© 2012 г. О.С. ЗАИКИН, канд. техн. наук
(Институт динамики систем и теории управления СО РАН, Иркутск),
М.А. ПОСЫПКИН, канд. физ.-мат. наук
(Институт системного анализа РАН, Москва),
А.А. СЕМЕНОВ, канд. техн. наук
(Институт динамики систем и теории управления СО РАН, Иркутск)

ПРОЦЕДУРЫ ПОСТРОЕНИЯ ДЕКОМПОЗИЦИОННЫХ МНОЖЕСТВ ДЛЯ РАСПРЕДЕЛЕННОГО РЕШЕНИЯ SAT-ЗАДАЧ В ПРОЕКТЕ ДОБРОВОЛЬНЫХ ВЫЧИСЛЕНИЙ SAT@home[1]

В статье предложен новый подход к построению декомпозиционных множеств, используемых для крупноблочного распараллеливания SAT-задач и их решения в распределенных вычислительных средах. Предложенные алгоритмы используется в проекте добровольных распределенных вычислений SAT@home.

CONSTRUCTING A DECOMPOSITION SET FOR DISTRIBUTED SOLVING OF SAT-PROBLEMS IN THE PROJECT OF VOLUNTEER COMPUTING SAT@HOME / O.S. Zaikin, A.A. Semenov (Institute for system dynamics and control theory of Siberian Branch of Russian Academy of Sciences, Lermontov street 134, Irkutsk, 664033, Russia, E-mail: zaikin.icc@gmail.com, biclop.rambler@yandex.ru), M.A. Posypkin (Institute for Systems Analysis of Russian Academy of Sciences, pr. 60-letiya Oktyabrya 9, Moscow, 117312, Russia, E-mail: posypkin@isa.ru). In the paper an approach for constructing a decomposition set for coarse-grained parallelization of SAT-problems is proposed. We use such sets for distributed solving of hard SAT-problems. The proposed algorithms are used in the project of volunteer distributed computing SAT@home.

1. Введение

Обширный класс задач современной кибернетики можно рассматривать в контексте общей проблемы поиска решений булевых уравнений. Задачи поиска решений уравнений вида КНФ=1 (КНФ – конъюнктивная нормальная форма) называются SAT-задачами [1]. Спектр применения SAT-подхода очень широк – на сегодня известно множество работ, в которых различные комбинаторные проблемы ставятся и решаются в форме SAT-задач. Сказанное касается верификации, криптографии, комбинаторики, биоинформатики и других областей. Все известные алгоритмы решения SAT-задач экспоненциальны в худшем случае (SAT-проблемы NP-трудны в общей постановке). Однако современные SAT-решатели успешно справляются с обширными классами «индустриальных» тестов, в основе которых лежат задачи из перечисленных выше областей. Повышение эффективности

[1]Работа выполнена при поддержке РФФИ: гранты № 11-07-00377-а и 10-07-00301-а.

решения SAT-задач, в том числе разработка алгоритмов, работающих в параллельных и распределенных вычислительных средах, является практически важным и актуальным направлением исследований.

При разработке вычислительных алгоритмов, применяемых к решению NP-трудных задач, принципиальным является вопрос аргументации эффективности таких алгоритмов. Если предлагаемые алгоритмы показывают хорошие результаты на аргументированно трудных тестах, то разумно предполагать, что они будут применимы и к задачам, вычислительная трудность в которые не заложена искусственно. В настоящей работе в качестве аргументированно трудных тестов используются SAT-задачи, кодирующие проблемы криптоанализа ряда систем шифрования.

Далее мы представим новый подход к крупноблочному распараллеливанию SAT-задач, применяемый в проекте добровольных распределенных вычислений SAT@home. Проект SAT@home [2]–[4] создан в ИДСТУ СО РАН в сотрудничестве с ИСА РАН и представляет собой систему добровольных распределенных вычислений на платформе BOINC [5], предназначенную для решения задачи о булевой выполнимости (SAT). Данный проект был запущен 29.09.2011г. С 21.12.2011г. по 07.05.2012г. в SAT@home решались задачи криптоанализа известного генератора ключевого потока A5/1.

Приведем краткий план статьи. В следующем разделе будут в общих чертах описаны техники сведения различных комбинаторных проблем к SAT-задачам. Третий раздел содержит описание как известных, так и новых методов крупноблочного распараллеливания SAT-задач, ориентированных на использование в распределенных вычислительных средах. В четвертом разделе кратко описан проект добровольных распределенных вычислений SAT@home. Пятый раздел содержит результаты вычислительных экспериментов.

2. Сведение комбинаторных проблем к SAT-задачам

Теоретически возможность эффективного (за полиномиальное время) преобразования проблемы распознавания произвольного языка из класса NP в проблему распознавания языка выполнимых КНФ есть следствие теоремы С. Кука [6]. Используя общие идеи С.Кука, можно сводить к проблеме поиска набора, выполняющего КНФ, разнообразные комбинаторные задачи. Смысл этих действий в том, что для решения SAT-задач существуют программные разработки, называемые SAT-решателями, которые весьма хорошо себя зарекомендовали за последние несколько лет [7]. Подавляющее большинство современных эффективных SAT-решателей построено на базе алгоритма DPLL [8], [9].

Известно множество различных примеров преобразования комбинаторных задач в SAT-задачи [10]. В ситуациях, когда исходная задача является задачей обращения дискретной функции (к данному типу относятся задачи криптоанализа), можно использовать системы автоматической трансляции процедурных описаний функций в виде программ в системы булевых уравнений и SAT-задачи (например, [11]). Общий принцип такого рода трансляторов состоит в следующем.

Пусть дана дискретная функция

$$f : \{0,1\}^* \to \{0,1\}^*,$$

определенная всюду на $\{0,1\}^*$ и вычислимая за полиномиальное время программой T_f для детерминированной машины Тьюринга. Данная программа естественным образом задает счетное семейство функций

$$f_k : \{0,1\}^k \to \{0,1\}^*, k \in \mathrm{N}.$$

Каждую функцию f_k из данного семейства можно реализовать схемой $S(f_k)$ из функциональных элементов некоторого полного базиса, например, $\{\&, \neg\}$. Функция роста размера (т.е. числа функциональных элементов) получаемого семейства схем будет ограничена сверху полиномом от k. Рассматриваем схему $S(f_k)$ как направленный граф, в множестве вершин которого выделены k входных. Входным вершинам сопоставляются булевы переменные $x_1,...,x_k$. Пусть g – произвольная внутренняя вершина схемы $S(f_k)$. Данной вершине соответствует функциональный элемент $E(g) \in \{\&, \neg\}$. Сопоставим вершине g новую булеву переменную $v(g)$ и формулу $F(g)$, которая имеет вид либо $v(g) \equiv u \& w$, если $E(g)$ – это «&», либо $v(g) \equiv \bar{u}$, если $E(g)$ – это «¬». Здесь u, w – булевы переменные, соответствующие входам элемента g, то есть родителям вершины g в графе, представляющем схему $S(f_k)$. Пусть $C(g)$ – КНФ-представление формулы $F(g)$. КНФ, кодирующая схему $S(f_k)$, имеет следующий вид:

$$\&_{g \in S(f_k)} C(g).$$

Тогда КНФ, кодирующая задачу обращения функции f_k в точке $y \in Range\ f_k$, $y = (\sigma_1,...,\sigma_m)$, $\sigma_j \in \{0,1\}$, $j \in \{1,...,m\}$, выглядит следующим образом:

$$C(f_k, y) = \left(\&_{g \in S(f_k)} C(g) \right) \cdot y_1^{\sigma_1} \cdot ... \cdot y_m^{\sigma_m},$$

где

$$z^{\sigma} = \begin{cases} z, \sigma = 1 \\ \bar{z}, \sigma = 0, \end{cases}$$

$y_j, j \in \{1,...,m\}$ – булевы переменные, соответствующие выходам схемы $S(f_k)$. Используя идеи работы [12], несложно показать, что решив булево уравнение $C(f_k, y) = 1$, мы можем найти такое слово $x \in \{0,1\}^k$, что $f_k(x) = y$.

Подобный подход используется в системе Transalg [11], которая предназначена для трансляции в булевы уравнения программ, написанных на специальном С-подобном языке. Все рассматриваемые в настоящей работе SAT-задачи, кодирующие задачи обращения дискретных функций, построены при помощи этой системы.

3. Алгоритмы крупноблочного распараллеливания SAT-задач

Как уже говорилось, SAT-подход можно применять к решению комбинаторных задач из весьма широкого класса. В связи с этим актуальна проблема построения для решения SAT-задач алгоритмов, работающих в параллельных и распределенных вычислительных средах. За последние 4 года появился ряд SAT-решателей, использующих обмен накапливаемыми конфликтными ограничениями между параллельно работающими вычислительными узлами. Большинство таких решателей являются многопоточными приложениями [13]–[15] и обычно задействуют небольшое число ядер. Известные MPI-решатели применимы лишь к некоторым ограниченным классам задач [16]. В силу высокой вычислительной трудности некоторых SAT-задач особую актуальность приобретает проблема разработки распределенных алгоритмов, работающих в распределенных средах со слабым взаимодействием между узлами. Исследования по данной тематике стали появляться совсем недавно. Отметим статью [17], в которой описан распределенный SAT-

решатель, работающий в peer-to-peer–сетях, а также работы [18], [19], содержащие опыт решения SAT-задач в грид-средах.

Специализированная грид-среда, в которой решалась SAT-задача, кодирующей криптонализ широко известного генератора поточного шифрования A5/1, была описана в [20], [21]. В данных работах использовалась техника распараллеливания SAT-задач, предложенная в статье [22]. Далее мы в общих чертах описываем подход работы [22] и развиваем его в направлении построения автоматических процедур поиска декомпозиционных множеств с «хорошими» свойствами.

Итак, пусть нам дана произвольная SAT-задача в виде уравнения $C=1$, где C – конъюнктивная нормальная форма над множеством булевых переменных $X=\{x_1,...,x_n\}$. Произвольное множество $\tilde{X}=\{x_{i_1},...,x_{i_d}\}$, $\tilde{X}\subseteq X$, назовем декомпозиционным множеством. Пусть $Y\in\{0,1\}^d$ – произвольный набор значений истинности переменных из $\tilde{X}$. Через $C|_Y$ обозначается КНФ, полученная в результате подстановки в C набора Y. Множество $\Delta(C,\tilde{X})=\{C|_Y\}_{Y\in\{0,1\}^d}$ называется декомпозиционным семейством для исходной SAT-задачи. Очевидно, что решив все SAT-задачи из $\Delta(C,\tilde{X})$, мы получим решение исходной SAT-задачи. Обработку множества $\Delta(C,\tilde{X})$ можно осуществлять в распределенной вычислительной среде. Однако при различных альтернативах $\Delta(C,\tilde{X})$ мы будем получать различное время его обработки. Очевидно, что в данной ситуации необходим компромисс между числом SAT-задач в $\Delta(C,\tilde{X})$ и их сложностью. Для построения таких компромиссных декомпозиционных множеств в [22] был предложен метод прогнозных функций. Значением прогнозной функции на конкретном $\tilde{X}$ является прогноз общей трудоемкости обработки множества $\Delta(C,\tilde{X})$. Вычисляется это значение следующим образом: случайно из $\{0,1\}^d$ выбираются векторы $Y_1,...,Y_Q$, где Q – число, такое, что $Q<<2^d$. Строится семейство КНФ $\{C|_{Y_1},...,C|_{Y_Q}\}$, которое обрабатывается SAT-решателем S (на рабочей станции или вычислительном кластере). Пусть t – время обработки данной случайной выборки. Тогда $F(C,\tilde{X})=2^d\cdot\frac{t}{Q}$ – значение прогнозной функции на $\tilde{X}$.

Несложно понять, что всегда можно эффективно вычислить некоторое стартовое значение прогнозной функции. Действительно, это справедливо, например, если $\tilde{X}=X$. Если мы решаем SAT-задачу, кодирующую обращение некоторой дискретной функции, и в роли S используем решатель на базе алгоритма DPLL, то в качестве стартового декомпозиционного множества можно выбрать ядро DPLL-вывода [23]. Значение прогнозной функции на этом множестве также подсчитывается эффективно.

В работах [20]–[22] была использована весьма простая стратегия улучшения значений прогнозной функции. А именно, в качестве стартового декомпозиционного множества $\tilde{X}_1$ выбиралось ядро DPLL-вывода (поскольку рассматривались только задачи обращения дискретных функций). Далее делались попытки улучшить значение прогнозной функции на множествах следующего вида:

$$\tilde{X}_1\supset\tilde{X}_2\supset...\supset\tilde{X}_r\supset\tilde{X}_*\supset...\supset\tilde{X}_s,$$

причем для всех $k \in \{1,...,s-1\}$ выполнялось $|\tilde{X}_{k+1}| = |\tilde{X}_k| - 1$. Здесь $\tilde{X}_*$ – множество с наилучшим значением прогнозной функции среди множеств $\tilde{X}_1,...,\tilde{X}_s$. Оно и выбиралось на роль декомпозиционного множества для решения рассматриваемой SAT-задачи в распределенной вычислительной среде.

Описанная стратегия, несмотря на свою простоту, дала неплохие результаты в криптоанализе некоторых генераторов ключевого потока. Однако она оказалась бесполезной при решении задачи криптоанализа генератора A5/1. Декомпозиционное множество для SAT-задачи, кодирующей криптоанализ данного генератора, фактически было найдено «вручную» [21]. Это множество изображено на рисунке 1 – булевы переменные, включаемые в множество, кодируют значения ячеек, выделенных серой заливкой (описание генератора A5/1 взято из статьи [24]).

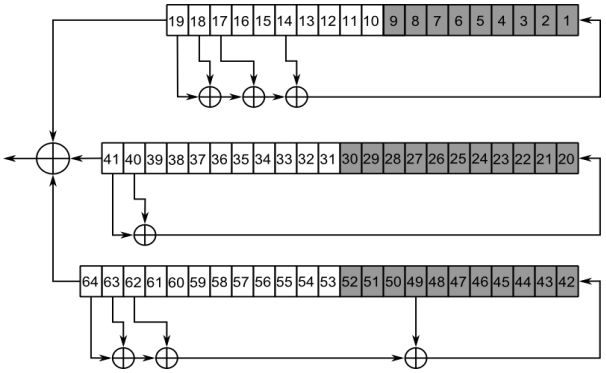

Рис. 1 Декомпозиционное множество, структура которого обусловлена особенностями алгоритма генератора A5/1 [21].

Далее мы рассматриваем общую задачу поиска декомпозиционного множества с наилучшим значением прогнозной функции как задачу глобальной оптимизации на конечном множестве и предлагаем для ее решения новую стратегию.

Итак, рассматриваем SAT-задачу $C = 1$, где C – КНФ над множеством булевых переменных $X = \{x_1,...,x_n\}$. Произвольное множество $\tilde{X} \subset X$ будем задавать характеристическим вектором $\alpha^{\tilde{X}} = \left(\alpha_1^{\tilde{X}},...,\alpha_n^{\tilde{X}}\right)$, таким, что

$$\alpha_i^{\tilde{X}} = \begin{cases} 1, \textit{если } x_i \in \tilde{X} \\ 0, \textit{если } x_i \notin \tilde{X}. \end{cases}$$

Значениями аргумента прогнозной функции $\Phi(\cdot)$ – являются всевозможные векторы вида $\alpha^{\tilde{X}}$. Полагаем, что $\Phi\left(\alpha^{\tilde{X}}\right) = F\left(C, \tilde{X}\right)$ (определение функции $F\left(\tilde{X}, C\right)$ дано выше). Переход к векторам $\alpha^{\tilde{X}}$ нам нужен лишь для того, чтобы корректно определять окрестности точек в пространстве поиска (очевидно, что таким пространством является множество 2^X). После этого мы можем использовать любую схему поиска глобального минимума функции $\Phi(\cdot)$ на 2^X.

Приведем здесь ряд аргументов, обосновывающих использование схемы, описание которой приведено ниже. Во-первых, отметим, что функция $\Phi(\cdot)$ не задана аналитически – ее значения есть величины, наблюдаемые во времени, фактически это реакция вычислительной среды на выбранную форму распараллеливания (вид декомпозиционного множества). В связи с этим невозможно использовать для оптимизации $\Phi(\cdot)$ «традиционные»

методы, так или иначе привлекающие «аналитические» свойства рассматриваемой функции (гладкость, выпуклость, d.c. [25], и др.). В данной ситуации нам представляется оправданным применить для решения описанной проблемы метаэвристические алгоритмы. В частности, на данном этапе реализована вычислительная схема имитации отжига [26]. Вторая особенность рассматриваемой задачи состоит в том, что далеко не все значения $\Phi(\cdot)$ необходимо досчитывать. Действительно, если Φ' – некоторое рекордное значение данной функции на данный момент времени, и для точки $\alpha^{\tilde{X}}$ можно сделать вывод, что $\Phi\left(\alpha^{\tilde{X}}\right) > \Phi'$, то очевидно, что дальнейшее вычисление значения $\Phi\left(\alpha^{\tilde{X}}\right)$ можно прервать и перейти к следующей точке, либо принять точку $\alpha^{\tilde{X}}$ за центр новой окрестности с некоторой вероятностью.

В соответствии со схемой имитации отжига [26], минимизация функции $\Phi(\cdot)$ рассматривается как итеративный процесс переходов между точками пространства поиска:

$$\alpha^0 \to \alpha^1 \to \ldots \to \alpha^i \to \ldots \to \alpha^*.$$

Переход от α^i к α^{i+1} осуществляется в два этапа. Сначала к α^i применяется вероятностное преобразование, результатом которого является точка $\tilde{\alpha}^i$ из некоторой окрестности α^i. Точка $\tilde{\alpha}^i$ становится точкой α^{i+1} с вероятностью, обозначаемой через $\Pr\left\{\tilde{\alpha}^i \to \alpha^{i+1} \mid \alpha^i\right\}$. Данная вероятность задается следующим образом:

$$\Pr\left\{\tilde{\alpha}^i \to \alpha^{i+1} \mid \alpha^i\right\} = \begin{cases} 1, \text{ if } \Phi\left(\tilde{\alpha}^i\right) < \Phi\left(\alpha^i\right) \\ \exp\left(-\dfrac{\Phi\left(\tilde{\alpha}^i\right) - \Phi\left(\alpha^i\right)}{T_i}\right), \text{ if } \Phi\left(\tilde{\alpha}^i\right) \geq \Phi\left(\alpha^i\right) \end{cases}$$

Изменение параметра T_i соответствует уменьшению «температуры кристаллизирующейся среды». Обычно полагают, что $T_i = Q \cdot T_{i-1}$, $i \geq 1$, где $Q \in (0,1)$. Процесс стартует при некотором начальном значении T_0 и продолжается до тех пор, пока «температура» не станет меньше заданного порогового значения $T_{\inf}$.

Описанная схема минимизации прогнозных функций была реализована в виде параллельного MPI-приложения, работающего на вычислительном кластере. Во время вычислений приложение отслеживает ситуации превышения рекордных значений прогнозной функции с последующим прерыванием соответствующего вычисления либо переходом к новой точке. В целом данный процесс работает в соответствии со схемой, описанной в [27] (используются неблокирующие обмены в MPI среде). Результаты вычислительных экспериментов приведены в разделе 5.

4. Проект добровольных распределенных вычислений SAT@home

4.1. Общие сведения о добровольных распределенных вычислениях

Здесь мы кратко коснемся общей идеологии добровольных вычислений. Первыми добровольными проектами были GIMPS и distributed.net, запущенные соответственно в 1996 и 1997 годах. Проект GIMPS был направлен на поиск чисел Мерсенна, а distributed.net на решение различных вариантов задачи криптоанализа шифра RC5. В 1996 году на одной из конференций по радиоастрономии 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. И если изначально для

создания добровольных проектов требовались ресурсы больших научных коллективов, то с использованием BOINC построение каждого нового проекта стало вполне по силам небольшим лабораториями и даже отдельным энтузиастам. Из 70 активных на данный момент проектов добровольных вычислений 65 построены на платформе BOINC.

Проект на платформе BOINC состоит из следующих основных частей: серверного ПО, веб-сайта и прикладного ПО. Серверное и прикладное ПО образуют распределенное приложение. Далее перечислены основные службы, входящие в серверное ПО проекта:

work_generator – создает задания для обработки;

validator – проверяет корректность присланных с ПК пользователей результатов, а также начисляет кредиты за корректные результаты;

assimilator – обрабатывает корректные результаты.

Для того чтобы подключиться к проекту, пользователь (доброволец) должен установить на свой ПК стандартный «BOINC-клиент». Это специальная программа, позволяющая подключать ПК пользователя к любым проектам на основе BOINC, осуществляющая обмен данными с сервером выбранного проекта и выделяющая ресурсы ПК пользователя для работы прикладного ПО. Для подключения пользователь должен указать URL сайта выбранного проекта и зарегистрироваться на нем (достаточно сообщить только адрес своей электронной почты). После этого вычисления на стороне пользователя происходят автоматически – BOINC-клиент скачивает с сервера проекта прикладное ПО (ориентируясь на тип ОС и процессора ПК пользователя) и задания для обработки. В BOINC-клиенте существует гибкая система настроек, позволяющая эффективно задействовать свободные ресурсы ПК и распределять их между проектами. После настройки BOINC-клиента пользователь становится полноправным участником выбранных им проектов.

Сайт проекта содержит следующую информацию: цели проводимых исследований, полученные результаты, список публикаций авторов проекта, производительность проекта, рейтинг лучших (в смысле количества набранных в проекте кредитов) участников и т.п. Также участнику доступен форум, на котором идет общение с разработчиками проекта.

При организации проекта следует учитывать ряд факторов, способствующих его активному развитию. 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. Участники могут объединяться в команды по разным признакам (национальному, региональному, пр.), которые также соревнуются между собой;
- при нахождении очередного объекта поиска информация об участнике, на ПК которого данный объект был найден, публикуется на сайте проекта;
- ощущение причастности к важным научным исследованиям – именно поэтому большой популярностью пользуются медицинские проекты, направленные на поиск новых лекарств (World Community Grid, Rosetta@home).

4.2. Проект добровольных распределенных вычислений SAT@home

Данный проект был запущен 29 сентября 2011 года. SAT@home [2] – совместный проект Института динамики систем и теории управления СО РАН и Института системного анализа РАН. При создании проекта была использована открытая платформа BOINC и пакет SZTAKI Desktop Grid [28]. Схема работы распределенного приложения проекта представлена на рисунке 2. 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.

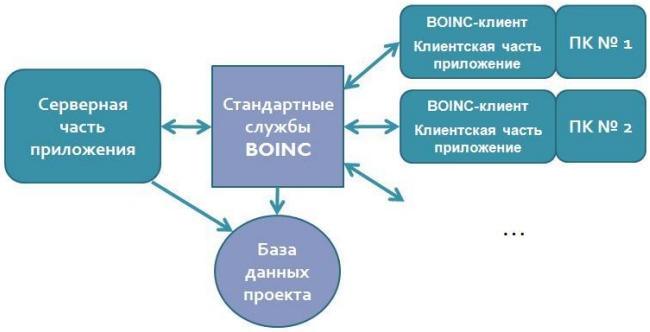

Рис. 2. Схема работы распределенного приложения проекта SAT@home

Формирование списка заданий осуществляется при помощи решателя PD-SAT [27]. Данный решатель, запущенный на вычислительном кластере, получает исходную SAT-задачу и находит для нее некоторое декомпозиционное множество в соответствии с описанными выше алгоритмами минимизации прогнозных функций. 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 решаемых в проекте SAT-задач. Полученные результаты BOINC-клиент отправляет на сервер. В результате решения в общем случае всех подзадач находится решение исходной SAT-задачи.

По состоянию на 12 сентября 2012 г. SAT@home имеет следующие характеристики:

- 2683 пользователей, 78 % иностранных;
- 9402 ПК, суммарно 37090 ядер процессоров, около 80 % под управлением ОС Windows;
- версии клиентского приложения: windows x86, linux x86, linux x64;
- средняя реальная производительность грида проекта 1.3 терафлопс, пиковая 4,3 терафлопс.

На рисунке 3 представлена динамика изменения реальной производительности грида проекта SAT@home (с 6 сентября 2011 года по 12 сентября 2012 года).

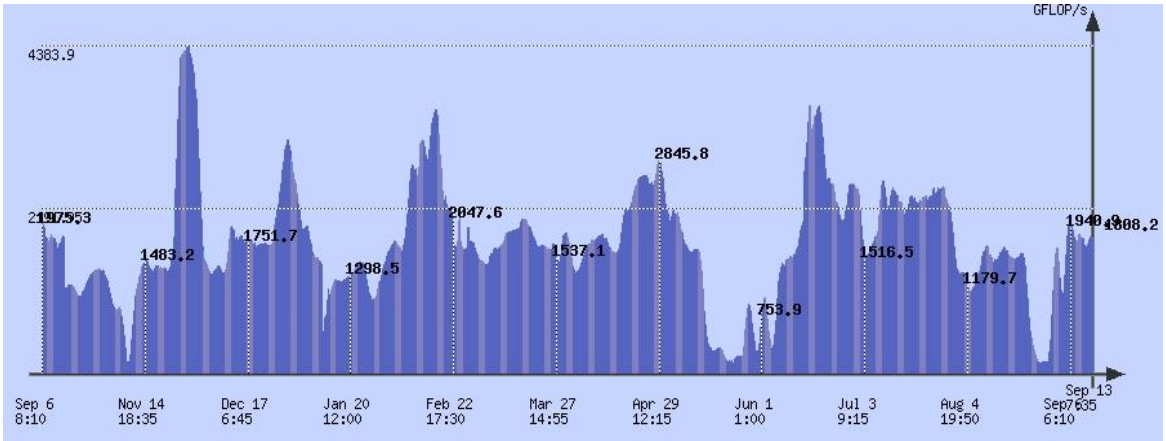

Рис. 3. Динамика изменения реальной производительности SAT@home (в гигафлопсах)

5. Вычислительные эксперименты

В данном разделе мы приводим результаты тестирования описанных выше процедур минимизации прогнозных функций, а также результаты решения реальных задач в проекте SAT@home. На рисунке 1 приведено декомпозиционное множество, использованное для решения задачи криптоанализа известного генератора поточного шифрования A5/1. Наиболее успешным методом криптоанализа данного генератора является т.н. «rainbow»-метод. Однако известные rainbow-таблицы [30] покрывают ключевое пространство A5/1 примерно на 88% и не дают результатов (при реалистичных предположениях на условия криптоанализа) для тестов, в которых используются оставшиеся 12% ключей. Для решения в рамках проекта SAT@home были построены 10 таких тестов [2] (раздел «найденные решения»). Все они были решены за полгода работы проекта (с 21.12.2011г. по 07.05.2012 г.).

Как уже отмечалось выше, декомпозиционное множество для SAT-задачи, кодирующей криптоанализ A5/1, фактически вычислялось «вручную». Использованные при этом соображения основаны на известных особенностях алгоритма A5/1. В частности, относительно некоторых переменных в кодирующей КНФ можно сделать вывод о их более сильном приоритете перед остальными при включении в декомпозиционное множество. Это обусловлено т.н. «функциональной семантикой» проблемы – в задачах обращения функций стартовым декомпозиционным множеством может быть множество всех переменных, кодирующих входные значения схемы, реализующей рассматриваемую функцию. Однако далеко не для всех комбинаторных задач можно делать подобные выводы. В этих ситуациях, конечно же, при построении декомпозиционных множеств необходимо использовать автоматические процедуры. Описанный выше метод минимизации прогнозных функций является такой процедурой.

При тестировании метода на адекватность необходимо сравнение выдаваемых им результатов с некоторыми эталонами. В качестве такого эталона мы рассмотрели декомпозиционное множество для SAT-задачи, кодирующей криптоанализ A5/1 (рис. 1). На рисунке 4 приведена структура декомпозиционного множества, найденного при помощи представленной выше процедуры минимизации прогнозных функций. При этом были использованы следующие параметры схемы имитации отжига: величина начальной «температуры» T_0 выбиралась равной 5% от значения прогнозной функции в начальной точке (выбранное случайным образом множество мощности 40 среди 64 булевых переменных, кодирующих вход генератора A5/1); конечная «температура» $T_{\text{inf}} = 30000$.

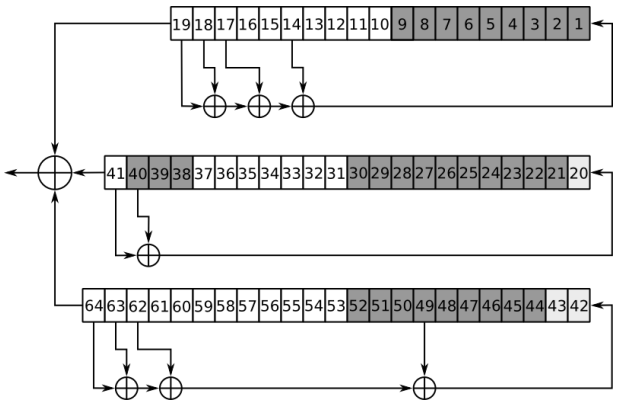

Рис. 4 Декомпозиционное множество, построенное автоматически (без привязки к особенностям исходной задачи) при помощи метода, описанного в настоящей работе.

Из рисунков 1 и 2 можно сделать вывод, что предложенная схема строит декомпозиционное множество, структура которого очень близка к структуре эталонного множества – метод находит множество той же мощности, что и эталонное, в котором лишь три переменные находятся «не на своем месте». Данный эффект наблюдался и в применении к другим криптографическим задачам (криптоанализ суммирующего и порогового генераторов). Это позволяет надеяться, что описанный подход позволит находить адекватные декомпозиционные множества в SAT-задачах, кодирующих не только задачи обращения функций, но и различные комбинаторные проблемы с невыраженной функциональной семантикой.

6. Заключение

В работе предложен новый подход к построению декомпозиционных множеств для крупноблочного распараллеливания SAT-задач. Показано, что декомпозиционное множество, построенное предложенным алгоритмом для SAT-задачи, кодирующей криптоанализ генератора A5/1, совсем незначительно отличается от известного эталонного множества. Разработанные алгоритмы предполагается использовать для решения различных комбинаторных задач в проекте распределенных вычислений SAT@home. В частности, предполагается использовать данный проект для поиска новых ортогональных систем латинских квадратов порядков 9 и 10.

СПИСОК ЛИТЕРАТУРЫ

1. Biere A., Heule V., van Maaren H., Walsh T. Handbook of Satisfiability. IOS Press, 2009.
2. Проект добровольных распределенных вычислений SAT@home: http://sat.isa.ru/pdsat/
3. Posypkin M., Semenov A., Zaikin O. Using BOINC desktop grid to solve large scale SAT problems // Computer Science Journal. 2012. Vol.13. №1. Pp. 25 – 34.
4. Заикин О.С., Посыпкин М.А., Семёнов А.А., Храпов Н.П. Организация добровольных вычислений на платформе BOINC на примере проектов OPTIMA@home и SAT@home // CAD/CAM/CAE Observer. 2012. №3(71). С. 87 – 92.
5. Список активных BOINC-проектов: http://boinc.berkeley.edu/projects.php
6. Cook S.A. The complexity of theorem-proving procedures // Third annual ACM symposium on Theory of computing, 1971, Ohio, USA. ACM. 1971. Pp. 151 – 159.
7. Сайт, посвященный задаче о булевой выполнимости: www.satlive.org
8. Davis M., Logemann G., Loveland D. A machine program for theorem proving // Communication of the ACM. 1962. V. 5, Issue 7. Pp. 394 – 397.
9. Marques-Silva J.P., Sakallah K.A. GRASP: A search algorithm for propositional satisfiability // IEEE Transactions on Computers. 1999. V. 48. N5. Pp. 506 – 521.
10. Prestwich S. CNF encodings. In Handbook of Satisfiability (editors: A. Biere, M.Heule, H. van Maaren, T. Walsh). 2009. IOS Press. P. 75 – 97.
11. Отпущенников И.В., Семенов А.А. Технология трансляции комбинаторных проблем в булевы уравнения // Прикладная дискретная математика. 2011. № 1. С. 96 – 115.
12. Цейтин Г.С. О сложности вывода в исчислении высказываний // Записки научных семинаров ЛОМИ АН СССР. 1968, т.8. С. 234 – 259.
13. Gil L., Flores P., Silveira L.M. PMSat: a parallel version of MiniSAT // Journal on Satisfiability, Boolean Modeling and Computation. 2009. V. 6. Pp. 71 – 98.
14. Schubert T., Lewis M., Becker B. PaMiraXT: Parallel SAT Solving with Threads and Message Passing // Journal on Satisfiability, Boolean Modeling and Computation. 2009. V. 6, Pp. 203 – 222.

15. Hamadi Y., Jabbour S., Sais L. ManySAT: a Parallel SAT Solver // Journal on Satisfiability, Boolean Modeling and Computation. 2009. V. 6. Pp. 245 – 262.
16. Ignatiev A., Semenov A. DPLL+ROBDD Derivation Applied to Inversion of Some Cryptographic Functions // LNCS. 2011. V. 6695. Pp. 76 – 89 (SAT-2011).
17. Schulz S., Blochinger W. Parallel SAT Solving on Peer-to-Peer Desktop Grids // Journal Of Grid Computing. 2010. V. 8, No. 3. P. 443 – 471.
18. Heule M., Kullmann O., Wieringa S., Biere A. Cube and Conquer: Guiding CDCL SAT Solvers by Lookaheads // In Proc. of 7th Intl. Haifa Verification Conference (HVC-11). 2011.
19. Hyvarinen A., Niemela I., Junttila T. Grid-Based SAT Solving with Iterative Partitioning and Clause Learning // LNCS. 2011. V. 6876. Pp. 385 – 399.
20. Посыпки М.А., Заикин О.С., Беспалов Д.В., Семенов А.А. Решение задач криптоанализа поточных шифров в распределенных вычислительных средах // Труды ИСА РАН. 2009. №46. С. 119 – 137.
21. Semenov A., Zaikin O., Bespalov D., Posypkin M. Parallel logical cryptanalysis of the generator A5/1 in BNB-Grid system // LNCS. 2011. V. 6873. P. 473 – 483.
22. Заикин О.С., Семенов А.А. Технология крупноблочного параллелизма в SAT-задачах // Проблемы управления. 2008. № 1. С. 43 – 50.
23. Семенов А.А. Декомпозиционные представления логических уравнений в задачах обращения дискретных функций // Известия РАН. Теория и системы управления. 2009. №5. С. 47 – 61.
24. Biryukov A., Shamir A., Wagner D. Real time cryptanalysis of A5/1 on a PC // LNCS. 2001. V. 1978. Pp. 1 – 18.
25. Стрекаловский А.С. Элементы невыпуклой оптимизации. – Новосибирск, «Наука». 2003. 355 с.
26. Kirkpatrick S., Gelatt C.D., Vecchi M.P. Optimization by simulated annealing // Science. 1983. V. 220. Pp. 671 – 680.
27. Заикин О.С. Реализация процедур прогнозирования трудоемкости параллельного решения SAT-задач // Вестник УГАТУ. 2010. Т. 14, № 4. С. 210 – 220.
28. 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. V. 7, No. 4. Pp. 439 – 461.
29. SAT-решатель minisat: http://minisat.se/MiniSat.html
30. A5/1 Cracking project: http://reflextor.com/trac/a51/wiki