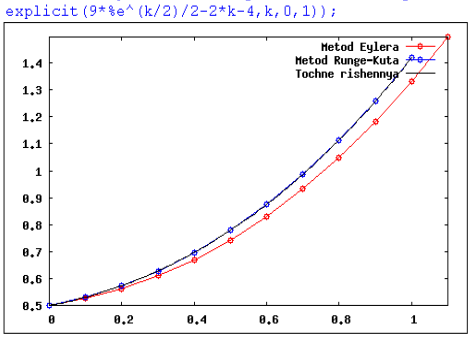
```
(%i22) sol: rk(x+0.5*y,y,0.5,[x,0,1,0.1]);
(%o22) [[0,0.5],[0.1,0.530719921875],[0.2,
0.57326910649445],[0.3,0.6282540530975],[0.4,
0.69631235680268],[0.5,0.77811430292759],[0.6,
0.87436454305121],[0.7,0.98580385700967],[0.8,
1.11321100523152],[0.9,1.257404676044277],[1.0,
1.419245532821431]]
```

Построим все интегральные кривые, полученные разными методами в одной системе координат:

```
(%i28) wxdraw2d(point_type=circle, point_size=1, points_joined=true,
       key="Metod Eylera",color=red,points(Xeyl,Yeyl),point_size=1,
       points_joined=true,key="Metod Runge-Kuta",
       color=blue, points(sol),key="Tochne rishennya", color=black,
       explicit(9*%e^(k/2)/2-2*k-4,k,0,1));
```

(%t28)



Выводы. Таким образом, применение СКМ Maxima существенно ускоряет процесс программной реализации метода, сокращая объем программы и затраты времени на ее написание, делая ее более «прозрачной».

Конструктивное построение курса численных методов позволяет расширить его разделами, программная реализация которых в процедурной методологии вызывает осложнения; ликвидировать повторяемость численных методов в различных разделах курса; привлечь к курсу эффективные аналитические и на половину численные методы, по-новому взглянуть на традиционные методы и расширить границы их применения.

На уровне целей обучения применения системы компьютерной математики Maxima определяет цель изучения численных методов как логического продолжения курсов математики и программирования и необходимой основы курса моделирования, а на уровне организационных форм дает возможность внедрения таких прогрессивных форм обучения, как групповая и индивидуально - дифференцированная.

Список литературы

1. **Ильина В.А., Силаев П.К.** Система аналитических вычислений Maxima для физиков-теоретиков / МГУ им. М.В.Ломоносова, физический факультет, кафедра квантовой теории и ФВЭ. – М.: 2007. – 112с.

2. **Стахин Н.А.** Основы работы с системой аналитических (символьных) вычислений Maxima: Учебное пособие. – М.: Федеральное агентство по образованию, 2008. – 86с.

3. **Губина Т.Н., Андронова Е.В.** Решение дифференциальных уравнений в системе компьютерной математики Maxima: учебное пособие. – Елец: ЕГУ им. И.А.Бунина, 2009. – 99с.

Рукопись поступила в редакцию 12.12.11

УДК 004.75

А.А. АЗАРЯН[·], д-р техн. наук, проф., Ю.В. ЧМЫХАЛО, М.С. КУКУШКИН, студенты
ГВУЗ «Криворожский национальный университет»

РАСПРЕДЕЛЕНИЕ АТОМАРНЫХ ЕДИНИЦ В ВЫЧИСЛИТЕЛЬНЫХ УЗЛАХ GRID-СИСТЕМЫ

Рассмотрен пример реализации функций системы Grid-технологии для распределенных вычислений и обработки данных.

Актуальность проблемы. В последние годы быстрое развитие получили технологии организации распределенной обработки информации и высокопроизводительных вычислений. Одним из классов таких технологий являются grid-технологии - инфраструктурные технологии

промежуточного слоя, предоставляющие возможность интеграции вычислительных и информационных ресурсов глобальных сетей для решения сверхсложных и ресурсоемких задач вычислительного характера и/или обработки информации. Grid-инфраструктуры являются разновидностью распределенных параллельных систем, определяемых наборами открытых стандартов и протоколов, и служащих для обеспечения доступа к данным, вычислительным мощностям, средам хранения и широкому набору других ресурсов, доступных при помощи Интернета.Одной из актуальных задач в настоящее время является эффективноеуправление вычислительными ресурсами в распределенной среде. Распределение должно выполняться таким образом, чтобы удовлетворить все требования пользователей, а также оптимизировать общее время выполнения и стоимость используемых ресурсов.Цель состоит в определении эффективности такого распределения задач при использовании различных критериев. В статье рассмотрена иерархическая система распределения задач, которая имеет два уровня: распределение вычислительной задачи на фрагменты и распределение фрагментов на вычислительные блоки (атомарные единицы).

Анализ публикаций. В настоящее время опубликовано много научно – технического материала, который дает общий анализ Grid–систем, так как в последнее время к вычислительным кластерам проявляется повышенный интерес со стороны науки, образования и промышленности. В данных научных трудах рассмотрены такие проблемы, как эволюция архитектуры grid-компьютинга и модели адаптации кластерных систем, перспективы развития распределенных вычислительных сетей, а также многое другое. Критический анализ научных трудов показал, что некоторые вопросы данной темы недостаточно раскрыты, в частности, модульной, открытой к расширению системы выявлено не было, как и универсального алгоритма распределения вычислительных ресурсов.

Постановка задачи.В данной статье представлена разработка универсального иерархического алгоритма распределения вычислительных ресурсов в grid-системе, основанного на использовании алгоритма Дейкстры и алгоритма Флойда для нахождения кратчайших путей в топологиях типа клиент-сервер и клиент - клиент, с учетом уникальных характеристик каждого клиента.

В силу технической сложности интеграции компьютеров для решения общих задач, не теряет актуальности проблема эффективности использования ресурсов GRID, что означает минимизацию простаивания определенных ресурсов, в то время как другие загружены работой.

При рассмотрении «сервисно-ориентированной архитектуры», следует определить ключевые термины [1]:

Сервис (служба) - программный компонент, к которому можно удаленно обратиться посредством компьютерной сети, и предоставляющая некоторые функциональные возможности запрашивающей стороне.

Сервисно-ориентированная архитектура (service-oriented architecture, SOA) является основой построения надежных распределенных систем, которые в качестве услуг предоставляют функциональные возможности, с дополнительным акцентом на слабые связи между взаимодействующими сервисами, рис.1.

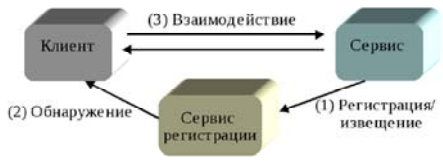


Рис. 1. Цикл взаимодействия сервисов

Этот рисунок иллюстрирует простой цикл взаимодействия сервисов, который начинается с того, что данный сервис оповещает о своем существовании и свойствах посредством известного сервиса регистрации (1), свойства и способы взаимодействия с которым должны быть заранее известны клиентам. Потенциальный клиент, который может быть другим сервисом (или человеком), делает запрос в сервис регистрации (2), чтобы найти сервис, который удовлетворяет его потребностям. Регистрационный сервис возвращает (возможно, пустой) список подходящих сервисов; клиент выбирает один из них и передает ему запрос, используя любой взаимно распознаваемый протокол (3). На рисунке показан самый простой случай. Представленный иллюстративный пример соответствует простому синхронному, двунаправленному способу обмена сообщениями, в то время как в реальной жизни взаимодействие может быть односторонним, или ответ может быть получен не от того сервиса, которому клиент посылал запрос, но от некоторого другого, которому он был передан для завершения обработки.В настоящее время выделяют три основных типа грид-систем [1]:

1. Добровольные гриды – на основе использования добровольно представляемого свободного ресурса персональных компьютеров.

2. Научные гриды — хорошо распараллеливаемые приложения программируются специальным образом (например, с использованием GlobusToolkit);

3. Гриды на основе выделения вычислительных ресурсов по требованию (коммерческий грид, англ. enterprisegrid) - обычные коммерческие приложения работают на виртуальном компьютере, который, в свою очередь, состоит из нескольких физических компьютеров, объединенных с помощью грид-технологий.

Технологические требования, предъявляемые к Grid, определены следующим образом [2]:

1. Гибкиеотношениядоступа (client-server, peer-to-peer).

2. Чёткий высокоуровневый контроль над использованием ресурсов.

3. Многоуровневый контроль прав доступа, локальные и глобальные политики доступа.

4. Поддержка распределения различных ресурсов - программ, данных, устройств, вычислительных мощностей.

5. Поддержка различных моделей пользования - многопользовательской, однопользовательской, режимов performance-sensitive и cost-sensitive.

6. Контроль над качеством предоставляемых услуг, планирование, резервное предоставление услуг.

Для Grid-технологии характерны два (традиционно смешиваемых) направления развития, определяемые их важностью и различиями в исследовании [4]:

Первое направление, связанное с информационными задачами, представляет сегодня сущность Интернета. Это - обмен новостями, приобретение новых знаний, программных средств, мульти-медиа-развлечения, диалоговые системы телеконференции, электронная почта, финансовые операции и т.д. Эти функции "Всемирной Паутины" уже представляются достаточными самым широким слоям населения, покрывая весь компьютерный сервис.

Сделать всю информацию доступной, установить гарантированное время поиска, породить у пользователя ощущение того, что вся информация по его запросу находится рядом, - это фантастическая задача. Можно ли объединить всю информацию мира, блуждающую в Интернете, в одну огромную базу данных, размеры которой даже трудно оценить? Да еще с неограниченным многоканальным доступом?

Движение в направлении осуществления этой глобальной идеи и является задачей исследователей в области Grid-технологий. Ясно, что идеального решения эта задача может никогда не получить. Однако стремиться к минимизации времени выполнения запроса, несмотря на структурную "удаленность" информации, несомненно, следует.

Второе направление развития Grid-технологий связано с вычислительным характером. Интернет должен принимать заказ на работу с задачами высокой сложности, большой размерности, с задачами моделирования сложных физических явлений, таких, например, как точный метеорологический прогноз, с оптимизационными задачами хозяйственного планирования, с задачами статистической обработки экспериментов, с задачами-запросами по контролю космического пространства, с имитационными задачами для испытания новых технических средств и т.д.

Это означает, что в составе Интернета должны быть мощные вычислительные центры, снабженные развиваемыми пакетами прикладных программ решения сложных задач, оболочками для решения классов задач. Такие центры должны комплектоваться высококвалифицированными математиками, развивающими вычислительную базу, готовыми принимать заказы, консультировать пользователей при "доводке" заказов до требуемого формального представления, так как мечты о полной автоматизации процесса их выполнения еще долго будут неисполнимыми.

Какие модели программирования Грид станут успешными, во многом будут определяться такими аспектами, как мобильность, интероперабельность, адаптивность, а также способность поддерживать обнаружение, безопасность и отказоустойчивость при сохранении производительности. Модели и инструментальные средства программирования в значительной степени зависят от доминирующей парадигмы программирования. Имеется также различие между возможностями общей инфраструктуры и возможностями моделей программирования и инструментальных средств, построенных на основе данных инфраструктур.

Грид-технологии в целом и грид-инфраструктура вступают в пору зрелости – происходит переход от тестовых испытаний и пробного обслуживания пилотных приложений к постоянной

устойчивой работе по обслуживанию самых разнообразных прикладных областей науки и производства. В связи с этим перед разработчиками нового прикладного ПО, перед разработчиками грид-ПО, и перед персоналом, обеспечивающим функционирование грид-инфраструктуры встают новые масштабные задачи.

Для выполнения в GRID характерны программы со слабой связью по данным и управлению между выполняющимися параллельно участками кода. Для подобных программ актуальной является распределение вычислений. Целью распределения является минимизация общего времени выполнения параллельной программы в GRID, то есть перераспределение заданий в ходе работы программы, с учетом накапливаемой информации о дисбалансе. В силу того, что зачастую передача задания с одного процессора на другой требует переноса существенных объемов данных, ставится задача распределения с целью минимизировать число пересылок для достижения требуемого уровня балансировки.

GRID система состоит из координирующих элементов и рабочих узлов. Рабочий узел выполняет функцию вычисления. Координирующие элементы распределяют вычислительные задания между рабочими узлами и управляют передачей данных, необходимых для выполнения заданий.

Отправка заданий осуществляется через клиентский интерфейс. Задание вместе с необходимыми данными передается на рабочий узел (отдельный компьютер или кластер) и выполняется.

Каждому пользователю GRID выдается определенные вычислительные ресурсы и время для расчета на них его заданий. Типичное выполнение пользовательского задания состоит из трех этапов: загрузки данных, расчетов и выгрузки результатов. Для произведения расчетов обычно на каждом из рабочих узлов создается процесс, назовем его рабочим, который последовательно определяет выделенную ему очередь заданий. В процессе расчетов пользователь не изолирован от взаимодействия с данными процессами, имеет определенный доступ к ресурсам, на которых идут вычисления. Распределение заданий следует производить как можно более точным образом, а перераспределение производить лишь в случае неприемлемого для системы дисбаланса, рис. 2.

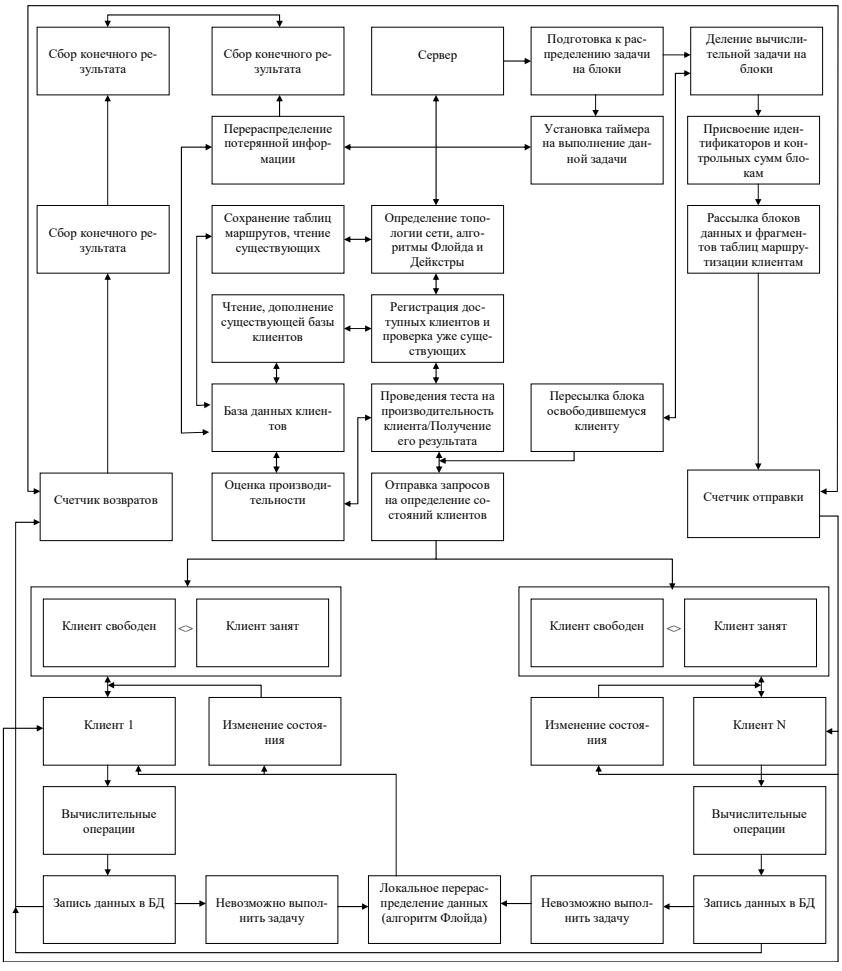


Рис. 2. Функциональная схема grid-систем

Правильно разработанная и хорошо реализованная Grid-среда характеризуется следующими основными функциональными возможностями [3]:

доступ к вычислительным ресурсам, данным, устройствам, измерительным инструментам должен быть простым, прозрачным, удаленным, и безопасным;

доступ должен быть виртуальным (нужен доступ не к серверам, а к сервисам, поставляющим данные или вычислительные ресурсы - причем без необходимости знания аппаратной структуры, обеспечивающей эти сервисы);

доступ должен осуществлять по требованию (с заданным качеством), а ресурсы должны предоставляться тогда, когда в них возникает нужда;

доступ должен быть распределенным, обеспечивая возможность совместной коллективной работы виртуальных команд.

Иерархическая система распределения задач имеет два уровня: распределение задач по кластерам и распределение многопроцессорных задач на кластере. Принцип действия системы включает следующие этапы.

Определение клиентов в сети, сравнение с уже зарегистрированными и дополнение базы.

Определение топологии сети. Определение кратчайшего пути к каждому компьютеру (Алгоритм Дейкстры).

Определение статуса свободен/занят (речь идет о клиенте).

Реализация алгоритма Флойда для маршрутизации данных в группе клиентов.

Оценка производительности узлов.

Подготовка данных к рассылке (расчет балансировки на основе данных, полученных ранее (пункты 1-5), разбиение данных на атомарные единицы, присвоение каждому блоку идентификационного номера и генерация ХЭШ-суммы).

Рассылка данных и таблиц маршрутизации для каждого клиента.

Ожидание ответа о выполнении операций клиентами, выполнение пунктов 1-7 через определенную задержку. Проверка системы на наличие дисбаланса по средствам превышения порогового значения выбывших клиентов.

Следует в grid-системе выделить пять основных блоков, это: сервер, клиент-1, клиент-N, информационная база данных, база данных клиентов. Взаимодействие между этими блоками приведено ниже на функциональной схеме (см. рис. 2).

Сервер сканирует доступных клиентов в сети, после чего сравнивает существующую базу данных клиентов (изначально пустую) с результатами сканирования и выполняет регистрацию новых доступных клиентов. В базе данных клиентов хранится вся информация (такая как: уникальный идентификатор, индекс производительности, отдаленность от сервера, состояние клиента) о всех зарегистрированных в системе компьютерах. После этого, производит сбор информации о новых зарегистрированных клиентах и обновляет информацию об уже существующих в базе данных состояние клиента свободен/занят. Далее, выполняется тестирование производительности клиентов (тактовая частота процессора,объем оперативной памяти и скорость обращения к ней, пропускная скорость интернет канала) и вычисление оценки производительности узла, анализ общей топологии сети методом Дейкстры и построение таблиц маршрутизации для сегментов сети методом Флойда, запись собранной информации в БД клиентов).

Клиент в свою очередь может иметь только два состояния (занят/свободен). Иными словами будет использован генератор случайных чисел, который будет присваивать клиенту ноль или же единицу. Также клиенты смогут благодаря локальным таблицам маршрутизации, в случае невозможности выполнения вычисления поставленной и уже выданной им задачи, перенаправить ее выполнение на ближайший свободный компьютер из их окружения.

После сбора информации о клиентах и ее анализа сервер начинает вычислять оптимальный вариант распределения данных для последующего их деления на атомарные единицы и установки таймера для выполнения одного такта вычислений (таймер необходим для того, чтобы по истечению вычисленного времени сервер мог собрать готовые результаты и определить утерянные). Когда выбран оптимальный вариант, учитывающий все возможности и свойства каждого компьютера, сервер распределяет вычислительную задачу на блоки, затем присваивает каждому блоку его собственный идентификационный номер,действительный внутри данного такта рассылки информации (в следующем такте, если блок будет не выполнен, его нумерация изменится) и контрольную сумму (она необходима для проверки целостности блока клиентом).

Теперь сервер может выполнить рассылку блоков информации и локальных таблиц маршрутизации доступным клиентам. При этом активизируется счетчик отправки, для того, чтобы после сравнения его со счетчиком возвратов и таймером для выполнения действий, сервер мог определить количество утраченных блоков, вычислить выбывших клиентов и корректно перераспределить вычислительные задачи для следующего такта.

Итак, допустим, блоки поступили к клиентам. При этом они меняют свое состояние на «Занят», то есть, единицу, и сверяют контрольные суммы для проверки целостности данных. Локальная таблица маршрутизации, высланная клиенту, сохраняется у него до завершения 1-го такта вычислений, после чего будет заменяться новой, присланной от сервера, если это необходимо. Далее, происходит выполнения расчета присланной вычислительной задачи, после чего реализуется запись результатов в базу данных, при этом, счетчик возвратов увеличивается каждый раз на единицу. Если по каким то причинам клиент не может справиться с задачей, но находится в сети (например, пользователь запустил ресурсоемкое приложение), то он может перенаправить исходные данные, высланные ему, любому ближайшему к нему клиенту, в локальной таблице маршрутизации,если этот клиент уже освободился от выполнения своей задачи, иначе данные в итоге будут считаться утерянными (если клиент не передаст результат вычислений в базу данных до завершения одного такта обработки).

По истечению времени установленного сервером в таймер для выполнения сервер сравнивает счетчик отправки со счетчиком возвратов. Если данные равны, то сервер сохраняет все результаты и переходит к следующему такту, не выполняя предварительного перераспределения вычислительных задач, поскольку все клиенты справились с заданием. Если же счетчики не равны, тосервер сохраняет уже готовые результаты, и анализирует, какие сегменты данных были утрачены, и какие клиенты не справились с задачей. В таком случае он произведет перераспределение вычислительной задачи относительно изменившихся данных и установит новый таймер на выполнение. При этом, блоки которые не были вычислены во время предыдущего такта будут заложены в новый.

На схеме клиенты изображены двумя блоками клиент-1 и клиент-N, это подразумевает, что клиентов может быть сколько угодно много.

Список литературы

1. **Азарян А.А., Сапончук Ю.В., Кукушкин М.С.** Грид-технологии для распределенных вычислений и обработки данных. //Качество минерального сырья: Сб. научн. тр. -Кривой Рог, 2011. -С.80-87.

2. **Демичев А.П.**Введение в Грид-технологии / **А.П. Демичев, В..А. Ильин, А. П. Крюков**. – М. : Дело и Сервис, 2007.

3. **Коваленко В.Н.** Система выполнения массовых запросов на множестве распределённых реляционных баз данных/ **В.Н. Коваленко, Е.И. Коваленко, А.Ю. Куликов** – М, 2007.

4. http://ru.wikipedia.org/wiki/Грид

Рукопись поступила в редакцию 06.03.12

УДК 622.413.3

А. А. ЛАПШИН, Э.В. СЕРЕБРЕНИКОВ[·], кандидаты, техн. наук, доц.,
Д.А. ЛАПШИНА, студентка, ГВУЗ «Криворожский национальный университет»

ТЕОРЕТИЧЕСКОЕ ОБОСНОВАНИЕ ВОЗДУШНО-ВОДОИСПАРИТЕЛЬНОГО ОХЛАЖДЕНИЯ РУДНИЧНОЙ АТМОСФЕРЫ

На основании экспериментального материала проведено математическое моделирование теплообменных процессов при воздушно-водоиспарительном охлаждении рудничной атмосферы, позволяющее с использованием вычислительной техники исследовать температурные режимы с целью оценки санитарно-гигиенических условий работы.

Проблема и ее связь с научными и практическими задачами. Анализ условий ведения подземных горных работ указывает на существование проблемы, связанной с нарушениями тепловых режимов, что может привести к нарушению условий труда, определяемых техникой безопасности. В связи с этим возникла необходимость устранения таких нарушений.

Анализ исследований и публикаций. Одним из возможных путей решения этой проблемы является создание устройств, обеспечивающих охлаждение рудничной атмосферы. Для этого в подземных выработках были проведены эксперименты, которые показали целесообраз-