ISSN 2079-3316 ПРОГРАММНЫЕ СИСТЕМЫ: ТЕОРИЯ И ПРИЛОЖЕНИЯ № 2(16), 2013, с. 43–69

УДК 655.523

М. А. Посыпкин, Н.П. Храпов, В.Н. Филиппов

Метод ветвей и границ на грид-системах персональных компьютеров[1]

Аннотация. В работе рассмотрен вопрос эффективной реализации метода ветвей и границ на грид-системах персональных компьютеров. Приводится описание платформы BOINC, ее особенностей, существенных с точки зрения реализации данного метода. Рассматривается один из возможных подходов к реализации, результаты эксперимента. Делается вывод о необходимых условиях эффективной реализации МВГ на грид-системах персональных компьютеров.

Ключевые слова и фразы: метод ветвей и границ, грид-системы персональных компьютеров, оптимизация.

Введение

Задачи глобальной оптимизации применяются в различных областях современной науки и техники. Эти задачи являются чрезвычайно важными для развития экономики в целом и таких отраслей, как нанотехнологии, микроэлектроника, биология, легкая и тяжелая промышленность. Поэтому разработка эффективных методов решения таких задач и их реализация в виде прикладных программных комплексов представляется актуальной научной и практической проблемой. Отличительной особенностью задач по-

[1] Работа выполнена при поддержке гранта РФФИ № 13-07-00768 А, программы 6П Президиума РАН «Проблемы создания высокопроизводительных, распределенных и облачных систем и технологий», проекта "Математическое моделирование, разработка, исследование и реализация методов решения задач динамической оптимизации большой размерности на современной высокопроизводительной вычислительной технике" номер гос. регистрации проекта - 0115РК00554.

иска глобального экстремума является высокая вычислительная сложность. Поэтому для решения таких задач целесообразно привлекать методы параллельных и распределенных вычислений.

В настоящее время наблюдается стремительное развитие архитектуры ЭВМ. Относительно недавно появились многоядерные процессоры, превратившие персональные компьютеры в параллельные системы с общей памятью. Также быстро развивается сетевая инфраструктура, позволяющая выполнять вычисления в распределенной среде, состоящей из тысяч компьютеров, находящихся в географически удаленных точках.

Всё большую популярность в мире набирают проекты добровольных вычислений, идея которых состоит в том, что владелец персонального компьютера предоставляет неиспользуемые ресурсы своей машины в качестве вычислительной мощности для общественно-значимого проекта. Основная идея грид-систем из персональных компьютеров [1-3] (далее ГСПК) состоит в том, что на вычислительных узлах устанавливается и настраивается клиентское программное обеспечение, которое выполняет периодические запросы удаленному серверу на наличие заданий для своей платформы. Если на центральном сервере таковые задания имеются, то клиентская машина скачивает задание в виде исполняемого файла с необходимыми данными, и запускает его, результат работы приложения возвращается обратно на сервер.

К сожалению, далеко не все распределенные приложения могут эффективно выполняться на подобных системах из-за серьезных ограничений, накладываемых возможностями по передаче данных и высокой вероятностью отказа узлов, участвующих в вычислениях. Вместе с тем, достаточно широкий класс практических задач укладывается в модель управляющий-рабочие, которая является основной моделью приложения в ГСПК. К этому классу относятся многие переборные и комбинаторные задачи, моделирование методом Монте-Карло, задачи идентификации и многие другие. Для таких задач использование ГСПК оправдано и позволяет разгрузить суперкомпьютеры и сервисные гриды. Резюмируя,

можно сказать, что грид-системы персональных компьютеров являются дешёвой альтернативой суперкомпьютерам и сервисным гридам и для ряда задач могут их успешно заменять.

Одной из актуальных задач распределенных вычислений является расширение спектра задач, которые можно решать с использованием таких систем. В данной работе рассматривается проблема эффективной реализации на ГСПК одного из основных методов решения задач глобальной оптимизации – метода ветвей и границ [4-6].

1. Общие сведения о методе ветвей и границ

Можно выделить два основных семейства методов решения задач конечномерной оптимизации: точные и эвристические. Точные методы позволяют гарантировать точность найденного решения. К этому классу можно отнести различные варианты метода ветвей и границ, отсечений и др. Для точных методов характерна высокая трудоемкость, которая часто не позволяет применять их при решении реальных задач. Эвристические методы основаны на предположениях о свойствах оптимального решения. В отличие от точных, эвристические методы не дают гарантии оптимальности найденного решения. Однако в условиях ограниченности вычислительных ресурсов эвристики зачастую являются единственным способом нахождение решения. Также распространены гибридные методы, при которых эвристические методы применяются для нахождения решения, а точные – для доказательства оптимальности. Эффективность гибридных методов обусловлена тем, что эвристические алгоритмы нередко обладают более высокой скоростью сходимости к оптимуму по сравнению с точными методами.

Одной из наиболее распространенных схем организации точных методов является т.н. метод ветвей и границ (МВГ), основанный на разбиении допустимого множества. Приведем общую схему метода. На протяжении всего времени работы поддерживается список подмножеств допустимого множества («подзадач»). Первоначально он состоит из одного элемента – допустимого множества. Далее выбирается один из элементов списка – подмножество . Если удо-

влетворяет правилам отсева, то выбранный элемент удаляется из списка. В противном случае он подвергается дроблению на более мелкие подмножества с помощью правила декомпозиции. Полученные подмножества замещают в списке выбранный элемент. Алгоритм завершает свою работу, когда в последовательности не остается ни одного элемента. Совокупность правил отсева, правила декомпозиции и способ вычисления рекорда должны выбираться так, чтобы обеспечивать конечность числа шагов МВГ.

2. Принципы реализации метода ветвей и границ в Грид-системах персональных компьютеров

Перечислим особенности, отличающие ГСПК от традиционных параллельных систем:

(1) Невозможность передачи данных между рабочими узлами (клиентами) в процессе расчетов, обмены проводятся только между клиентскими компьютерами и сервером проекта;
(2) Большое время на инициализацию задания, включающее в себя подготовку, копирование данных на вычислительный узел и получение результатов обратно;
(3) Нестабильность работы узлов – любой узел может выйти из расчетов в любой момент времени на неопределенный срок, что приводит к очень существенной разнице во времени обработки расчетных блоков.

Для исследования влияния перечисленных факторов на производительность распределенной реализации метода ветвей и границ и выработки оптимальной стратегии такой реализации была разработан программный прототип для платформы BOINC [7]. BOINC (Berkeley Open Infrastructure for Network Computing) представляет собой платформу с открытым кодом для организации проектов добровольных вычислений. Разработка системы ведется в U.C. Berkeley Spaces Sciences Laboratory (США) исследовательской группой, которая также разрабатывала проект SETI@home. Работа над BOINC была начата в 2002 году с целью создания универсальной программной платформы для проектов подобного рода, кото-

рая бы упростила процесс развертывания необходимой инфраструктуры и разработки приложений. Первый проект добровольных вычислений на основе BOINC был запущен в 2004 году. В настоящее насчитывается более 80 публичных проектов на основе BOINC, делая платформу стандартом де-факто в данной области.

Все программное обеспечение BOINC можно разделить на две компоненты: клиентскую и серверную части программного обеспечения. Клиентская часть устанавливается на вычислительном узле. В её задачи входит:

(1) Подключиться к одному из проектов, к какому именно указывает владелец машины;
(2) Запрашивать задания у центрального сервера;
(3) Скачивать задания с сервера, если они там есть;
(4) Запускать у себя скачанные задания;
(5) Результаты работы заданий отсылать обратно на сервер.

Серверная часть программного обеспечения BOINC выполняет следующие действия:

(1) Создает задания для пересылки на вычислительные узлы;
(2) Отвечает на клиентские запросы, отправляет задания на вычислительные узлы;
(3) Получает результаты работы задания и передает их для дальнейшей обработки;
(4) Содержит в себе web-сервер для получения информации о проекте через web-интерфейс.

Соответственно, распределённое приложение для инфраструктуры BOINC можно разделить на две основные компоненты: клиентскую и серверную части распределённого приложения.

Клиентская часть распределённого приложения и есть исполняемый файл, запускаемый на вычислительном узле. Она выполняет основную вычислительную нагрузку. Серверная часть распределённого приложения создает задания (расчетные блоки) для вычислительных узлов. Как правило, расчётный блок состоит из исполняемого файла клиенткой части, объединённый со специфическим для конкретного задания входным файлом с данными. После отправки задания в вычислительную инфраструктуру серверная часть распределённого приложения ждёт результатов задания. По-

лучив из инфраструктуры все результаты заданий, серверная часть производит их обработку, и создает единый результат работы распределённого приложения.

3. Экспериментальное исследование

Решалась задача глобальной оптимизации

$$f(x) = \sum_{i=1}^{n} x_i^4 - 5x_i^2 + 4 \to \min,$$

$$x_i \in [-A, A], i = 1,..., n.$$

Полагалось $A = 20$, $n = 8$. Искалось решение с точностью $\varepsilon = 0.01$ с помощью метода неравномерных покрытий [8], численная реализация которого следует схеме ветвей и границ.

Распределенные вычисления были организованы по следующей схеме:

(1) Изначально в инфраструктуру отправляется инициирующее задание, содержащее описание поставленной задачи оптимизации с малым числом шагов;
(2) Результат инициирующего задания содержит некоторый достигнутый им минимум и несколько десятков подзадач (областей для дальнейшего исследования);
(3) Серверная часть распределённого приложения запоминает достигнутый минимум и для каждых m подзадач генерирует новый расчетный блок. Таким образом, на основе результатов инициирующего задания создаётся несколько десятков новых заданий. Каждое последующее задание содержит ограничение в $2x10^6$ шагов;
(4) Для каждого последующего расчетного блока возможны два варианта работы. При первом варианте задание полностью выполняется и на сервер проекта возвращается найденный минимум. При втором варианте - в процессе выполнения задания достигается ограничение по числу шагов. В данном случае на сервер отправляется найденный минимум и подзадачи (области) для дальнейшего рассмотрения.

(5) При получении очередного результата серверная часть распределённого приложения обновляет, если необходимо, достигнутую точку экстремума. Если результат содержит области для дальнейшего рассмотрения, то на их основе генерируются новые задания;
(6) Задача считается решенной, если обработаны все результаты итоговых заданий.

Вычислительный эксперимент проводился в рамках тестовой вычислительной инфраструктуры, основанной на технологии BOINC. Серверная часть распределённого приложения была разработана на языке PHP. Для описания задачи и предоставления входных/выходных данных заданий использовался формат JSON.

4. Результаты вычислительного эксперимента

Всего было проведено три серии вычислительных экспериментов.

Данные по каждой из серий приведены в таблице 1.

Таблица 1.Параметры для каждой из серий вычислительных экспериментов

Номер серии	n	m	***Время CPU, сек***	***Всего заданий***	***Заданий с τ < 1 сек.***
1	8	1	2776	608	541
2	8	10	1676	58	33
3	10	10	106460	1926	677

Распределение числа заданий по времени их выполнения для каждой из серий показано в виде гистограмм на рисунке 1. Зависимость общего числа сгенерированных заданий от времени в процессе выполнения для 3-й серии представлена на рисунке 2.

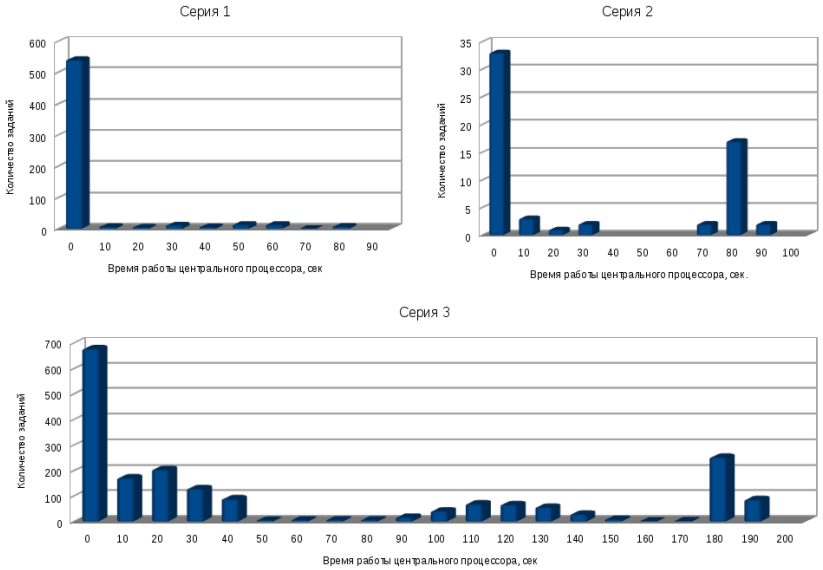

Рис. 1. Распределение числа заданий по времени работы

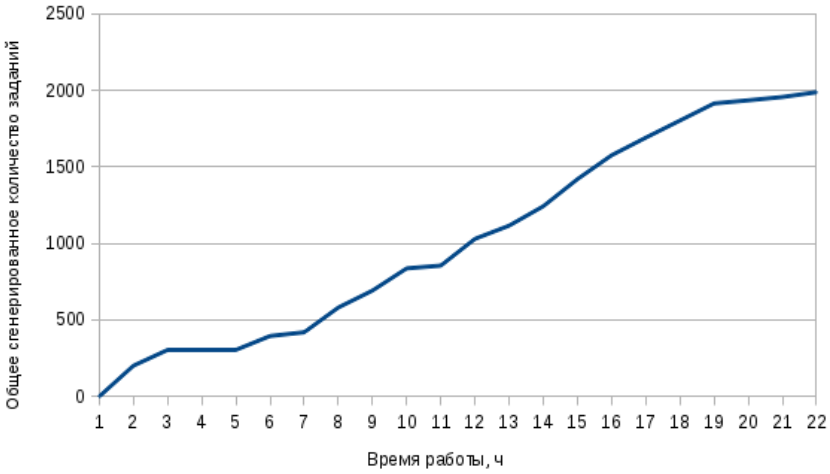

Рис. 2. Зависимость числа заданий от времени для серии №3

В процессе работы распределённого приложения наблюдались следующие эффекты:

- *влияние времени выполнения отдельного запроса.* Выполнение отдельного запроса включает в себя обмен данными и служебной информацией между клиентом и сервером, а это требует передачи некоторого количества данных и некоторого времени на обработку (в большинстве случаев приблизительно 10 секунд). По гистограммам видно, что наибольшее количество заданий выполняются за наименьшее время (меньше секунды). Таким образом, получается, что данные задания неэффективно используют вычислительную инфраструктуру. Повышение числа исследуемых областей на в одном расчетном блоке (число m) понижает количество заданий с минимальным временем выполнения, тем самым несколько исправляя ситуацию. Один из возможных способов повышения эффективности состоит в предварительной оценке вычислительной сложности задания и варьировании числа m в соответствии с вычислительной сложностью.
- *влияние интервала между запросами.* BOINC-клиент по умолчанию устроен таким образом, что при отсутствии заданий на сервере интервал между запросами увеличивается во времени. Данный подход необходим для того, чтобы избежать аномально высокой нагрузки на сеть для больших проектов (несколько тысяч узлов). Новые задания генерируются на основе результатов предыдущих заданий. Таким образом, в процессе работы распределённого приложения периодически возникал эффект, при котором на сервере некоторое время отсутствовали вычислительные задания. Узлы за данное время делали несколько запросов к проекту, не получая заданий, на достаточно продолжительное время переставали делать запросы. За это время работающие узлы возвращали свои результаты на сервер, на основе

этих результатов генерировалась новая серия заданий, однако большая часть вычислительных узлов находилась в состоянии ожидания и забирала задания только через некоторое время. Один из возможных способов повышения эффективности состоит также в контроле числа готовых к выполнению заданий на сервере проекта посредством разделения исследуемых областей и оценки сложности исследования каждой из них.

- *эффект масштаба.* Практика расчётов показала, что для более вычислительно-ёмких заданий снижается роль двух предыдущих эффектов.

Заключение

В процессе работы метод ветвей и границ был реализован на ГСПК, проведены первые тестовые экспериментальные расчёты, выявлены слабые места, понижающие эффективность. На основании проведенного анализа предложены методы для повышения эффективности.

Практика проведения расчётов показала, что:

- для дальнейших исследований необходимо выработать метод предварительной оценки вычислительной сложности задания;
- эффективность использования ресурсов зависит от соотношения общей вычислительной сложности решаемой задачи и потенциальных возможностей инфраструктуры, при выработке наиболее оптимальной стратегии использования ресурсов необходимо учитывать данное соотношение;
- дальнейшие вычислительные эксперименты целесообразно производить с более вычислительно-ёмкими задачами.

Список литературы

[1] Посыпкин М.А. Грид-системы из персональных компьютеров в России: текущее состояние и перспективы // Сборник избранных трудов VII Международной научно-практической конференции «Современные информационные технологии и ИТ-образование». Москва. Изд-во Интуит.ру. 2012 .

[2] Ватутин Э.И. Добровольный метакомпьютинг: современное состояние и перспективы развития // Сборник материалов IX Международной конференции «Распознавание 2010». Курск. Изд-во ЮЗГУ. 2010.

[3] Ивашко Е.Е., Никитина Н.Н. Использование BOINC-грид в вычислительноемких научных исследованиях // Вестник Новосибирского государственного университета. Серия: Информационные технологии. 2013. Т. 11. № 1. С. 53-57.

[4] Евтушенко Ю. Г. Методы решения экстремальных задач и их применение в системах оптимизации. – 1982.

[5] Сигал И.Х., Иванова А.П. Введение в прикладное дискретное программиро-вание. Физматлит, 2002, 240 с.

[6] Евтушенко Ю. Г. Численный метод поиска глобального экстремума функций (перебор на неравномерной сетке) //Журнал вычислительной математики и математической физики. – 1971. – Т. 11. – №. 6. – С. 1390-1403.

[7] D. P. Anderson. Boinc: A system for public-resource computing and storage. In R. Buyya, editor, Fifth IEEE/ACM International Workshop on Grid Computing, pages 4-10, 2004.

[8] Ю. Г. Евтушенко, М. А. Посыпкин. Применение метода неравномерных покрытий для глобальной оптимизации частично целочисленных нелинейных задач. // ЖВМиМФ, 2011, том 51, № 8, с. 1376–1389.