumGRID, дается анализ проблем организации распределенных вычислений и обзор родственных проектов. В разделе 3 излагаются принципы организации программного комплекса NumGRID. В разделе 4 представлены результаты экспериментов по выполнению вычислений на объединении вычислительных кластеров.

2. Задача объединения вычислительных кластеров

2.1 Объединение вычислительных систем сегодня

В целях обеспечения ученых вычислительными ресурсами разрабатываются сверхмощные суперкомпьютеры [3], развиваются инфраструктурные проекты по организации доступа ученых к распределенным вычислительным ресурсам (грид) [4-6], к вычислительным центрам коллективного пользования [7,8], создаются распределенные вычислительные системы на основе объединения персональных компьютеров добровольцев [9,10] через Интернет. Широкое распространение получили относительно малые вычислительные системы (порядка десятков TFlops) кластерного типа (кластеры), которые приобретают университеты, институты и прочие организации. Такие малые системы являются наиболее доступными для ученых средствами решения вычислительных задач. Обычно, системы используются в многопользовательском режиме, и пользователи ставят свои задачи в очередь. Такого рода системы также объединяют в грид с целью распределения пользовательских задач между вычислительными ресурсами дружественных организаций. Однако, как правило, такое объединение предполагает, что задача пользователя назначается целиком в один из кластеров, а распределение процессов параллельной задачи по вычислительным узлам нескольких кластерам невозможно: штатные средства современных кластеров спроектированы так, чтобы обеспечивать высокоскоростные коммуникации между процессами, расположенными на узлах кластеров, и не предусматривают общения процессов между кластерами. Такое положение дел имеет исторические причины: во-первых, кластерные системы появились раньше, чем желание их объединить, и, во-вторых, существенно худшая производительность сетей связи между кластерами по сравнению с производительностью внутренних сетей кластеров делала нецелесообразным запуск параллельных программ с распределением процессов между кластерами. Считалось, что для распределенного счета подходят только задачи, где отсутствует необходимость в коммуникациях между процессами, либо коммуникации крайне малы. Инструменты для решения таких задач были разработаны [9, 11].

2.2 Зачем распределять процессы параллельных задач между кластерами?

Какие выгоды дала бы возможность запускать параллельные задачи с распределением процессов между кластерами? Во-первых, большее количество процессоров и памяти для одной задачи позволили бы увеличить масштаб решаемых задач. Во-вторых, возможность произвольного распределения процессов повысила бы гибкость при планировании распределения задач между кластерами в грид. В-третьих, появилась бы возможность вовлекать в расчеты устаревающие вычислительные комплексы, собственная производительность которых недостаточна для актуальных задач. В-четвертых, появилась бы возможность учитывать наличие специализированного оборудования или программного обеспечения, в т.ч. дорогостоящего, на тех или иных кластерах при распределении процессов параллельной программы: например, часть процессов программы в таком случае бы производила расчеты с использованием графических ускорителей на одном кластере, другая часть процессов получила бы возможность использовать библиотеки численных методов, имеющиеся на втором кластере. В-пятых, наличие средств для подключения дополнительных компьютеров позволило бы планировать постепенное наращивание мощностей распределенной системы.

2.3 Три технологических прорыва

Интерес к совместному использованию вычислительных ресурсов, неоднородных как в смысле процессоров, так и в смысле связей между процессорами, приводит к 1) развитию вычислительных алгоритмов, допускающих гибкость в организации коммуникаций между процессами и даже частичную потерю сообщений [12,13], 2) развитию методов организации вычислений, позволяющих выполнять коммуникации на фоне счета и динамически перераспределять вычисления с целью оптимизации загрузки вычислительных узлов и сетей связи [14-16]. В то же время, 3) характеристики (пропускная способность, латентность) каналов, связывающих кластеры, становится сопоставимыми с характеристиками сетей связи между узлами кластеров.

Эти три технологических прорыва открывают перспективу для применения распределенных вычислительных систем к решению крупных задач с более интенсивными коммуникациями.

В связи с этим актуальной является проблема разработки средств, которые позволили бы распределять процессы параллельных задач между вычислительными узлами нескольких кластеров и обеспечивать коммуникации между распределенными процессами.

2.4 Цель проекта NumGRID и требования к объединению вычислительных кластеров в проекте NumGRID

Цель проекта NumGRID заключается в разработке программного комплекса для поддержки исполнения параллельных программ на объединении вычислительных кластеров с распределением процессов параллельных программ между узлами вычислительных кластеров. В отличие от таких систем, как Globus Toolkit [17], ставящих задачу объединения вычислительных ресурсов в общем, проект NumGRID сконцентрирован на разработке простого инструмента пользовательского уровня для распределения процессов параллельлной программы между кластерами, к которым пользователь имеет непосредственный доступ посредством личных учетных записей. Принципиальным является обеспечение единой коммуникационной среды для распределенных процессов на основе стандартов MPI. Выбор стандарта MPI в качестве коммуникационного протокола обосновывается доминирующим положением MPI среди средств разработки параллельных программ численного моделирования.

К разработке программного комплекса в проекте NumGRID предъявляются требования, обоснованные сложившейся практикой организации вычислений и целью проекта:

1. Общая коммуникационная среда для процессов распределенных по нескольким кластерам должны быть рализована на основе стандартов MPI.
2. О структуре кластеров и сети связи между ними нужно предполагать следующее: каждый кластер состоит по крайней мере из головного/управляющего узла и вычислительных узлов; при этом вычислительные узлы предназначены для запуска на них вычислительных процессов и связаны между собой высокоскоростной сетью, а с головным узлом – сетью, как правило, меньшей пропускной способности, поддерживающей протокол TCP; головной

узел имеет еще по крайней мере один сетевой интерфейс, поддерживающий протокол TCP, через который узел может общаться с головными узлами других кластеров; головной узел используется для компиляции задач, управления локальными очередями задач и мониторинга задач.
3. Процессы параллельной программы должны размещаться на вычислительных узлах кластеров, узлы не имеют прямого сообщения друг с другом.
4. Должен быть предоставлен удобный интерфейс для задания конфигурации объединения кластеров, управления ресурсами и задачами.
5. Должны быть созданы условия для обеспечения динамических свойств прикладных программ (динамическая настраиваемость на доступные ресурсы, динамическая балансировка нагрузки, системы мониторинга и т.д.).
6. Должна обеспечиваться безопасность вычислений.
7. Каждому пользователю для запуска распределенных зирадач на нескольких кластерах должно быть достаточно иметь учетные записи на этих кластерах и пакет NumGRID. Организация NumGRID не должна требовать изменений в практике и политиках администрати-вания кластеров.
8. Кластеры могут быть разнородными в аппаратном, системном программном обеспечениии, линии связи могут быть разной пропускной способности, кластеры могут иметь различную административную подчиненность.

2.5 Обзор родственных проектов

Известно несколько проектов [18-22], которые ставят своей задачей объединение вычислительных систем на основе стандартов MPI. Системы, разрабатываемые в рамках этих проектов, не удовлетворяют полностью списку требований к NumGRID, однако многие проблемы стоящие перед NumGRID также решались в этих проектах. Далее рассмотрены наиболее значимые проблемы и подходы, которые применяются для их решения.

2.5.1 Обеспечение эффективности коммуникаций внутри кластеров

Как правило, в современных кластерах используется несколько типов сетей для объединения узлов. Сети с относительно малой пропускной способностью и относительно большой задержкой (латентностью) используются для задач управления узлами кластера и организации распределенных файловых систем. Для обмена данными между процессами параллельных вычислительных приложений используются скоростные сети с высокой пропускной способностью и малыми латентностями. При объединении кластеров важно, чтобы коммуникации внутри кластеров оставались эффективными. В то время как между кластерами без существенного изменения системного ПО возможно устанавливать соединение только по протоколам TCP/IP, то в зависимости от технологии скоростной сети внутри кластеров могут применяться различные реализации MPI, способные использовать оборудование более эффективно, чем на высоком уровне TCP/IP. Обычно производитель вычислительных систем поставляет такие реализации вместе с аппаратурой, если применяются специальные сетевые аппаратные решения, либо администраторы и пользователи систем могут выбрать свободно или коммерчески доступные библиотеки (MPICH-GM, OpenMPI, Intel MPI, Voltaire MPI, HP-MPI) для распространенных коммуникационных технологий (Myrinet, Infiniband).

Необходимость использования специализированных реализаций MPI для внутрикластерных коммуникаций обусловила отказ от коммуникационной библиотеки Nexus [23] в проекте MPICH-G [24] и разработку новой версии системы MPICH-G2 [19].

2.5.2 Обеспечение коммуникаций между внутренними узлами различных кластеров

Ранние подходы к объединению вычислительных систем на основе MPI [24, 19], предполагали, что процессы, располагающиеся на одной вычислительной системе, могут установить

прямое соединение в смысле TCP с процессами, расположенными на другой системе. Для современных кластерных систем установление таких прямых соединений невозможно (см. требования в разделе 2.4, п.2), поэтому необходимо организовывать трансляцию сообщений с внутренних узлов кластера через узел, имеющий сетевые интерфейсы как внутрь кластера, так и в сеть связи между кластерами. Далее такой узел будем называть узлом трансляции. Трансляция может быть осуществлена неявно с помощью технологий NAT [25] или VPN. Эти технологии обеспечивают возможность установления соединения TCP между процессами расположенными на компьютерах в различных частных сетях. Таким образом, для процессов создается иллюзия нахождения в одной сети и могут быть применены средства, например MPICH-G2 [19] для организации межкластерных взаимодействий. Однако, такая организация вычислений не будет учитывать фактическую топологию сети связи, соответственно организация межкластерных коммуникаций будет неэффективной, особенно для операций коллективного взаимодействия процессов (см. раздел 2.5.4 о коллективных коммуникациях). В проекте PACX-MPI [20] предполагается, что узел трансляции имеет доступ как в высокоскоростную сеть внутри кластера, так и в сеть между кластерами. Для организации передачи сообщений между кластерами на узле трансляции запускается дополнительный процесс MPI, помимо процессов, необходимых для выполнения прикладной логики параллельной программы. Через этот процесс все остальные процессы внутри данного кластера общаются с внешним миром. Преимущества такого подхода – PACX-MPI осведомлен о топологии сети и на этой основе может оптимизировать межкластерные взаимодействия, эффективно используется высокоскоростная сеть. Недостатки – 1) часто в современных кластерах узлы, имеющие доступ во внешние сети, не имеют подключения к скоростной сети; в таких системах требуется реализация трансляции сообщений через внутреннюю медленную TCP/IP сеть; 2) для трансляции требуются дополнительные процессы MPI, не соответствующие логике приложения.

Поскольку идеология грид предполагает сохранение локальных политик администрирования отдельных вычислительных систем при включении их в метакомпьютер, то все порты TCP, кроме, обычно, порта для доступа по протоколу SSH закрыты с помощью сетевого экрана (файервола). Поэтому, для передачи сообщений между кластерами применяются технологии проксирования и туннелирования через доступные порты [26].

В NumGRID предполагается использование ограниченного числа доверенных кластеров с выделенными линиями связи между ними, поэтому необходимые порты для связи между кластерами могут быть открыты. В случае если NumGRID нужно использовать в общественных сетях, возможно применения туннелирования соединений через SSH.

2.5.3 Учет особенностей сетевой организации для реализации эффективных межкластерных коммуникаций

Учет топологии сети рассматривается на уровне реализации взаимодействия между двумя процессами (p2p-взаимодействия) и на уровне реализации коллективных операций (см. раздел 2.5.2). Передача p2p может быть оптимизирована, когда существует более одного физического пути между коммуницирующими узлами. В таком случае применяется разбиение сообщения на части и параллельная доставка частей по разным путям [27].

В случае, когда для доставки сообщения требуется использование узлов трансляции, необходимо контролировать использование памяти на узлах трансляции, и кроме того, нельзя допускать блокирование узла трансляции в процессе доставки сообщения большого размера: нужно обеспечивать возможность прохождения сообщений малого размера на фоне доставки большого. Контроль памяти возможен только посредством разбиения сообщения на части. Избежать блокирования можно, если поддерживать многопоточную обработку сообщений на узле трансляции, либо посредством разбиения сообщений на части. Таким образом, для решения рассмотренных задач целесообразно применение пакетирования сообщений [28].

Применение классификации сообщений по приоритетам [29] целесообразно для обеспечения приоритетной доставки сообщений малого размера, имеющих смысл для управления ходом вычислений.

Учет топологии сети в реализации коллективных операций рассмотрен в следующем параграфе.

2.5.4 Эффективная реализация коллективных коммуникаций в неоднородной коммуникационной среде

Большинство упомянутых проектов занималось вопросом оптимизации коллективных коммуникаций в неоднородных сетях.

В работе [30] показано, что если время завершения посылки сообщения близко ко времени завершения приема сообщения, то оптимальной схемой рассылки сообщений для реализации широковещательной рассылки будет бинарное дерево. Если же время завершения посылки существенно больше времени завершения отправки, то оптимальная схема рассылки: корневой узел самостоятельно посылает получателям сообщение непосредственно. Таким образом, в случае объединения кластеров, оптимальной схемой коллективных коммуникаций будет бинарное дерево внутри кластеров, и посылка каждому кластеру по отдельности между кластерами [31]. Дополнительная оптимизация возможна, если кластеры не связаны физически в полный граф. В таком случае между кластерами не имеющими непосредственной связи сообщения могут двигаться через промежуточные кластеры. Поэтому, если требуется широковещательная рассылка, то корневой процесс должен послать лишь одно сообщение в по каждому физическому соединению, а промежуточные кластеры должны распространить это сообщение далее самостоятельно. Такой образом удается избежать многократной передачи одиних и тех же данных по одним и тем же отрезкам сети.

2.5.5 Оптимизация приложений с учетом неоднородностей распределенной вычислительной среды

Несмотря на любые оптимизации алгоритмов межкластерных коммуникаций, неустранима проблема относительно низкой пропускной способности и высоких задержек в сети между кластерами. В таких условиях приложения, которые по своей природе требуют частых коммуникаций и в большом объеме между всем процессами приложения, не могут эффективно выполняться в распределенной среде. Для того, чтобы приложение могло выполняться эффективно, необходимо, чтобы имелась возможность равномерно распределить вычислительную загрузку между всеми узлами вычислительной системы и передавать сообщения на фоне счета: то есть сообщения должны идти по сети тогда, когда процессоры в это время заняты другой работой. Для минимизации простоев процессоров может потребоваться реализация динамического перераспределения работы между процессорами.

В проектах [32-34] реализовано автоматическое перераспределение загрузки на уровне процессов MPI: между узлами вычислительной системы с передаются именно процессы MPI. Такой подход упрощает разработку приложений, однако не учитывает структуру взаимодействий процессов прикладной программы.

2.5.6 Авторизация и шифрование сообщений

В работе [35] даются основные требования к обеспечению безопасности распределенных вычислений: 1. должно быть установлено взаимное доверие между тем, кто порождает процесс и ресурсом, где он порождается; при этом порожденные процессы должны наследовать способность к аутентификации от породивших процессов; 2. необходимо обеспечить шифрование данных на линиях передачи, где возможен перехват сообщений. Эти требования для распределенных приложений MPI были обеспечены реализацией MPICH-G [24] на основе библиотеки Nexus [23].

В отличие от MPICH-G, где предусматривался запуск процессов MPI на произвольных ресурсах, обнаруженных с помощью инструментария Globus Toolkit [16], пользователь NumGRID самостоятельно выбирает кластеры, которые он хочет объединить. Для объединения кластеров пользователь должен обладать учетными записями на каждом из них. Такая постановка облегчает проблему авторизации при порождении процессов. Авторизация может потребоваться для выполнения команд инструментария, обеспечивающих запуск процессов, например, при использовании очередей задач на выбранном вычислительном кластере. Использование выделенных сетевых линий между вычислительными кластерами позволяет на текущем этапе развития

системы NumGRID отказаться от реализации шифрования передаваемых данных. В случае необходимости использования публичных линий связи могут использоваться внешние средства для организации шифрованных туннелей [36, 37].

2.5.7 Устойчивость к сбоям

В проекте LA-MPI [27] надежность достигалась за счет применения проверки контрольных сумм и повторной пересылки данных. Признано, что использование протокола TCP для обеспечения надежности доставки внутри вычислительных систем нецелесообразно из-за больших накладных расходов. Проект [38] предоставляет возможности восстановления прерванных MPI-процессов даже после сбоя N-1 из N процессов. FT-MPI предоставляет прикладной программе сведения о состоянии процесса: завершён аварийно, прерван, восстановлен, перезапущен сначала. Сведения извлекаются с помощью обработчиков ошибок. Такой подход требует использования техник восстановления после сбоев внутри самой прикладной программы. Поскольку NumGRID в настоящее время ориентирован на объединение сравнительно небольших кластеров, полагается, что стабильность вычислительных процессов обеспечивается уровнем защищенности аппаратных компонентов вычислительной системы, на которой эти процессы запущены.

3. Программный комплекс NumGRID

Программный комплекс NumGRID состоит из трех компонентов:

- Шлюз
- Библиотека NumGRID-MPI
- Графическая система управления NumGRID

С помощью графической системы управления NumGRID пользователь задает набор кластеров, которые он хочет объединиеть для решения задачи, задает параметры авторизации на каждом из кластеров (пользователь должен иметь личные учетные записи на кластерах), параметры подзадач: сколько процессов на каждом кластере должно быть запущено и как распределены, путь на своей рабочей машине до архива с файлами своей программы и входными данными.

По команде запуска задачи, система управления осуществляет авторизацию на кластерах, отправляет файлы пользователя и системные файлы NumGRID на кластеры, выполняет сборку программы пользователя, а также шлюза и библиотеки NumGRID-MPI. Для обеспечения универсальности системы управления, сборка программы пользователя осуществляется на основе предоставленных пользователем инструкций (в виде Makefile). На головных узлах кластеров запускаются шлюзы и устанавливают между собой каналы связи в соответствии с топологией, заданной пользователем. Затем, на каждом кластере, в соответствии с параметрами подзадач, запускается или ставится в очередь программа пользователя. Сборка программы пользователя с библиотекой NumGRID-MPI вместо обычных библиотек MPI наделяет программу пользователя способностью ориентироваться в объединении кластеров. Поэтому запущенная программа пользователя автоматически в начале своей работы (в функции MPI_Init) осуществляет подключение к шлюзу, работающему на головном узле данного кластера. Шлюзы распространяют информацию о подключении подзадач между собой. Когда все подзадачи подключились к своим шлюзам, шлюзы информируют подзадачи об общей конфигурации системы и таким образом подзадачи осуществляют возврат из функции MPI_Init() с полной информацией о распределенной системе и проинициализированным коммуникатором MPI_COMM_WORLD, объединяющим все распределенные процессы.

Пользователь имеет возможность наблюдать за состоянием исполняющихся задач посредством графической системы управления NumGRID, завершать задачи, забирать результаты вычислений.

Все обращения к функциям MPI анализируются библиотекой NumGRID-MPI. Если требуемые коммуникации включают лишь процессы, расположенные в рамках одного кластера, то вызывается предустановленная на кластере библиотека MPI, способная эффективно использо-

вать высокоскоростную сеть кластера. Если требуются коммуникации между процессами, расположенными на различных кластерах, то сообщения транслируются через шлюзы.

Применяется метод коммутации пакетов при трансляции сообщений через шлюзы. Это позволяет контролировать расход памяти на головных узлах, параллельно отсылать несколько сообщений, использовать несколько путей доставки сообщения для повышения производительности сети. Реализовано расширение функций p2p станадрта MPI функциями с заданием приоритета сообщений (см. 2.5.3). Учитывается топология сети при реализации коллективных операций (см. 2.5.4).

4. Экспериментальное исследование NumGRID

Эксперименты по запуску приложений проводились на объединении кластеров Сибирского суперкомпьютерного центра (ССКЦ) [8] и Новосибирского государственного университета (НГУ). Кластеры построены на основе двухпроцессорных узлов с процессорами Intel Xeon E5540 (4 ядра) и внутренней коммуникационной сетью InfiniBand. Пропускная способность канала между головными узлами кластеров – 10 Гб/с.

4.1 Решение волнового уравнения

Ниже представлены графики, демонстрирующие характеристики исполнения программы решения волнового уравнения с помощью двухслойной явной схемы. На всех диаграммах запись «P (NxS)» означает «всего P ядер, из них N на кластере НГУ и S – на кластере ССКЦ.

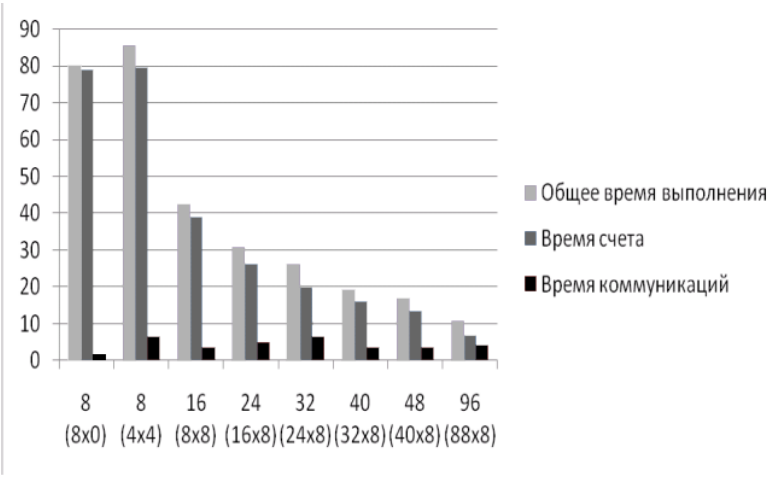

Рис. 2. Время решения волнового уравнения явным методом на NumGRID, сек.

На Рис.2 видно, что при переходе к распределенным вычислениям (с 8x0 на 4x4) увеличивается общее время работы программы за счет увеличения расходов на коммуникации. Расходы на коммуникации в данном примере увеличиваются в 5 раз, что приводит к росту времени работы программы на ~6%.

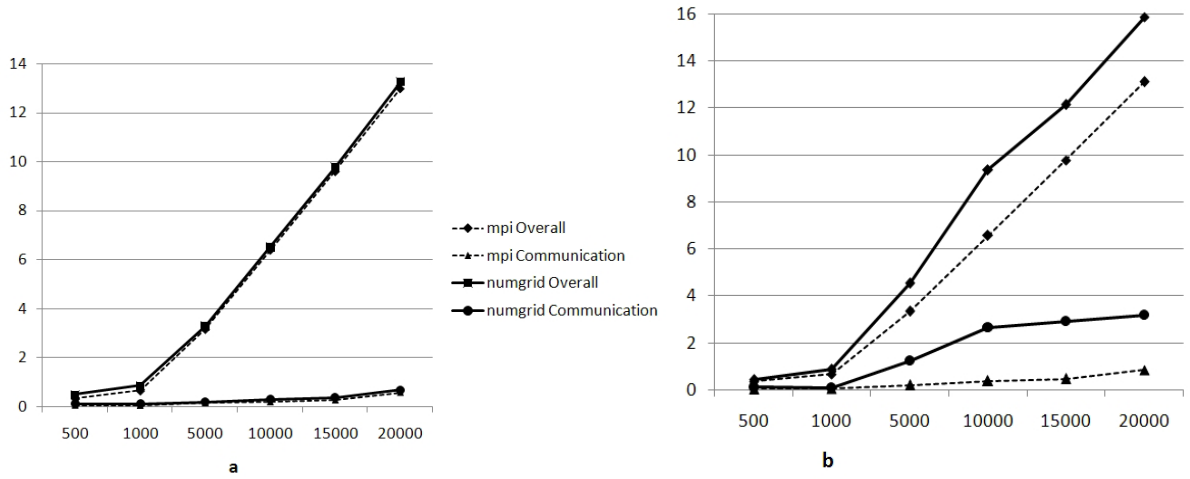

Рис. 3. Зависимость времени исполнения программы от увеличения объема коммуникаций на примере решения волнового уравнения для разных размеров задачи и способов распределения вычислений, время в сек.

На Рис. 3 показано, как изменяется время решения двумерного волнового уравнения на двух процессах в зависимости от размера задачи и способа распределения вычислений. Обозначения: mpiOverall – время работы программы, процессы которой расположены в рамках одного кластера; mpiCommunication – время, потраченное программой на коммуникации; соответствующие графики numgrid показывают время для процессов, расположенных на разных кластерах.

Распределение вычислений выполняется методом декомпозиции области. Таким образом, каждый процесс обрабатывает половину области. На части a Диаграммы показана ситуация, в которой область моделирования увеличивается (горизонтальная ось) по разрезанной размерности. При этом размер границы разреза, очевидно, не изменяется и объем коммуникаций между процессами остается постоянным. На части b Диаграммы показано, что происходит, когда изменяется размер области по другой размерности. В этом случае, с изменением размера области соответственно увеличивается длина разреза и объем коммуникаций. По сравнению с ситуацией a, видно, что в ситуации b время коммуникаций между процессами, расположенными на разных кластерах, существенно увеличивается в отношении времени коммуникаций между процессами внутри кластера.

4.2. Генерация случайных чисел и расчет статистик

Ниже представлены результаты тестирования программы, которая параллельно генерирует случайные величины и вычисляет статистики. Программа используется для моделирования физических процессов методами Монте-Карло. Программа осуществляет редкие коллективные коммуникации для сбора статистик, в остальное время процессоры выполняют существенные по объему вычисления независимо. Это позволяет ожидать, что относительно плохая пропускная способность сети между кластерами незначительно скажется на общей производительности программы.

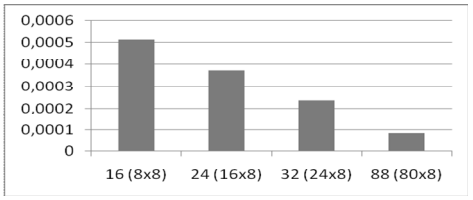

Рис. 4. Время генерации 2000 значений случайной величины и расчета статистик на NumGRID, сек. При этом время генерации 2000 значений случайной величины и расчета статистик на одном ядре кластера 1: 0,0109 сек.

Время исполнения программы при переходе от решения на одном ядре к решению на 16 ядрах, по 8 ядер на каждом кластере, уменьшается в 21 раз. Объяснить подобное сверхлинейное ускорение можно более эффективным использованием кэшей процессоров при увеличении количества процессоров и низкими коммуникционными расходами, свойственными данной задаче. При дальнейшем увеличении количества вовлеченных ресурсов так же отмечается сверхлинейное ускорение. Рис. 5 дает более ясное представление об ускорении времени решения при увеличении количества процессоров.

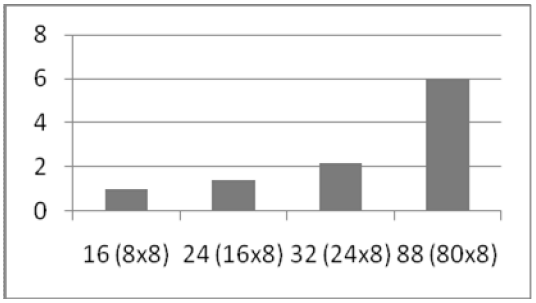

Рис. 5. Ускорение генерации 2000 значений случайной величины и расчета статистик на NumGRID относительно времени работы программы на двух узлах: один узел кластера НГУ (8 ядер) и один узел кластера ССКЦ (8 ядер).

4.3 Анализ экспериментов

В ходе экспериментов показано, что в зависимости от способа распределения вычислений между кластерами меняется эффективность работы приложений. Показано, что использование дополнительных процессоров из другого кластера позволяет уменьшать время выполнения приложения при определенных конфигурациях запуска. В частности, на задаче решения волнового уравнения достигается эффективность 90% на 40 процессорных ядрах относительно производительности программы на 8 ядрах, и 65% на 96 ядрах. На задаче генерации случайных чисел достигается эффективность 92% на 24 ядрах и 109% на 88 ядрах.

Несмотря на существенную разницу в производительности систем коммуникации внутри кластеров и между кластерами, время решения задач увеличивается умеренно для решения волнового уравнения на том же количестве процессоров и уменьшается для решения задачи поиска статистик. Можно заключить, что существует класс задач, которые могут быть решены на объединении кластеров за приемлемое для пользователей время. Вместе с тем, объединение кластеров может позволить решать задачи, большего объема, чем был бы способен решить один кластер за то же время.

5. Заключение

Растущий интерес к объединению разнородных вычислительных ресурсов для решения крупномасштабных задач и повышения эффективности использования вычислительных систем стимулирует развитие численных алгоритмов, методов планирования распределенной обработки данных, инструментов взаимодействия программных систем и средств взаимодействия пользователя с объединенными вычислительными системами.

В рамках проекта NumGRID проведен анализ проблем объединения вычислительных кластеров, существующих подходов к решению этой задачи, сформулированы требования к программному комплексу для организации распределенных вычислений на объединении вычислительных кластеров, программный комплекс реализован. Проведены испытания по объединению кластеров Сибирского суперкомпьютерного центра и Новосибирского государственного университета. Эксперименты демонстрируют, что существуют классы задач численного моделирования, которые целесообразно решать на объедниении вычислительных кластеров.

Дальнейшая работа заключается в расширении поддержки стандарта MPI-2.2, адаптации существующих приложений для работы в среде NumGRID, создании методов и средств разработки программ численного моделирования для неоднородных вычислительных сред.

Литература

1. D.Fougere, M.Gorodnichev, N.Malyshkin, V.Malyshkin, A. Merkulov, B.Roux. NumGrid Middleware: MPI Support for Computational Grids // Parallel Computing Technologies: 8th International Conference, PaCT 2005, Krasnoyarsk, Russia, September 5-9, 2005. Proceedings, Springer, 2005. - LNCS Vol. 3606, pp. 313-320.
2. MPI Documents. – URL: http://www.mpi-forum.org/docs/docs.html. Дата обращения: 15.12.2011.
3. Top 500 Supercomputer Sites. – URL: http://www.top500.org. Дата обращения: 15.12.2011.
4. XSEDE – Extreme Science and Engineering Discovery Environment. – URL: https://www.xsede.org. Дата обращения: 15.12.2011.
5. Worldwide LHC Computing Grid. – URL: http://lcg.web.cern.ch/lcg. Дата обращения: 15.12.2011.
6. Grid5000:Home. – URL: https://www.grid5000.fr. Дата обращения: 15.12.2011.
7. Межведомственный Суперкомпьютерный Центр РАН. – URL: http://www.jscc.ru. Дата обращения: 15.11.2011.
8. Сибирский Суперкомпьютерный Центр. – URL: http://www2.sscc.ru. Дата обращения: 15.11.2011.
9. BOINC. Open-source software for volunteer computing and grid computing. – URL: http://boinc.berkeley.edu/index.php. Дата обращения: 15.12.2011.
10. World Community Grid. Technology solving problems. – URL: http://www.worldcommunitygrid.org. Дата обращения: 15.12.2011.
11. Воеводин Вл.В., Жолудев Ю.А., Соболев С.И., Стефанов К.С. Эволюция системы метакомпьютинга X-Com // Вестник Нижегородского государственного университета им. Н.И. Лобачевского. №4. 2009. С 157-164.
12. F. Oboril, M. B. Tahoori, V. Heuveline, D. Lukarski, J.-Ph. Weiss. Fault Tolerance Technique for Iterative Solvers / Karlsruhe Institute of Technology – URL: http://www.emcl.kit.edu/preprints/emcl-preprint-2011-10.pdf. Дата обращения: 15.12.2011.
13. Peng Du, Piotr Luszczek, Jack Dongarra: High Performance Dense Linear System Solver with Soft Error Resilience // In. proc. of International Conference on Cluster Computing (CLUSTER), Austin, TX, USA, September 26-30, pp. 272-280.
14. Malyshkin V.E., Perepelkin V.A. LuNA Fragmented Programming System, Main Functions and Peculiarities of Run-Time Subsystem - In: Proceedings of the 11th Conference on Parallel Computing Technologis, LNCS 6873 - pp. 53-61, Springer, 2011
15. Система параллельного программирования OpenTS. Sergey Abramov, Alexei Adamovich, Alexander Inyukhin, Alexander Moskovsky, Vladimir Roganov, Elena Shevchuk, Yuri Shevchuk, and Alexander Vodomerov. 2005. OpenTS: An Outline of Dynamic Parallelization Approach // In proc of. PaCT 2005 conference, Krasnoyarsk, Russia, September 5-9, 2005, Springer, 2005. - LNCS Vol. 3606, pp. 303-312.
16. Globus Toolkit Homepage. – URL: http://www.globus.org/toolkit. Дата обращения: 15.11.2011.
17. C. Grelck and F. Penczek, Implementation Architecture and Multithreaded Runtime System of S-Net, in Implementation and Application of Functional Languages, 20th International Symposium, IFL'08, Hatfield, United Kingdom, Revised Selected Papers, 2010.
18. William L. George, John G. Hagedorn, and Judith E. Devaney. IMPI: Making MPI Interoperable //Journal of Research of the National Institute of Standards and Technology, vol. 105, 2000, pp. 343—428.
19. Nicholas T. Karonis, Brian Toonen, and Ian Foster. MPICH-G2: a Grid-enabled implementation of the Message Passing Interface // Journal of Parallel and Distributed Computing - Special issue on computational grids, Volume 63 Issue 5, May 2003, Academic Press, Inc. Orlando, FL, USA.
20. Edgar Gabriel, Michael Resch, and Roland Ruehle. Implementing MPI with Optimized Algorithms for Metacomputing // Proc. Message Passing Interface Developer's and User's Conference (MPIDC'99), pp. 31-41, Atlanta, GA, March 10-12, 1999.
21. Thilo Kielmann, Rutger F. H. Hofman, Henri E. Bal, Aske Plaat, and Raoul A. F. Bhoedjang. MagPIe: MPI's Collective Communication Operations for Clustered Wide Area Systems. Pro-

ceedings of the seventh ACM SIGPLAN symposium on Principles and practice of parallel programming ACM New York, NY, USA,1999.
22. IMPI Relay Trunking for Improving the Communication Performance on Private IP Clusters. R.Takano, M.Matsuda, T.Kudoh, Y.Kodama, F.Okazaki, Y.Ishikawa, and Y.Yoshizawa. In CCGrid2008, 2008.
23. I. Foster, C. Kesselman, and S. Tuecke. The Nexus Approach to Integrating Multithreading and Communication //Journal of Parallel and Distributed Computing, vol. 37, 1996, pp. 70-82.
24. I. Foster, J. Geisler, W. Gropp, N. Karonis, E. Lusk, G. Thiruvathukal, S. Tuecke. A wide-area implementation of the Message Passing Interface // Paral Comp (1998) Volume: 24, Issue: 12, Publisher: Elsevier, pp. 1735-1749.
25. P. Srisuresh, K. Egevang. Traditional IP Network Address Translator (Traditional NAT). Network Working Group, Request for Comments: 3022. – URL: http://tools.ietf.org/html/rfc3022. Дата обращения: 15.12.2011.
26. Yoshio Tanaka , Mitsuhisa Sato, Motonori Hirano, Hidemoto Nakada, and Satoshi Sekiguchi. Performance Evaluation of a Firewall-Compliant Globus-Based Wide-Area Cluster System // Proceedings of the 9th IEEE International Symposium on High Performance Distributed Computing IEEE Computer Society Washington, DC, USA, 2000.
27. Rob T. Aulwes, David J. Daniel, Nehal N. Desai, Richard L. Graham, L. Dean Risinger, Mark A. Taylor, Timothy S. Woodall, Mitchel W. Sukalski, "Architecture of LA-MPI, A Network-Fault-Tolerant MPI" // 18th International Parallel and Distributed Processing Symposium (IPDPS'04) – Papers, vol. 1, p.15, 2004.
28. Paul Baran, "On Distributed Communications Networks," IEEE Transactions on Communication Systems, Vol CS-12 (1), pp. 1-9, Mar 1964.
29. Real-Time Message Passing Specification based on MPI. – URL: http://www.mpirt.org. Дата обращения: 15.12.2011.
30. M. Bernaschi and G. Iannello. Collective Communication Operations: Experimental Results vs. Theory // Concurrency: Practice and Experience, 10(5), April 1998, pp. 359-386.
31. Thilo Kielmann, Rutger F. H. Hofman, Henri E. Bal, Aske Plaat, and Raoul A. F. Bhoedjang. MagPIe: MPI's Collective Communication Operations for Clustered Wide Area Systems. Proceedings of the seventh ACM SIGPLAN symposium on Principles and practice of parallel programming ACM New York, NY, USA,1999.
32. Laxmikant V. Kale, Eric Bohm, Celso L. Mendes, Terry Wilmarth, and Gengbin Zheng. Programming Petascale Applications with Charm++ and AMPI. Petascale Computing: Algorithms and Applications. Chapman & Hall / CRC Press, USA, 2008, pp. 421-441.
33. Xiaohui Wei, Hongliang Li, Dexiong Li MPICH-G-DM: An Enhanced MPICH-G with Supporting Dynamic Job Migration. Proceedings of the 2009 Fourth ChinaGrid Annual Conference. IEEE Computer Society Washington, DC, USA 2009.
34. A. Barak, O. La'adan and A. Shiloh, Scalable Cluster Computing with MOSIX for Linux . Proc. Linux Expo '99, pp. 95-100, Raleigh, N.C., May 1999.
35. I. Foster, N. T. Karonis, C. Kesselman, G. Koenig and S. Tuecke. A secure communications infrastructure for high-performance distributed computing. HPDC '97 Proceedings of the 6th IEEE International Symposium on High Performance Distributed Computing IEEE Computer Society Washington, DC, USA,1997.
36. D. Farinacci, T. Li, S. Hanks, D. Meyer, P. Traina. Generic Routing Encapsulation (GRE). // Network Working Group, Request for Comments: 2784, March 2000. – URL: http://tools.ietf.org/html/rfc2784. Дата обращения: 15.12.2011.
37. W. Townsley, A. Valencia, A. Rubens, G. Pall, G. Zorn, B. Palter. Layer Two Tunneling Protocol "L2TP". // Network Working Group, Request for Comments: 2661, August 1999. – URL: http://tools.ietf.org/html/rfc2661. Дата обращения: 15.12.2011.
38. Graham E. Fagg and Jack J. Dongarra. FT-MPI: Fault Tolerant MPI, Supporting Dynamic Applications in a Dynamic World // RECENT ADVANCES IN PARALLEL VIRTUAL MACHINE AND MESSAGE PASSING INTERFACE, LNCS, 2000, Volume 1908/2000, pp. 346-353.