Опыт организации добровольных вычислений на примере проектов OPTIMA@home и SAT@home*

О.С. Заикин[1], М.А. Посыпкин[2], А.А. Семенов[1], Н.П. Храпов[2]

Институт динамики систем и теории управления СО РАН[1],
Институт системного анализа РАН[2]

Описывается процесс построения проектов добровольных вычислений, базирующихся на платформе BOINC и предназначенных для решения задач, предполагающих массовый крупноблочный параллелизм. Детали и особенности этого процесса иллюстрируются на примере реально функционирующих проектов добровольных вычислений OPTIMA@home и SAT@home. Первый проект предназначен для решения задач глобальной оптимизации, второй – для решения комбинаторных задач, сведенных к задачам о булевой выполнимости (SAT).

1. Введение

Добровольные вычисления – относительно новое направление в распределенных вычислениях, идеология которого предполагает предоставление вычислительных ресурсов для организации масштабных расчетов т.н. «волонтерами» (volunteers). Волонтеры или добровольцы – это, как правило, пользователи, располагающие собственными персональными компьютерами (ПК), ресурсы которых они согласны предоставить для решения разнообразных научных задач. Следует отметить, что используются только свободные ресурсы (когда ПК не используется для других целей), поэтому участие в добровольных вычислениях не мешает работе пользователей. ПК добровольцев, которые могут находиться в географически удаленных друг от друга точках, объединяются в грид под управлением некоторой среды. Обычно такие грид-системы создаются под конкретные задачи, на решение которых могут уходить месяцы и даже годы, и называются проектами добровольных вычислений.

За последние несколько лет были организованы проекты добровольных вычислений, предполагающие решение широкого круга задач – от поиска новых космических объектов (Einstein@home) до проверки некоторых гипотез теории чисел (ABC@home) и криптографии (distributed.net).

Наиболее популярная открытая программная платформа для организации добровольных вычислений – это BOINC (Berkeley Open Infrastructure for Network Computing [1]). Она изначально разрабатывалась для проекта SETI@home в U.C. Berkeley Spaces Sciences Laboratory (США). С 2002 года платформа BOINC была сделана открытой и с 2004 года на ее основе стали создаваться другие проекты. На данный момент суммарная мощность всех проектов на платформе BOINC составляет около 6 петафлопс, что сопоставимо с мощностью самого высокопроизводительного суперкомпьютера из списка топ-500 [2].

В настоящей статье описывается технология организации проектов добровольных вычислений, созданных с использованием платформы BOINC. Характерные особенности технологии поясняются на примере двух реально функционирующих проектов – OPTIMA@home и SAT@home. Первый проект предназначен для решения разнообразных оптимизационных задач, второй – для решения комбинаторных задач, сведенных к задачам о выполнимости булевых формул (SAT-задачам).

*Работа выполнена при поддержке РФФИ (гранты № 11-07-00377-а и 10-07-00301-а) и Седьмой Рамочной программы Европейского Союза (FP7/2007-2013), грант № 261561 (DEGISCO).

2. Основные принципы организации проектов добровольных вычислений

Изначально для создания проектов распределенных добровольных вычислений требовалась работа больших коллективов разработчиков. Например, в 1996 году стартовал проект GIMPS по поиску простых чисел Мерсенна, а в 1997 году – distributed.net направленный на решение задачи криптоанализа шифра RC5. Изначально проект предполагал поиск 56-битного секретного ключа. В июле 1996 года на пятой международной конференции по биоастрономии была представлена концепция проекта SETI@home, предназначенного для обработки интенсивных потоков данных, поступающих от мощных радиотелескопов. Данный проект был запущен в 1999 году, а в 2002 году на его основе была разработана открытая платформа BOINC. Использование этой платформы значительно упростило создание новых проектов и на данный момент большинство крупных проектов используют именно BOINC.

Проект добровольных вычислений состоит из трех основных частей: сервера, веб-сайта и прикладного программного обеспечения (ПО). Процессом вычислений управляет сервер, именно на нем создаются задания для обработки. Для того чтобы подключиться к проекту, пользователь (доброволец) должен скачать и установить на свой ПК стандартный «BOINC-клиент». Это специальная программа, позволяющая подключать ПК пользователя к любым BOINC-проектам, осуществляющая обмен данными с сервером выбранного проекта и выделяющая ресурсы ПК пользователя для работы прикладного ПО. Подключение к проекту происходит путем указания URL сайта проекта, после чего BOINC-клиент автоматически скачивает с сервера прикладное ПО (ориентируясь на тип ОС и процессора ПК пользователя) и задания для обработки. Затем весь процесс вычислений на стороне пользователя происходит автоматически. В BOINC-клиенте существует гибкая система настроек, позволяющая эффективно задействовать свободные ресурсы ПК и распределять их между BOINC-проектами.

При организации проекта следует учитывать ряд факторов, способствующих его активному развитию. Специалистами по организации добровольных вычислений обычно выделяются следующие основные причины, мотивирующие пользователей на участие в проекте:

- за каждое выполненное задание пропорционально затраченным вычислительным ресурсам участникам начисляются т.н. «кредиты». Количество кредитов является характеристикой, по которой участники соревнуются между собой. Также участники объединяются в команды по разным признакам (национальному, региональному, пр.), которые также соревнуются между собой;
- при получении результатов обычно на сайте проекта выкладывается информация об участнике, на ПК которого был получен данный результат;
- ощущение причастности к важным научным исследованиям, именно поэтому большой популярностью пользуются медицинские проекты, направленные на поиск новых лекарств (Folding@home, WCG, Rosetta@home).

Поскольку в проектах добровольных вычислений используются ПК частных лиц, всегда имеется возможность того, что результаты вычислений будут сфальсифицированы пользователем, либо будут некорректными в результате аппаратного или программного сбоя. Для учета этого обычно используются избыточные вычисления. А именно, для каждого задания формируются несколько копий, которые отправляются на ПК разных пользователей. Полученные результаты сравниваются, если они совпадают, то задание считается решенным. Если есть расхождения, то результаты аннулируются, формируются новые копии задания, которые отправляются на ПК других пользователей. За различные копии одного задания могут быть запрошены различные количества кредитов. В зависимости от числа копий можно применять различные схемы расчета кредитов.

Существуют три основных статуса проектов добровольных вычислений: «альфа», «бета» и «релиз». Можно выделить следующие критерии, по которым проектам присваивают соответствующие статусы:

- стабильность (доступность сайта и сервера проекта в режиме 24/7, постоянное наличие готовых к отправке заданий);
- степень завершенности серверного и прикладного ПО;

- наличие версий прикладного ПО для основных операционных систем (windows, linux, mac);
- полнота и наглядность описания целей проекта, запланированных и полученных результатов.

Статусы проектов, использующих платформу BOINC, отображены на сайте Formula BOINC [3]. На начало февраля 2012 года из 46 активных проектов 25 имеют статус альфа, 10 – бета и 11 – релиз. Даже для получения статуса альфа проект проходит специальную экспертизу. Этот статус означает, что проект функционирует и находится на начальном этапе разработки. Следует отметить, что на территории СНГ на данный момент активно действуют четыре проекта добровольных вычислений на основе BOINC: три в России (включая OPTIMA@home и SAT@home, которые будут описаны ниже) и один на Украине (SLinCA@home). Все эти проекты имеют статус «альфа».

3. Проект добровольных распределенных вычислений OPTIMA@home

20 июня 2011 года в Институте системного анализа РАН был запущен проект добровольных распределенных вычислений OPTIMA@home [4], предназначенный для исследования эффективной реализации методов оптимизации. Проект OPTIMA@home реализован с использованием платформы BOINC. На **Рис. 1** показана общая схема работы распределенного приложения проекта, включающего в себя две основные части: управляющую и расчетную.

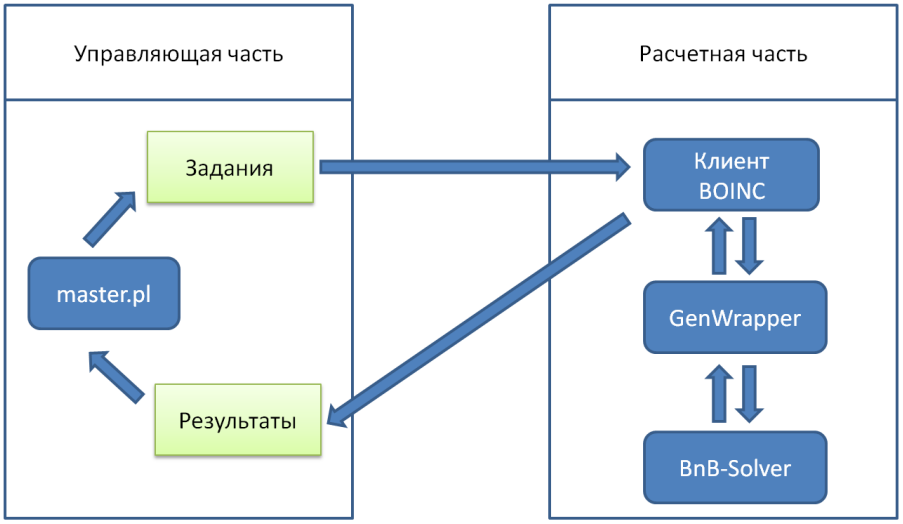

Рис. 1. Общая схема работы распределенного приложения OPTIMA@home

Управляющая часть выполняется на сервере проекта. Ее основная задача – генерировать новые задания и обрабатывать полученные с клиентских машин результаты. В проекте OPTIMA@home серверная часть представляет собой программу master.pl на языке Perl. Эта программа запускается периодически каждые полчаса, просматривает папку с полученными результатами, обрабатывает их и на их основе генерирует новые задания. Сгенерированные задания помещаются в базу данных заданий проекта.

Расчетная часть представляет собой архив выполняемых файлов, который загружается на ПК пользователей. Получив очередное задание, BOINC-клиент запускает расчетную часть, которая обрабатывает полученное задание. Результат возвращается на сервер. Основой расчетной части в проекте OPTIMA@home является выполняемый модуль библиотеки BNB-Solver [5]. Входной файл приложения содержит исходные данные задачи и параметры алгоритма, выходной файл содержит найденные решения. Оба файла используют формат XML.

Для запуска приложения BNB-Solver на пользовательских машинах используется инструмент GenWrapper [6, 7], предназначенный для портирования в грид существующих приложений. При этом не требуется переписывать приложение, вместо этого используется

выполняемый файл программы. На стороне пользователя программа GenWrapper распаковывает архив с выполняемыми файлами приложения и файлами данных и запускает командный файл, написанный на shell-подобном языке. Текст командного файла для проекта OPTIMA@home приведен на **Рис. 2.** Этот файл сначала получает глобальные имена входных/выходных файлов, необходимых для работы приложения, а затем выполняет запуск самого приложения, передавая эти файлы в качестве аргументов командной строки. Перед запуском командного файла программа `GenWrapper` получает исходные файлы от BOINC-клиента, а после выполнения приложения передает ему файлы с результатами.

```
IN=`boinc resolve_filename in`
OUT=`boinc resolve_filename out`
BNBAPP=`boinc resolve_filename bnbss`
./"${BNBAPP}" "${IN}" "${OUT}"
```

Рис. 2. Командный файл, выполняемый на стороне клиента

На данный момент в рамках проекта OPTIMA@home реализован метод Basin-Hopping [8]. Это – эвристический метод, который в результате последовательных возмущений некоторого решения улучшает его. Перед началом работы на сервер загружается набор случайных точек. Каждая из точек служит начальной для работы алгоритма Basin-Hopping на стороне пользователя. Улучшенное решение пересылается на сервер и опять заносится в список заданий, чтобы послужить новой начальной точкой. Таким образом, происходит постоянное улучшение получаемых решений.

В результате удалось создать устойчиво работающий проект. В качестве тестовой задачи была выбрана задача поиска минимума энергии молекулярного кластера Морзе [9]. Полученные решения сопоставимы с получаемыми параллельными методами на суперкомпьютере [10]. На 14 февраля 2011 года проект OPTIMA@home имеет более 2200 подключенных компьютеров. Реальная производительность проекта приближается к одному терафлопсу.

4. Проект добровольных распределенных вычислений SAT@home

4.1 SAT-задачи и сводимые к ним комбинаторные проблемы

SAT-задачи (или «задачи о булевой выполнимости») – это задачи поиска наборов, выполняющих булевы формулы, как правило, в виде конъюнктивных нормальных форм (КНФ). В своей общей постановке задача о булевой выполнимости NP-трудна, однако данная задача настолько важна для современной прикладной кибернетики, что алгоритмы, позволяющие быстро находить решения SAT-задач в различных частных случаях, являются чрезвычайно востребованными. Прогресс исследований в области алгоритмики SAT, наблюдающийся в последние 10 лет, во многом обусловлен тем фактом, что к данной задаче могут быть эффективно (за полиномиальное время) сведены обширные классы комбинаторных задач из, казалось бы, совершенно различных областей. Это и упоминавшиеся выше задачи из теории дискретных управляющих систем, и разнообразные комбинаторные задачи на графах, и задачи поиска различных экстремальных комбинаторных структур и, конечно же, криптографические задачи.

Техника сводимости различных задач к SAT представляет отдельное важное и интересное направление, хотя с чисто теоретической точки зрения эффективность соответствующих процедур есть прямое следствие теоремы С. Кука ([11]). На практике можно каждую конкретную задачу сводить к SAT особым способом либо использовать системы логического проектирования (например, [12]), либо использовать специализированные системы трансляции алгоритмических описаний в булевы уравнения (и в конечном счете в SAT) [13].

Когда SAT-задача, кодирующая некоторую комбинаторную проблему, построена, необходимо определиться с алгоритмом ее решения. На подавляющем множестве SAT-задач, имеющих реальные «прототипы», лучшие результаты показывают SAT-решатели, базирующиеся на алгоритме DPLL [14].

Параллельные SAT-решатели стали массово появляться совсем недавно, несмотря на то, что первые теоретические работы по распараллеливанию соответствующих алгоритмов были опубликованы в 90-х года прошлого века ([15]). Конкурсы параллельных SAT-решателей регулярно проводятся с 2008 года [16]. В своем подавляющем большинстве современные параллельные SAT-решатели предполагают интенсивный обмен данными между вычислительными узлами – конкретно, происходит обмен параллельно накапливаемыми булевыми ограничениями (т.н. «конфликтными дизъюнктами). Конечно же, использование таких решателей в проектах добровольных вычислений, да и вообще в грид, весьма проблематично. Тем не менее, отметим очень интересную работу [17], где описывается подход к построению распределенного SAT-решателя, обмен ограничениями в котором реализован в peer-to-peer сети, состоящей из ПК.

Другой подход к построению распределенных SAT-решателей в рамках проектов добровольных вычислений предполагает крупноблочный параллелизм [18]. В рамках данного подхода основной упор делается на исследование исходной комбинаторной задачи и некоторый (возможно, довольно трудоемкий) препроцессинг, результатом которого является построение параллельного списка заданий. Для обработки данного списка используются независимые друг от друга узлы распределенной вычислительной среды. Управление обработкой списка (рассылка заданий, получение и анализ ответов) происходит на специально выделенном для этой цели сервере.

Анализ исходной задачи и осуществление декомпозиции также может потребовать ресурсов параллельного вычислителя (как правило, для этой цели достаточно ресурсов маломощного кластера). Однако суммарные вычислительные затраты на этот этап существенно меньше тех, которые потом будут использованы для обработки построенного параллельного списка.

Весьма детально процесс декомпозиции SAT-задач, кодирующих проблемы обращения дискретных функций, описан в [19] (см. также [20]). Здесь мы приведем совсем краткое описание этой техники.

Рассматривается некоторая дискретная функция, преобразующая двоичные последовательности в двоичные последовательности, которая задается программой для детерминированной машины Тьюринга (ДМТ), имеющей полиномиальную сложность. Иными словами, речь идет о семействе функций вида

$$f_n : \{0,1\}^n \to \{0,1\}^*,$$

вычислимых некоторым полиномиальным (от n) алгоритмом A, причем для каждого $n \in \mathrm{N}$ $dom\, f_n = \{0,1\}^n$. Здесь $\{0,1\}^n$ – множество всех двоичных слов длины n, а $\{0,1\}^*$ – множество всевозможных двоичных слов произвольной конечной длины (включая, вообще говоря, пустое слово). Задача обращения функции f_n состоит в следующем: зная текст программы A и некоторое $y \in range\, f_n$, найти произвольное $x \in \{0,1\}^n$, такое, что $f_n(x) = y$.

Используя идеи С. Кука о пропозициональном кодировании алгоритмов, можно свести данную проблему к проблеме поиска выполняющего набора выполнимой КНФ. Для этой цели можно описать действие алгоритма A на всевозможных словах из $\{0,1\}^n$ в форме схемы из функциональных элементов над любым полным базисом (например, над $\{\&, \neg\}$), а затем при помощи стандартных процедур (см., например, [21]) построить по такой схеме и вектору $y \in range\, f_n$ КНФ $C(f_n)$, из произвольного выполняющего набора которой можно эффективно выделить такой $x \in \{0,1\}^n$, что $f_n(x) = y$. Таким образом, рассматриваемая задача обращения функции эффективно сведена к задаче поиска выполняющего набора КНФ $C(f_n)$, то есть к SAT-задаче.

Базовая схема крупноблочного распараллеливания полученной SAT-задачи заключается в следующем. Среди переменных КНФ $C(f_n)$ выбирается некоторое подмножество, состоящее из d переменных, после чего варьируются всевозможные их значения. Подстановка каждого

такого вектора значений истинности $\alpha \in \{0,1\}^d$ в $C(f_n)$ дает новую КНФ $C_\alpha(f_n)$, которая и является элементарным заданием параллельного списка. Всего, таким образом, в получаемом списке имеется 2^d элементарных заданий.

Если при переходе от текста алгоритма A к КНФ $C(f_n)$ использовать ряд простых правил (см. [19]), то весьма просто гарантировать, что мощность списка, каждое задание в котором является вычислительно простым, заведомо не превосходит 2^n. Как правило, эту мощность можно весьма существенно уменьшить, используя довольно естественные вычислительные процедуры, которые в совокупности составляют этап препроцессинга (см. [18]). После того как параллельный список сформирован, он загружается в грид, общие принципы архитектуры которой (клиент-сервер) были описаны во втором разделе.

Описанный подход был с успехом применен в криптоанализе ряда генераторов поточного шифрования. В 2009 году этот подход был реализован в вычислительной системе BNB-Grid (см. [22]) в применении к криптоанализу широко известного генератора ключевого потока A5/1 (описание генератора взято из работы [23]).

Сразу оговоримся, что задачи криптоанализа не являются самоцелью, а выступают в роли аргументированно трудных тестов. При этом мы исходим из того, что успешная апробация вычислительной технологии на криптографических тестах означает ее принципиальную применимость для решения задач, вычислительная трудность в которые не заложена искусственно. Успешность решения задач криптоанализа в грид-системе BNB-Grid стимулировала наши исследования в направлении построения проекта добровольных вычислений, ориентированного на решение SAT-задач. Таким проектом стал SAT@home.

4.2 Описание проекта добровольных вычислений SAT@home

29 сентября 2011 года был запущен SAT@home [24] – совместный проект Института системного анализа РАН и Института динамики систем и теории управления СО РАН. При создании проекта была использована открытая платформа BOINC [1] и пакет SZTAKI Desktop Grid [25]. С применением библиотеки DC-API [26] было создано распределенное приложение, состоящее из серверной и клиентской части. Схема работы приложения представлена на **Рис. 3**. Серверная часть отвечает за создание заданий в базе данных проекта, а также за обработку результатов выполнения заданий, присылаемых с ПК пользователей. Отправкой заданий на ПК пользователей и получением результатов занимаются стандартные службы BOINC. Для формирования заданий в серверную часть приложения были перенесены процедуры из решателя PD-SAT [27], отвечающие за декомпозицию SAT-задач.

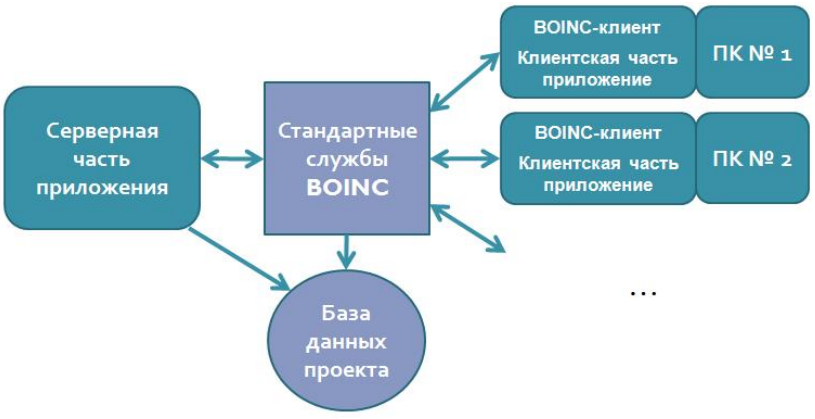

Рис. 3. Схема работы распределенного приложения в проекте SAT@home

В настоящий момент в проекте используется схема, представленная на **Рис. 4**. В соответствии с этой схемой решатель PD-SAT (запущенный на кластере Blackford ИДСТУ СО РАН [28]) получает исходную SAT-задачу (КНФ в формате DIMACS [16]) и вычисляет для нее «хорошие» параметры декомпозиции. Под «хорошими» имеются в виду параметры, найденные

в результате применения специальной техники прогнозных функций, описанной в [18]. Параметры декомпозиции включают в себя способ выбора переменных, включаемых в декомпозиционное множество, и число этих переменных. Данный этап требует относительно небольшого времени (от нескольких минут до нескольких часов). Найденные параметры передаются серверной части приложения проекта, которая осуществляет декомпозицию исходной SAT-задачи на подзадачи. Клиентскую часть приложения в виде исполняемых файлов для конкретной операционной системы запускает на ПК пользователей стандартный BOINC-клиент. Основу клиентской части приложения составляют SAT-решатели minisat-1.14.1 и minisat-2.0 [29], модифицированные с учетом особенностей КНФ, кодирующих задачи обращения дискретных функций [19]. В клиентскую часть приложения перенесены процедуры решения SAT-задач из решателя PD-SAT [27]. Полученные результаты BOINC-клиент отправляет на сервер. В результате решения в общем случае всех подзадач находится решение исходной SAT-задачи.

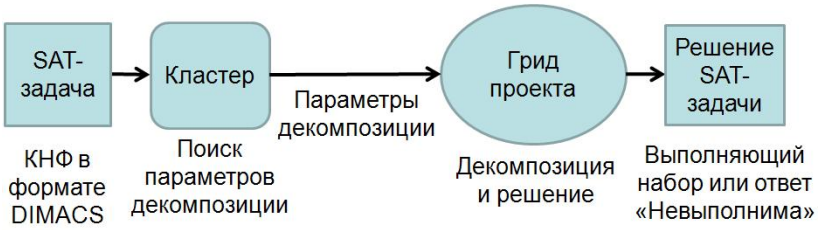

Рис. 4. Схема решения SAT-задач в проекте SAT@home

По состоянию на 14 февраля 2012 г. SAT@home имеет следующие характеристики:

- 1246 пользователей, 79 % иностранных;
- 3532 ПК, суммарно 13743 ядер процессоров, 80 % под управлением ОС Windows;
- версии клиентского приложения: windows x86, linux x86, linux x64;
- средняя реальная производительность грида проекта 1.5 терафлопс, пиковая 4,3 терафлопс.

На **Рис. 5** отображена динамика количества подключаемых к проекту ПК с 29 сентября 2011 года по 14 февраля 2012 года. Каждый столбец отображает количество подключенных ПК с момента старта проекта по конкретный день. На графике видно, что с середины октября 2011 года подключений стало значительно больше. Это объясняется тем, что 10 октября проект был добавлен на статистический сайт Free-DC [30] (содержащий информацию о проектах на платформе BOINC), а также был включен экспорт статистики, позволяющий другим статистическим сайтам добавлять проект в свои списки. Наличие информации о проекте на основных статистических сайтах – важный фактор для привлечения новых пользователей. На **Рис. 6** представлена динамика изменения реальной мощности грида проекта в гигафлопсах (с 17 января по 14 февраля 2012 года).

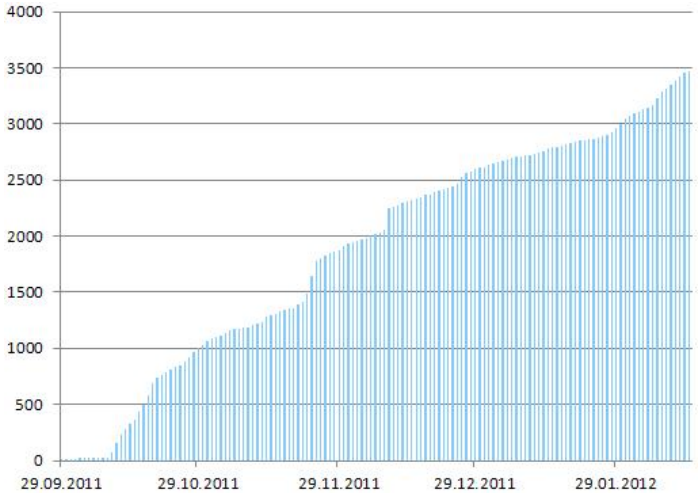

Рис. 5. Динамика количества ПК, подключенных к проекту SAT@home

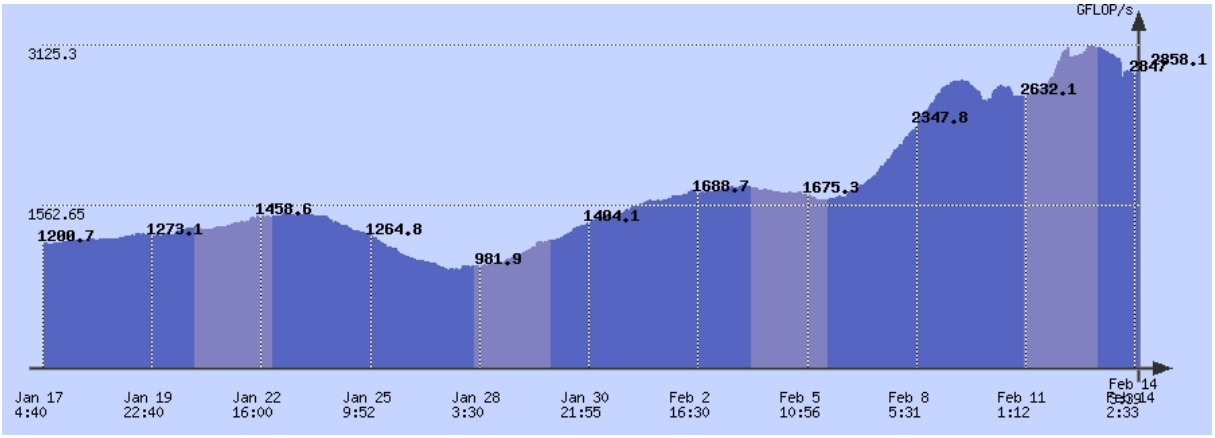

Рис. 6. Динамика изменения реальной производительности проекта (в гигафлопсах)

4.3 Результаты, полученные в проекте добровольных вычислений SAT@home

В проекте SAT@home в середине октября 2011 года запущен эксперимент, направленный на решение серии SAT-задач, кодирующих обращение функции, порождающей ключевой поток в генераторе A5/1 (длина инициализирующей последовательности 64 бита, анализируется фрагмент ключевого потока длиной 114 бит, соответствующий одному фрейму ключевого потока [31]). На сегодняшний день наиболее эффективным методом криптоанализа A5/1 является т.н. «rainbow-метод» [32]. Реализованные при этом rainbow-таблицы [32] покрывают ключевое пространство примерно на 88% (при реалистичных условиях криптоанализа). На текущей момент в проекте SAT@home решаются 10 примеров криптоанализа шифра A5/1, которые не покрываются этими rainbow-таблицами. Декомпозиция SAT-задачи, кодирующей криптоанализ A5/1, на семейство подзадач проводится по 31 переменной (структура декомпозиционного множества описана в [20]). Таким образом, для решения одного примера криптоанализа требуется решить 2^{31} (более 2 миллиардов) относительно простых подзадач, которые объединяются в пакеты по 32768 подзадачи в каждом. Каждый такой пакет является пользовательским заданием. На решение одного задания требуется обычно не более 4 часов работы одного ядра среднего по мощности процессора. На данный момент в SAT@home успешно решены пять SAT-задач из описанной выше серии. Эти результаты можно найти в разделе «Найденные решения» сайта проекта [24].

Следует отметить, что в процессе работы проектов добровольных вычислений целесообразно организовывать командные соревнования среди пользователей. За время работы проект SAT@home было организовано два таких соревнования на сайте BOINCStats [33]. На период каждого из этих соревнований производительность проекта возрастала примерно в 2 раза в сравнении с его средней производительностью. Это можно проследить на графике производительности проекта (с 10 февраля 2012 года началось семидневное соревнование).

Заключение

Концепция добровольных вычислений идеально подходит для реализации вычислительных экспериментов, привлекающих существенные ресурсы распределенных вычислителей на протяжении длительного времени (месяцы и даже годы). Описанные в работе технологии могут использоваться для организации специализированных проектов, ориентированных на решение крупномасштабных научных задач. Ключевые пункты таких технологий проиллюстрированы на примере двух реально работающих проектов – OPTIMA@home и SAT@home. Отметим, что проект SAT@home является на текущий момент единственным действующим на территории СНГ проектом добровольных распределенных вычислений, входящим в список активных проектов, составленный разработчиками платформы BOINC (внесен 7 февраля 2012 года).

Литература

1. Платформа BOINC для организации распределенных вычислений URL: http://boinc.berkeley.edu/ (дата обращения: 14.02.2012).
2. Рейтинг и описание 500 самых мощных общественно известных суперкомпьютеров URL: http://www.top500.org/ (дата обращения: 14.02.2012).
3. Formula BOINC - статистический сайт проектов добровольных вычислений URL: http://formula-boinc.org/ (дата обращения: 14.02.2012).
4. Проект распределенных добровольных вычислений OPTIMA@home URL: http://boinc.isa.ru/dcsdg/ (дата обращения: 14.02.2012).
5. Evtushenko Y., Posypkin M., Sigal I. A framework for parallel large-scale global optimization // Computer Science - Research and Development. 2009. V. 23. Issue 3-4. P. 211-215.
6. Marosi A. C., Balaton Z., Kacsuk P. GenWrapper: A Generic Wrapper for Running Legacy Applications on Desktop Grids // Third Workshop on Desktop Grids and Volunteer Computing Systems, 2009, Rome, Italy, IEEE.
7. Проект GenWrapper URL: http://genwrapper.sourceforge.net/ (дата обращения: 14.02.2012).
8. Leary R.H. Global Optima of Lennard-Jones Clusters // Journal of Global Optimization. 1997. Vol. 11, Issue 1. P. 35-53.
9. Longjiu Cheng, Jinlong Yang. Global Minimum Structures of Morse Clusters as a Function of the Range of the Potential: 81 <= N <= 160 // Journal of Physical Chemistry A. 2007. vol. 111. P. 5287-5293.
10. Посыпкин М.А. Методы и распределенная программная инфраструктура для численного решения задачи поиска молекулярных кластеров с минимальной энергией // Вестник нижегородского университета им. Н.И. Лобачевского. 2010. № 1. С. 210-219.
11. Cook S.A. The complexity of theorem-proving procedures // Third annual ACM symposium on Theory of computing, 1971, Ohio, USA. ACM. 1971. P. 151-159.
12. Thomas D., Moorby P. The Verilog hardware description language. Springer, 2002. 381 p.
13. Отпущенников И.В., Семенов А.А. Технология трансляции комбинаторных проблем в булевы уравнения // Прикладная дискретная математика. 2011. № 1. С. 96–115.
14. Davis M., Longemann G., and Loveland D. A machine program for theorem proving // Communication of the ACM. 1962. Vol. 5, Issue 7. P. 394-397.
15. Bohm M., Speckenmeyer E. A fast parallel SAT solver - efficient workload balancing // Annals of Mathematics and Artificial Intelligence. 1996. Vol. 17, No. 2, P. 381-400.
16. SATLive! – современные ссылки по задаче SAT URL: http://www.satlive.org/ (дата обращения: 14.02.2012).
17. Schulz S., Blochinger W. Parallel SAT Solving on Peer-to-Peer Desktop Grids // Journal Of Grid Computing. 2010. Vol. 8 No. 3. P. 443-471.
18. Заикин О.С., Семенов А.А. Технология крупноблочного параллелизма в SAT-задачах // Проблемы управления. 2008. № 1. С. 43–50.
19. Semenov A., Zaikin O., Bespalov D., Posypkin M. Parallel algorithms for SAT in application to inversion problems of some discrete functions. arXiv:1102.3563v1 [cs.DC].
20. Semenov A., Zaikin O., Bespalov D., Posypkin M. Parallel logical cryptanalysis of the generator A5/1 in BNB-Grid system // Lecture Notes in Computer Science. 2011. Vol. 6873. P. 473-483.

21. Prestwich S. CNF encodings. In Handbook of Satisfiability (editors: A. Biere, M.Heule, H. van Maaren, T. Walsh). 2009. IOS Press. P. 75-97.

22. Посыпкин М.А., Заикин О.С., Беспалов Д.В., Семенов А.А. Решение задач криптоанализа поточных шифров в распределенных вычислительных средах // Труды ИСА РАН. 2009. Т. 46. С. 119-137.

23. Biryukov A., Shamir A., Wagner D. Real Time Cryptanalysis of A5/1 on a PC // Seventh Fast Software Encryption Workshop, 2000, New York, USA. Springer-Verlag, 2000, P. 1-18.

24. Проект распределенных добровольных вычислений SAT@home
URL: http://sat.isa.ru/pdsat/ (дата обращения: 14.02.2012).

25. Kacsuk P., Kovács J., Farkas Z., Marosi A. Cs., Gombás G., Balaton Z. SZTAKI Desktop Grid (SZDG): A Flexible and Scalable Desktop Grid System // Journal Of Grid Computing. 2009. Vol. 7, No. 4. P. 439-461.

26. Balaton Z., Gombas G., Kacsuk P., Kornafeld A., Kovacs J., Marosi A. C., Vida G., Podhorszki N., Kiss T. Sztaki desktop grid: a modular and scalable way of building large computing grids // 21th International Parallel and Distributed Processing Symposium, 2007, Long Beach, California, USA. P. 1-8.

27. Заикин О.С. Реализация процедур прогнозирования трудоемкости параллельного решения SAT-задач // Вестник УГАТУ. 2010. Т. 14, № 4. С. 210-220.

28. Суперкомпьютерный центр ИДСТУ СО РАН
URL: http://www.mvs.icc.ru (дата обращения: 14.02.2012).

29. Решатель MiniSat
URL: http://minisat.se/MiniSat.html (дата обращения: 14.02.2012).

30. Free-DC - статистический сайт проектов добровольных вычислений
URL: http://www.free-dc.org/ (дата обращения: 14.02.2012).

31. Barkan E., Biham E., Keller N. Instant Ciphertext-Only Cryptanalysis of GSM Encrypted Communication // Proceedings in Crypto 2003. P. 600-616.

32. A5/1 Cracking project
URL: http://reflextor.com/trac/a51/wiki (дата обращения 14.02.2012).

33. BOINCStats
URL: http://ru.boincstats.com/ (дата обращения 14.02.2012).

34. Список активных BOINC-проектов
URL: http://boinc.berkeley.edu/projects.php (дата обращения 14.02.2012).