Поиск пар ортогональных диагональных латинских квадратов порядка 10 в проекте добровольных распределенных вычислений SAT@home*

О.С. Заикин [1], С.Е. Кочемазов [1], И.Д. Белоречев[2]
Институт динамики систем и теории управления СО РАН[1],
Институт математики, экономики и информатики ИГУ[2]

В статье рассматривается подход к решению задач поиска систем ортогональных латинских квадратов, основанный на сведении этих задач к проблеме булевой выполнимости. Была построена соответствующая кодировка для задачи поиска пар ортогональных диагональных латинских квадратов порядка 10. С помощью построенной кодировки в проекте добровольных распределенных вычислений SAT@home были найдены 17 новых пар. На основе 17 найденных пар, а также 3 ранее известных пар, были построены псевдотройки диагональных латинских квадратов порядка 10. Построение псевдотроек было осуществлено на вычислительном кластере, для этого была сделана параллельная реализация алгоритма генерации диагональных латинских квадратов порядка 10.

1. Введение

Комбинаторные задачи, связанные с латинскими квадратами, интересуют математиков со времен Леонарда Эйлера (см., например, обзорную статью [1]). Латинский квадрат порядка n – это квадратная таблица размера $n \times n$, заполненная элементами некоторого множества $M, |M| = n$ таким образом, что в каждой строке и в каждом столбце таблицы каждый элемент из M встречается в точности один раз. Изначально Леонард Эйлер в качестве множества M использовал множество букв латинского алфавита, именно поэтому такие квадраты были названы латинскими. В дальнейшем в настоящей статье в качестве M будет использовано множество $\{0, \dots, n-1\}$.

Пара латинских квадратов одинакового порядка называется ортогональной, если различны все упорядоченные пары чисел (a, b), где a – число в некоторой ячейке первого латинского квадрата, а b – число в ячейке с тем же номером второго латинского квадрата. Если имеется набор из m различных латинских квадратов, любая пара которых ортогональна, то говорят о системе из m попарно ортогональных латинских квадратов. Одной из самых известных нерешенных задач, касающейся латинских квадратов, является следующая: определить, существует ли система из трех попарно ортогональных латинских квадратов порядка 10 [2].

Задачи поиска комбинаторных структур можно решать различными подходами. В настоящей статье развивается SAT-подход к решению таких задач. SAT-подход состоит в сведении исходной задачи к проблеме булевой выполнимости (SAT [3]). Привлекательность SAT-подхода обусловлена тем, что к проблеме булевой выполнимости можно эффективно свести обширный класс задач криптографии, биоинформатики и поиска комбинаторных структур [4]. Все известные алгоритмы решения SAT-задач экспоненциальны в худшем случае, т.к. SAT является NP-трудной задачей. Несмотря на это, современные SAT-решатели (в том числе параллельные и распределенные) успешно справляются с решением целого ряда задач из упомянутых выше областей. Большинство из современных быстрых SAT-решателей основано на алгоритме Conflict-Driven Clause Learning (CDCL), базовые принципы которого рассмотрены в обзорной статье [5].

*Работа была поддержана РФФИ (гранты № 14-07-00403-а, 14-07-31172-мол-а и 15-07-07891-а) и Советом по грантам Президента РФ для государственной поддержки ведущих научных школ (НШ-5007.2014.9).

Для использования SAT-подхода необходимо перейти от исходной постановки к булевому уравнению вида «КНФ=1» (КНФ – конъюнктивная нормальная форма). Такой переход называется пропозициональным кодированием исходной проблемы. Попытки применить SAT-подход к задаче поиска систем ортогональных латинских квадратов регулярно предпринимаются с середины 90-х годов XX века. Много полезной информации о применении SAT-решателей к поиску различных систем ортогональных латинских квадратов содержится в обзорной статье [6]. Кроме всего прочего, автор статьи [6] пробовал решить упомянутую выше задачу поиска тройки попарно ортогональных квадратов порядка 10 с использованием SAT-решателя PSATO в течение более 10 лет в грид-системе из 40 персональных компьютеров (окончательный ответ так и не был получен).

Кратко опишем структуру статьи. Во втором разделе будет представлена техника пропозиционального кодирования задач поиска систем ортогональных латинских квадратов, а также результат применения этой техники к построению кодировки задачи поиска пар ортогональных диагональных латинских квадратов порядка 10. В третьем разделе будет описан эксперимент по поиску пар ортогональных диагональных латинских квадратов порядка 10 в проекте добровольных распределенных вычислений SAT@home. В четвертом разделе приведены результаты использования вычислительного кластера для поиска псевдотроек ортогональных диагональных латинских квадратов порядка 10.

2. Пропозициональное кодирование задач поиска систем ортогональных латинских квадратов

В силу того, что система попарно ортогональных латинских квадратов как комбинаторная структура эквивалентна ряду других комбинаторных структур, задачу поиска такой системы можно переформулировать через эквивалентные объекты. Это позволяет использовать для решения задачи различные пропозициональные кодировки. Так, например, можно кодировать задачу поиска пары ортогональных латинских квадратов при помощи ортогональных массивов или записывать ее через трансверсали. Вообще говоря, даже если использовать пропозициональную кодировку, основанную на латинских квадратах, возможны различные варианты формулирования необходимых условий. В описанных далее вычислительных экспериментах мы использовали так называемую «наивную» кодировку задачи поиска набора ортогональных латинских квадратов, аналогичную приведенной, например, в [6] и [7].

Приведем краткое описание данной кодировки. Рассматриваем две матрицы $A = \|a_{ij}\|$ и $B = \|b_{ij}\|$, $i, j = \overline{1, \dots, n}$. Содержимое каждой ячейки любой из матриц кодируется n булевыми переменными. Для кодирования всей матрицы, таким образом, требуется n^3 булевых переменных. Будем использовать запись $x(i, j, k)$ и $y(i, j, k)$ для обозначения переменных, кодирующих элементы матриц A и B, соответственно. При этом переменная $x(i, j, k)$ принимает значение «истина» тогда и только тогда, когда в ячейке, находящейся в строке с номером i и столбце с номером j, стоит число k. Чтобы матрицы A и B представляли латинские квадраты необходимо наложить на соответствующие им переменные ряд условий, рассмотренные далее на примере матрицы A. Эти условия естественным образом записываются в виде конъюнкций дизъюнктов.

В каждой ячейке матрицы стоит ровно одно число от 1 до n:

- $\wedge_{i=1}^{n} \wedge_{j=1}^{n} \vee_{k=1}^{n} x(i, j, k)$;
- $\wedge_{i=1}^{n} \wedge_{j=1}^{n} \wedge_{k=1}^{n-1} \wedge_{r=k+1}^{n} \left(\neg x(i, j, k) \vee \neg x(i, j, r)\right)$.

Каждое число от 1 до n появляется в каждой строке ровно один раз:

- $\wedge_{j=1}^{n} \wedge_{k=1}^{n} \vee_{i=1}^{n} x(i, j, k)$;
- $\wedge_{j=1}^{n} \wedge_{k=1}^{n} \wedge_{i=1}^{n-1} \wedge_{r=i+1}^{n} \left(\neg x(i, j, k) \vee \neg x(r, j, k)\right)$.

Каждое число от 1 до n появляется в каждом столбце ровно один раз:

- $\wedge_{i=1}^{n} \wedge_{k=1}^{n} \vee_{j=1}^{n} x(i, j, k)$;
- $\wedge_{i=1}^{n} \wedge_{k=1}^{n} \wedge_{j=1}^{n-1} \wedge_{r=j+1}^{n} \left(\neg x(i, j, k) \vee \neg x(i, r, k)\right)$.

Аналогичным образом записываются условия, кодирующие матрицу B. После этого необходимо закодировать условие ортогональности. Например, это можно сделать следующим образом:

- $\wedge_{i=1}^{n}\wedge_{j=1}^{n}\wedge_{k=1}^{n}\wedge_{p=1}^{n}\wedge_{q=1}^{n}\wedge_{r=1}^{n}\left(\neg x(i,j,k)\vee\neg y(i,j,k)\vee\neg x(p,q,r)\vee\neg y(p,q,r)\right)$.

В результате применения всех описанных выше условий для задачи поиска пар ортогональных латинских квадратов порядка 10 мы получили КНФ, состоящую из 432600 дизъюнктов над множеством из 2000 булевых переменных. Соответствующий файл с КНФ в формате DIMACS занимает 9,5 мегабайт.

Рассмотрим задачу поиска пар ортогональных диагональных латинских квадратов. Латинский квадрат является диагональным, если для него справедливо следующее условие: главная и побочная диагонали содержат все числа от 0 до $n-1$, где n – это порядок латинского квадрата. Таким образом, условие уникальности элементов накладывается не только на строки и столбцы квадрата, но и на главную и побочную диагонали. Известно, что пары ортогональных диагональных латинских квадратов порядка 10 существуют – в 1992 году была опубликована статья [8], в которой были представлены три такие пары. В открытых источниках нам не удалось найти другие пары, кроме этих трех. Поэтому, на наш взгляд, интересным представляется поиск новых пар ортогональных диагональных латинских квадратов порядка 10. Решению именно этой задачи, в основном, посвящена настоящая статья. Для получения пропозициональной кодировки задачи поиска систем таких квадратов необходимо добавить в описанную выше кодировку дизъюнкты, соответствующие условию диагональности. Полученная в результате КНФ состоит из 2000 переменных и 434440 дизъюнктов, файл с КНФ в формате DIMACS занимает 10 мегабайт.

3. Поиск систем ортогональных латинских квадратов в проекте добровольных распределенных вычислений SAT@home

Задачи поиска систем ортогональных латинских квадратов при помощи SAT-подхода хорошо подходят для организации масштабных экспериментов в распределенных вычислительных средах, в частности, в проектах добровольных распределенных вычислений. Это объясняется тем, что SAT-задачи сами по себе допускают естественные стратегии крупноблочного распараллеливания. Авторами статьи в сотрудничестве с коллегами из ИППИ РАН был разработан проект SAT@home [9–11], функционирующий с сентября 2011 года. Данный проект предназначен для решения трудных экземпляров SAT-задач из различных предметных областей. При создании проекта была использована открытая платформа BOINC [12, 13]. По состоянию на 23 ноября 2014 года к проекту подключено 3243 активно работающих ПК участников со всего мира, обеспечивающих среднюю производительность около 4.4 TFLOPs. Схема решения SAT-задач с помощью проекта SAT@home представлена на **Рис. 1**. Предварительно осуществляется поиск «хорошей» декомпозиции SAT-задачи с помощью метода Монте-Карло на вычислительном кластере [10, 14]. Необходимость использования кластера исходит из того, что на данном этапе используются интенсивные межпроцессорные обмены.

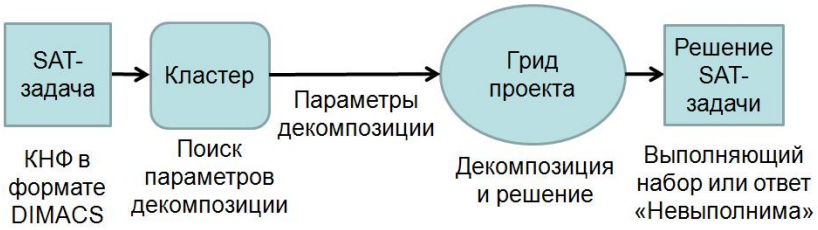

Рис. 1. Схема решения SAT-задач с помощью проекта SAT@home

В мае 2012 года в проекте SAT@home был успешно завершен полугодовой эксперимент, направленный на решение 10 задач криптоанализа генератора ключевого потока A5/1 [10], которые не решаются с помощью известных rainbow-таблиц [15]. В каждой задаче нужно было найти неизвестное начальное заполнение регистров генератора.

На следующем этапе в SAT@home был запущен поиск пар ортогональных диагональных латинских квадратов порядка 10 с использованием пропозициональной кодировки, описанной в предыдущем разделе. В качестве ядра клиентского приложения (которое запускается на ПК участников) использовался CDCL-решатель MiniSat 2.2 [16], в который были внесены небольшие изменения, направленные на уменьшение объема используемой оперативной памяти. Первая строка первого квадрата была изначально зафиксирована и равнялась «0 1 2 3 4 5 6 7 8 9». Это было сделано потому, что любую пару ортогональных латинских квадратов путем перестановок, не нарушающих условия ортогональности и диагональности, можно преобразовать к паре, в которой первая строка первого квадрата будет именно такой. Выбор способа декомпозиции SAT-задач – нетривиальная проблема, в решении которой нужно использовать свойства исходных постановок [10, 14]. Декомпозиция задачи поиска пар ортогональных диагональных латинских квадратов была осуществлена следующим образом. Варьировались значения первых 8 ячеек второй и третьей строки первого квадрата. Всего оказалось около 230 миллиардов возможных вариантов значений этих 16 ячеек, не нарушающих условие, что квадрат является диагональным латинским квадратом. Было решено сформировать для решения в SAT@home первые 20 миллионов подзадач из 230 миллиардов (т.е. всего около 0.0087 % описанного выше пространства поиска). В итоге каждая подзадача формировалась в результате подстановки в первый квадрат 8 первых ячеек второй и третьей строки (при фиксированной первой строке). Значения остальных 74 ячеек первого квадрата и всех 100 ячеек второго квадрата были неизвестны – их должен был найти SAT-решатель. На каждую подзадачу был установлен лимит в 2600 рестартов решателя MiniSat 2.2, что примерно соответствует 5 минутам работы одного ядра процессора Intel Core 2 Duo E8400. При достижении лимита вычисления прерывались. Каждое задание, которое получал участник SAT@home, состояло из 20 таких подзадач. Формирование именно такого количества подзадач в задании было продиктовано необходимостью обеспечить в задании около 2 часов работы на одном ядре процессора (такая продолжительность выполнения заданий хорошо подходит для BOINC-проектов). На обработку 20 миллионов подзадач (в виде 1 миллиона заданий), сгенерированных для данного эксперимента, потребовалось около 9 месяцев работы проекта (с сентября 2012 года по май 2013 года). Вычисления практически для всех подзадач были прерваны при достижении лимита, но решение 17 подзадач закончилось успешно – в результате были найдены 17 новых пар ортогональных диагональных латинских квадратов порядка 10 (мы сравнивали их с тремя ранее известными парами квадратов из статьи [8]). Все найденные пары выложены на сайте проекта SAT@home [11] в разделе «Найденные решения». На **Рис. 2** приведена первая пара ортогональных диагональных латинских квадратов порядка 10, найденная в проекте SAT@home.

$$\begin{bmatrix} 0&1&2&3&4&5&6&7&8&9\\ 1&2&0&4&3&7&9&8&5&6\\ 7&3&5&9&0&4&8&6&2&1\\ 3&5&6&8&9&0&4&1&7&2\\ 4&9&7&2&6&8&1&5&0&3\\ 5&8&4&6&7&1&3&2&9&0\\ 8&4&9&1&2&3&7&0&6&5\\ 6&7&3&0&1&2&5&9&4&8\\ 9&0&1&5&8&6&2&4&3&7\\ 2&6&8&7&5&9&0&3&1&4 \end{bmatrix} \quad \begin{bmatrix} 0&1&2&3&4&5&6&7&8&9\\ 7&5&1&9&2&8&0&4&6&3\\ 1&0&3&4&6&7&5&2&9&8\\ 9&8&4&7&5&2&1&0&3&6\\ 6&7&9&0&8&3&2&1&5&4\\ 4&6&5&1&0&9&8&3&2&7\\ 2&3&8&5&1&6&4&9&7&0\\ 5&2&7&8&3&4&9&6&0&1\\ 3&4&6&2&9&0&7&8&1&5\\ 8&9&0&6&7&1&3&5&4&2 \end{bmatrix}$$

Рис. 2. Первая пара ортогональных диагональных латинских квадратов порядка 10, найденная в проекте SAT@home

Как уже было отмечено в предыдущем разделе, есть много различных вариантов формирования пропозициональной кодировки задачи поиска пар ортогональных диагональных латинских квадратов. При этом встает вопрос сравнения различных кодировок по эффективности. Как показала практика, количество переменных и дизъюнктов в КНФ не всегда адекватно указывает на скорость ее обработки с помощью SAT-решателей. Мы предполагаем в ближайшем будущем использовать найденные пары ортогональных диагональных латинских квадратов (а также ранее известные пары) для оценки эффективности возможных кодировок

рассмотренной задачи. Сравнение можно проводить по набору КНФ (каждая КНФ соответствует известной паре ортогональных диагональных латинских квадратов порядка 10), в каждой из которых подставлено правильное означивание нескольких строк первого квадрата из соответствующей пары. Также с помощью такого тестового набора задач можно выбрать наиболее подходящие для этой предметной области SAT-решатели.

4. Поиск псевдотроек ортогональных диагональных латинских квадратов на вычислительном кластере

Большой научный интерес представляет знаменитая задача о существовании тройки попарно ортогональных латинских квадратов порядка 10. Однако, данная задача, как уже отмечалось выше, крайне трудна. Поэтому активно исследуется задача нахождения таких троек латинских квадратов, что условие ортогональности выполняется не для всех пар квадратов (например, только для одной или для двух пар из трех возможных). Далее подобные системы латинских квадратов называются *псевдотройками*. Главная характеристика псевдотроек, которая будет рассматриваться далее – это количество упорядоченных пар элементов, по которым выполняется условие ортогональности одновременно для всех трех пар квадратов (A и B, A и C, B и C), составляющих псевдотройку. На данный момент рекордный по этому показателю набор – это опубликованная в статье [17] псевдотройка с характеристикой 91, при этом для двух пар квадратов из трех условие ортогональности выполняется полностью. На **Рис. 3** представлена данная псевдотройка, в ней квадрат A ортогонален квадратам B и C, при этом квадраты B и C ортогональны только по 91 упорядоченной паре элементов из 100 возможных.

$$A=\begin{bmatrix}0&8&9&7&5&6&4&2&3&1\\9&1&4&6&2&7&3&8&0&5\\7&4&2&5&1&3&8&6&9&0\\8&6&5&3&9&2&1&0&4&7\\6&2&1&8&4&0&9&5&7&3\\4&9&3&2&7&5&0&1&6&8\\5&3&7&1&0&8&6&9&2&4\\3&5&0&9&8&4&2&7&1&6\\1&7&6&0&3&9&5&4&8&2\\2&0&8&4&6&1&7&3&5&9\end{bmatrix}\ B=\begin{bmatrix}0&7&8&9&1&2&3&4&5&6\\9&0&6&1&8&3&2&5&4&7\\7&2&0&4&3&9&1&8&6&5\\8&5&3&0&2&1&7&6&9&4\\6&9&5&3&0&7&4&2&1&8\\4&1&7&6&5&0&8&9&3&2\\5&4&2&8&9&6&0&3&7&1\\3&6&1&7&4&8&5&0&2&9\\1&8&4&2&6&5&9&7&0&3\\2&3&9&5&7&4&6&1&8&0\end{bmatrix}\ C=\begin{bmatrix}0&7&8&9&1&2&3&4&5&6\\6&4&2&8&9&5&1&3&7&0\\4&9&5&3&2&7&6&0&1&8\\5&1&7&6&4&3&8&9&0&2\\3&2&9&0&7&1&5&6&8&4\\1&0&3&7&6&8&2&5&4&9\\2&8&0&1&3&4&9&7&6&5\\9&5&4&2&8&6&0&1&3&7\\7&3&6&5&0&9&4&8&2&1\\8&6&1&4&5&0&7&2&9&3\end{bmatrix}$$

Рис. 3. Рекордная псевдотройка ортогональных латинских квадратов порядка 10 из статьи [16]

Нами была предпринята попытка улучшения данного рекорда, однако на данный момент сделать это не удалось. Нами была рассмотрена другая задача, а именно – задача поиска псевдотроек ортогональных диагональных латинских квадратов порядка 10, по аналогии с описанной выше задачей. В явном виде такая постановка задачи нами в открытых источниках найдена не была. Однако в неявном виде данная постановка присутствует в статье [8]. В этой статье приведены 3 пары ортогональных диагональных латинских квадратов порядка 10, при этом в первой и второй паре фигурирует один и тот же квадрат (второй в обеих парах). Упомянутые наборы квадратов изображены на **Рис. 4** и **Рис. 5** соответственно.

$$A=\begin{bmatrix}0&9&4&6&1&7&5&8&2&3\\7&1&9&4&5&3&8&0&6&2\\4&6&2&8&3&1&7&5&9&0\\6&0&7&3&2&8&4&9&1&5\\5&3&6&7&4&2&9&1&0&8\\8&4&1&2&9&5&0&6&3&7\\2&5&3&0&8&9&6&4&7&1\\3&2&8&9&0&4&1&7&5&6\\9&7&5&1&6&0&3&2&8&4\\1&8&0&5&7&6&2&3&4&9\end{bmatrix}\ B=\begin{bmatrix}0&8&5&1&7&3&4&6&9&2\\5&1&7&2&9&8&0&3&4&6\\1&7&2&9&5&6&8&0&3&4\\9&6&4&3&0&2&7&1&5&8\\3&0&8&6&4&1&5&9&2&7\\4&3&0&8&6&5&9&2&7&1\\7&2&9&5&1&4&6&8&0&3\\6&4&3&0&8&9&2&7&1&5\\2&9&6&4&3&7&1&5&8&0\\8&5&1&7&2&0&3&4&6&9\end{bmatrix}$$

Рис. 4. Первая пара ортогональных диагональных латинских квадратов порядка 10 из статьи [8]

$$A = \begin{bmatrix} 0 & 4 & 1 & 9 & 8 & 2 & 7 & 3 & 5 & 6 \\ 3 & 1 & 6 & 8 & 2 & 9 & 4 & 5 & 0 & 7 \\ 6 & 5 & 2 & 4 & 9 & 0 & 3 & 8 & 7 & 1 \\ 1 & 8 & 5 & 3 & 7 & 4 & 9 & 0 & 6 & 2 \\ 9 & 2 & 0 & 5 & 4 & 7 & 8 & 6 & 1 & 3 \\ 8 & 6 & 3 & 7 & 1 & 5 & 0 & 9 & 2 & 4 \\ 4 & 0 & 7 & 2 & 5 & 3 & 6 & 1 & 9 & 8 \\ 2 & 9 & 4 & 1 & 6 & 8 & 5 & 7 & 3 & 0 \\ 7 & 3 & 9 & 6 & 0 & 1 & 2 & 4 & 8 & 5 \\ 5 & 7 & 8 & 0 & 3 & 6 & 1 & 2 & 4 & 9 \end{bmatrix} B = \begin{bmatrix} 0 & 8 & 5 & 1 & 7 & 3 & 4 & 6 & 9 & 2 \\ 5 & 1 & 7 & 2 & 9 & 8 & 0 & 3 & 4 & 6 \\ 1 & 7 & 2 & 9 & 5 & 6 & 8 & 0 & 3 & 4 \\ 9 & 6 & 4 & 3 & 0 & 2 & 7 & 1 & 5 & 8 \\ 3 & 0 & 8 & 6 & 4 & 1 & 5 & 9 & 2 & 7 \\ 4 & 3 & 0 & 8 & 6 & 5 & 9 & 2 & 7 & 1 \\ 7 & 2 & 9 & 5 & 1 & 4 & 6 & 8 & 0 & 3 \\ 6 & 4 & 3 & 0 & 8 & 9 & 2 & 7 & 1 & 5 \\ 2 & 9 & 6 & 4 & 3 & 7 & 1 & 5 & 8 & 0 \\ 8 & 5 & 1 & 7 & 2 & 0 & 3 & 4 & 6 & 9 \end{bmatrix}$$

Рис. 5. Вторая пара ортогональных диагональных латинских квадратов порядка 10 из статьи [8].

На основе этих двух пар ортогональных диагональных латинских квадратов порядка 10 можно естественным образом построить псевдотройку порядка 10 (см. **Рис. 6**).

$$A = \begin{bmatrix} 0 & 8 & 5 & 1 & 7 & 3 & 4 & 6 & 9 & 2 \\ 5 & 1 & 7 & 2 & 9 & 8 & 0 & 3 & 4 & 6 \\ 1 & 7 & 2 & 9 & 5 & 6 & 8 & 0 & 3 & 4 \\ 9 & 6 & 4 & 3 & 0 & 2 & 7 & 1 & 5 & 8 \\ 3 & 0 & 8 & 6 & 4 & 1 & 5 & 9 & 2 & 7 \\ 4 & 3 & 0 & 8 & 6 & 5 & 9 & 2 & 7 & 1 \\ 7 & 2 & 9 & 5 & 1 & 4 & 6 & 8 & 0 & 3 \\ 6 & 4 & 3 & 0 & 8 & 9 & 2 & 7 & 1 & 5 \\ 2 & 9 & 6 & 4 & 3 & 7 & 1 & 5 & 8 & 0 \\ 8 & 5 & 1 & 7 & 2 & 0 & 3 & 4 & 6 & 9 \end{bmatrix} B = \begin{bmatrix} 0 & 9 & 4 & 6 & 1 & 7 & 5 & 8 & 2 & 3 \\ 7 & 1 & 9 & 4 & 5 & 3 & 8 & 0 & 6 & 2 \\ 4 & 6 & 2 & 8 & 3 & 1 & 7 & 5 & 9 & 0 \\ 6 & 0 & 7 & 3 & 2 & 8 & 4 & 9 & 1 & 5 \\ 5 & 3 & 6 & 7 & 4 & 2 & 9 & 1 & 0 & 8 \\ 8 & 4 & 1 & 2 & 9 & 5 & 0 & 6 & 3 & 7 \\ 2 & 5 & 3 & 0 & 8 & 9 & 6 & 4 & 7 & 1 \\ 3 & 2 & 8 & 9 & 0 & 4 & 1 & 7 & 5 & 6 \\ 9 & 7 & 5 & 1 & 6 & 0 & 3 & 2 & 8 & 4 \\ 1 & 8 & 0 & 5 & 7 & 6 & 2 & 3 & 4 & 9 \end{bmatrix} C = \begin{bmatrix} 0 & 4 & 1 & 9 & 8 & 2 & 7 & 3 & 5 & 6 \\ 3 & 1 & 6 & 8 & 2 & 9 & 4 & 5 & 0 & 7 \\ 6 & 5 & 2 & 4 & 9 & 0 & 3 & 8 & 7 & 1 \\ 1 & 8 & 5 & 3 & 7 & 4 & 9 & 0 & 6 & 2 \\ 9 & 2 & 0 & 5 & 4 & 7 & 8 & 6 & 1 & 3 \\ 8 & 6 & 3 & 7 & 1 & 5 & 0 & 9 & 2 & 4 \\ 4 & 0 & 7 & 2 & 5 & 3 & 6 & 1 & 9 & 8 \\ 2 & 9 & 4 & 1 & 6 & 8 & 5 & 7 & 3 & 0 \\ 7 & 3 & 9 & 6 & 0 & 1 & 2 & 4 & 8 & 5 \\ 5 & 7 & 8 & 0 & 3 & 6 & 1 & 2 & 4 & 9 \end{bmatrix}$$

Рис. 6. Псевдотройка ортогональных диагональных латинских квадратов порядка 10, которую можно построить на основе первых двух пар квадратов из статьи [8].

Нетрудно посчитать, что характеристика такой псевдотройки равна 60. Напомним, что под характеристикой псевдотройки понимается число упорядоченных пар элементов, по которым одновременно выполняется условие ортогональности для всех трех пар квадратов псевдотройки. На **Рис. 7** представлены соответствующие 60 упорядоченных пар элементов для данной псевдотройки.

$$\begin{bmatrix} 00 & 01 & 02 & - & - & 05 & 06 & - & 08 & - \\ 10 & 11 & - & 13 & - & 15 & 16 & - & 18 & - \\ - & 21 & 22 & - & 24 & 25 & - & 27 & - & 29 \\ - & - & 32 & 33 & - & - & 36 & 37 & - & 39 \\ - & 41 & - & - & 44 & 45 & 46 & - & 48 & 49 \\ 50 & - & 52 & 53 & - & 55 & - & 57 & 58 & 59 \\ 60 & 61 & - & - & - & 65 & 66 & - & - & 69 \\ - & - & 72 & 73 & 74 & 75 & - & 77 & - & 79 \\ - & - & - & 83 & 84 & 85 & - & 87 & 88 & - \\ 90 & 91 & - & 93 & 94 & - & 96 & 97 & 98 & 99 \end{bmatrix}$$

Рис. 7. Набор из 60 упорядоченных пар элементов, по которым ортогональны все три пары квадратов псевдотройки, изображенной на **Рис. 6**

Мы рассмотрели следующую постановку задачи: найти псевдотройку ортогональных диагональных латинских квадратов порядка 10 с характеристикой больше, чем у псевдотройки из статьи [8]. Для решения поставленной задачи было решено использовать пары ортогональных диагональных латинских квадратов порядка 10 (3 пары из статьи [8] и 17 пар, найденных в результате эксперимента, описанного в предыдущем разделе). Идея состоит в генерации диагональных латинских квадратов порядка 10 с последующим формированием двадцати псевдотроек на основе известных пар и каждого сгенерированного квадрата. Для каждой псевдотройки можно быстро вычислить ее характеристику. Таким образом, можно выбрать лучшую псевдотройку (из двадцати) для конкретного сгенерированного квадрата, а также лучшую псевдотройку для всех сгенерированных квадратов.

Для генерации диагональных латинских квадратов был использован алгоритм поиска с возвратом. Основная идея этого алгоритма заключается в последовательной подстановке допустимых значений в ячейки таблицы, которая изначально пуста. Заполнение происходит слева направо сверху вниз по строкам таблицы. Поиск завершается, если найдено заполнение таблицы, которое соответствует некоторому диагональному латинскому квадрату. После подстановки каждого нового значения в некоторую ячейку осуществляется проверка, нарушает ли текущее заполнение таблицы условия, накладываемые на элементы в диагональном латинском квадрате. Если заполнение не прошло проверку, происходит возврат до ближайшей ячейки, элемент которой нарушает условия, после чего этой ячейке назначается другое допустимое значение. В алгоритме хранится история уже использованных неудачных значений для каждой ячейки таблицы. Это нужно для того, чтобы исключить повтор вариантов, которые приводят к недопустимым вариантам заполнений. Если для некоторой ячейки требуется поменять значение, но исчерпано множество допустимых вариантов, то происходит возврат на предыдущую ячейку.

Ключевой особенностью описанного алгоритма является то, что для увеличения количества сгенерированных диагональных латинских квадратов за единицу времени следует прерывать текущий поиск после достижения некоторого лимита времени. Это связано с тем, что при выборе «неудачных» значений в первых строках таблицы алгоритму может потребоваться очень большое время для возврата к этим начальным значениям. Если же начальные значения выбраны «удачно», то искомый квадрат может быть найден очень быстро. Опытным путем было установлено, что наиболее подходящий лимит по времени для программы, реализующей данный алгоритм – это 0.001 секунды. При этом фактически этот временной порог лишь немного превышает минимальное время, необходимое для «достройки» диагонального латинского квадрата при «удачно» выбранных начальных значениях.

На основе описанного алгоритма был создана MPI программа [18] на языке программирования C++. Управляющий процесс этой программы получает сообщения обо всех рекордных псевдотройках, найденных в процессе работы параллельной программы. Вычислительные процессы генерируют диагональные латинские квадраты порядка 10 с использованием приведенного выше алгоритма. При нахождении очередной рекордной псевдотройки вычислительный процесс отправляет соответствующее сообщение на управляющий процесс.

Созданная MPI программа была запущена на 1 сутки на 30 узлах вычислительного кластера «Академик В.М. Матросов» Иркутского суперкомпьютерного центра СО РАН[1]. На этих 30 узлах было задействовано 60 16-ядерных процессоров AMD Opteron 6276, т.е. суммарно 960 процессорных ядер. В результате работы MPI программы была найдена псевдотройка с характеристикой 62, что является рекордом в сравнении с рассмотренной выше псевдотройкой из статьи [8]. Рекордная псевдотройка была найдена через 7 часов 20 минут после старта работы программы. Всего же за сутки было сгенерировано около 20 миллионов разных диагональных латинских квадратов порядка 10, соответственно было сформировано около 400 миллионов псевдотроек. Описанные результаты аргументирует применение для рассмотренной задачи высокопроизводительных вычислений, ведь на обычном ПК для проведения таких расчетов потребовалось бы несколько месяцев. На **Рис. 8** представлена найденная нами рекордная псевдотройка, а на **Рис. 9** представлены соответствующие ей 62 упорядоченные пары элементов. Отметим, что данная псевдотройка основана на одной из 17 пар ортогональных диагональных латинских квадратов порядка 10, найденных в проекте SAT@home.

[1] http://hpc.icc.ru/

$$A = \begin{bmatrix} 0&1&2&3&4&5&6&7&8&9 \\ 1&2&0&4&6&7&9&5&3&8 \\ 6&9&8&5&1&3&2&0&4&7 \\ 9&0&3&1&2&6&5&8&7&4 \\ 3&4&5&0&7&1&8&9&6&2 \\ 7&3&9&2&8&4&0&1&5&6 \\ 8&5&6&7&9&2&3&4&0&1 \\ 5&7&4&9&3&8&1&6&2&0 \\ 4&6&1&8&5&0&7&2&9&3 \\ 2&8&7&6&0&9&4&3&1&5 \end{bmatrix} B = \begin{bmatrix} 0&1&2&3&4&5&6&7&8&9 \\ 6&7&4&5&9&3&0&8&1&2 \\ 7&6&1&0&3&4&8&2&9&5 \\ 8&5&7&9&1&2&3&6&4&0 \\ 9&2&6&1&8&7&5&4&0&3 \\ 2&8&3&4&0&6&9&5&7&1 \\ 4&9&5&6&7&0&2&1&3&8 \\ 1&0&8&2&5&9&4&3&6&7 \\ 3&4&0&7&2&8&1&9&5&6 \\ 5&3&9&8&6&1&7&0&2&4 \end{bmatrix} C = \begin{bmatrix} 3&6&8&5&9&4&2&0&7&1 \\ 5&9&6&0&8&1&4&3&2&7 \\ 1&8&7&2&0&6&9&5&3&4 \\ 6&3&0&4&1&7&8&2&9&5 \\ 2&1&3&8&5&9&0&7&4&6 \\ 8&4&2&1&6&0&7&9&5&3 \\ 9&2&5&3&7&8&6&4&1&0 \\ 0&7&4&9&2&5&3&1&6&8 \\ 4&0&1&7&3&2&5&6&8&9 \\ 7&5&9&6&4&3&1&8&0&2 \end{bmatrix}$$

Рис. 8. Псевдотройка ортогональных диагональных латинских квадратов порядка 10, найденная с помощью MPI программы

$$\begin{bmatrix} - & 01 & 02 & 03 & 04 & 05 & 06 & 07 & 08 & - \\ 10 & 11 & - & 13 & 14 & 15 & 16 & - & - & - \\ - & 21 & - & - & - & - & 26 & 27 & 28 & 29 \\ 30 & - & 32 & - & 34 & 35 & 36 & - & 38 & - \\ 40 & 41 & - & 43 & - & - & - & - & - & 49 \\ 50 & - & 52 & 53 & 54 & 55 & - & - & 58 & - \\ 60 & - & 62 & 63 & 64 & 65 & 66 & - & 68 & - \\ 70 & 71 & - & - & - & 75 & - & 77 & 78 & 79 \\ 80 & - & 82 & - & - & 85 & 86 & 87 & - & 89 \\ - & 91 & 92 & 93 & 94 & - & 96 & 97 & 98 & 99 \end{bmatrix}$$

Рис. 9. Набор из 62 упорядоченных пар элементов, по которым ортогональны все три пары квадратов, входящих в псевдотройку, изображенную на **Рис. 8**

Заключение

В статье описано успешное применение проекта добровольных распределенных вычислений SAT@home для поиска пар ортогональных диагональных латинских квадратов порядка 10. Найдено 17 таких пар, все они являются новыми (в сравнении с 3 ранее известными). На основе найденных пар с помощью параллельной реализации алгоритма генерации диагональных латинских квадратов была найдена псевдотройка с рекордным количеством упорядоченных пар элементов, по которым ортогональны все три пары квадратов из псевдотройки. В дальнейшем мы планируем разработать и реализовать новые пропозициональные кодировки для приведенных в статье задач, а также ускорить работу используемого SAT-решателя на задачах поиска систем ортогональных латинских квадратов.

Список литературы

1. Тужилин М. Э. Об истории исследований латинских квадратов // Обозрение прикладной и промышленной математики. 2012. Т. 19, Вып. 2. С. 226-227.
2. Colbourn C.J., Dinitz J.H. Handbook of Combinatorial Designs. Second Edition. Chapman&Hall, 2006. 984 p.
3. Biere A., Heule V., van Maaren H., Walsh T. (eds.) Handbook of Satisfiability. IOS Press, 2009.
4. Семенов А.А., Заикин О.С., Отпущенников И.В., Кочемазов С.Е. Применение алгоритмов решения проблемы булевой выполнимости (SAT) к комбинаторным задачам // Труды XII Всероссийского совещания по проблемам управления. Изд-во ИПУ РАН, 2014. С. 7361-7374.
5. Семенов А.А., Беспалов Д.В. Технологии решения многомерных задач логического поиска // Вестник Томского государственного университета. 2005. № 14. С. 61.
6. Zhang H. Combinatorial Designs by SAT Solvers. In: Biere A., Heule V., van Maaren H., Walsh T. (eds.) Handbook of Satisfiability. IOS Press, 2009, pp. 533-568.

7. Gomes C. and Shmoys D. Completing quasigroups or latin squares: A structured graph coloring problem // Proceedings of the Computational Symposium on Graph Coloring and Generalizations. 2002. P. 22-39.
8. Brown et al. Completion of the Spectrum of Orthogonal Diagonal Latin Squares. Lecture notes in pure and applied mathematics. 1992. Vol. 139. P. 43-49.
9. Заикин О.С., Посыпкин М.А., Семёнов А.А., Храпов Н.П. Опыт организации добровольных вычислений на примере проектов OPTIMA@home и SAT@home // Вестник Нижегородского университета им. Н.И. Лобачевского. 2012. № 5-2. С. 340-347.
10. Заикин О.С., Семенов А.А., Посыпкин М.А. Процедуры построения декомпозиционных множеств для распределенного решения SAT-задач в проекте добровольных вычислений SAT@home // Управление большими системами. Вып. 43. М.: ИПУ РАН, 2013. С. 138-156.
11. SAT@home: проект добровольных распределенных вычислений для решения крупномасштабных SAT-задач. URL: http://sat.isa.ru/pdsat/ (дата обращения 30.11.2014).
12. Anderson, D.P. BOINC: A System for Public-Resource Computing and Storage // In: Buyya, R. (ed.) GRID. IEEE Computer Society, 2004. P. 4-10.
13. 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.
14. Заикин О.С., Семенов А.А. Применение метода Монте-Карло к прогнозированию времени параллельного решения проблемы булевой выполнимости // Вычислительные методы и программирование: новые вычислительные технологии. 2014. Т. 15, № 1. С. 22-35.
15. Rainbow-таблицы для шифра A5/1. URL: http://opensource.srlabs.de/projects/a51-decrypt (дата обращения 30.11.2014).
16. Een N., Sorensson N. An Extensible SAT-solver // In: Giunchiglia, E., Tacchella, A. (eds.) SAT. LNCS, vol. 2919, Springer (2003). P. 502-518.
17. J. Egan, I. M. Wanless. Enumeration of MOLS of small order. arXiv:1406.3681v2 [math.CO].
18. Гришагин В.А., Свистунов А.Н. Параллельное программирование на основе MPI. Изд-во ННГУ им. Н.И. Лобачевского, 2005. 93 с.