О ВЫЧИСЛИТЕЛЬНОМ ПОИСКЕ КВАЗИОРТОГОНАЛЬНЫХ СИСТЕМ ЛАТИНСКИХ КВАДРАТОВ, БЛИЗКИХ К ОРТОГОНАЛЬНЫМ СИСТЕМАМ

Белей Е.Г.
Институт математики, экономики и информатики
Иркутского государственного университета.

Abstract. Работа посвящена исследованию возможностей применения современных алгоритмов вычислительной логики к задачам поиска комбинаторных структур. Конкретно, рассматривается известная открытая комбинаторная задача о существовании тройки попарно ортогональных латинских квадратов 10-го порядка. Предлагается схема поиска такой тройки в форме итеративной процедуры, на каждой итерации которой строятся т.н. квазиортогональные системы. Вводится специальная характеристика близости квазиортогональной системы к ортогональной, называемая индексом ортогональности. Искомая ортогональная тройка латинских квадратов порядка 10, если она существует, является квазиортогональной системой с индексом ортогональности 100. В работе построены семейства квазиортогональных систем с индексом ортогональности до 80 включительно. Для этой цели использованы алгоритмы решения задачи о булевой выполнимости (SAT).

Ключевые слова: латинские, греко-латинские квадраты, ортогональные и квазиортогональные системы латинских квадратов, задача о булевой выполнимости, SAT

1. Введение.

Задачи поиска комбинаторных конфигураций, связанных с латинскими квадратами, притягивают внимание ученых на протяжении нескольких столетий. Латинские квадраты можно рассматривать в тесной связи с различными математическими объектами: таблицы Кэли для групп и квазигрупп [1], ортогональные массивы [2,3], математические головоломки (квадраты Судоку [4]) и многое другое.

Считается, что первое упоминание о латинских квадратах содержится в труде исламского шейха и мистика Ахмада Аль-Буни «Солнце познания», написанном примерно в 1200 году. Систематическое исследование латинских квадратов было предпринято Леонардом Эйлером. Собственно, ему данный объект и обязан своим названием, поскольку для заполнения ячеек квадрата Эйлер использовал буквы латинского алфавита. Именно Эйлер, по-видимому, был первым, кто стал исследовать т.н. «греко-латинские квадраты». Уже на тот момент было понятно, что задачи существования греко-латинских квадратов далеко не тривиальны, в отличие от задач существования латинских квадратов (которые существуют для любого значения порядка). Эйлером был выдвинут ряд гипотез о греко-латинских квадратах, одна из которых продержалась 177 лет – гипотеза о несуществовании греко-латинских квадратов для всех порядков вида $n = 4k + 2$. Данная гипотеза была опровергнута Р. Боузом, С. Шрикханде и Э. Паркером лишь в 1959 году. В процессе получения этого результата был построен греко-латинский квадрат 10-порядка. Для этой цели использовался вычислительный эксперимент.

Пример с греко-латинскими квадратами 10-го порядка показывает, что даже для комбинаторных задач весьма высокой размерности и изначально плохой (верхней) оценкой количества вычислений можно надеяться на успешное решение задачи за счет различных техник сокращения перебора. В последние годы возникли и развиваются целые классы комбинаторных алгоритмов и сопутствующих им структур данных, для которых разработаны и продолжают разрабатываться весьма успешные на практике стратегии сокращения перебора. В качестве таковых имеются в виду, прежде всего, алгоритмы решения задачи целочисленного

программирования [5], а также алгоритмы решения задачи о булевой выполнимости (SAT) [6]. В настоящей работе основное внимание уделено второму классу алгоритмов. Обоснование этого выбора заключается в том, что SAT-подход весьма успешно применяется к решению индустриальных задач, без преувеличения, огромных размерностей (в первую очередь подразумеваются задачи символьной верификации). Более того, алгоритмы решения SAT справляются даже с такими аргументированно трудными задачами, как криптоанализ некоторых действующих систем шифрования ([7], [8] и др).

Конкретно, в настоящей работе изучается одна из наиболее известных открытых проблем, связанных с латинскими и греко-латинскими квадратами: существует ли система из трех латинских квадратов 10-порядка, любая пара которых образует греко-латинский квадрат? Основной результат работы состоит в вычислительном построении объектов, близких к искомой структуре в смысле некоторой естественной меры. В вычислительных экспериментах использовались два класса алгоритмов решения SAT – CDCL (Conflict Driven Clause Learning) алгоритмы и SLS (Stochastic Local Search) алгоритмы.

Приведем краткое описание структуры настоящей работы. Во втором разделе даются определения, связанные с основным объектом исследований. Третий раздел представляет собой краткое введение в применение SAT-подхода к решению комбинаторных задач. В четвертом разделе описаны SAT-кодировки проблемы поиска квазиортогональных систем латинских квадратов с ограничениями на индекс ортогональности. В пятом разделе приводятся результаты вычислительных экспериментов. Шестой раздел представляет собой краткий перечень близких по тематике работ, вышедших в последнее время. В седьмом разделе кратко описаны ближайшие планы.

2. Латинские квадраты, ортогональные и квазиортогональные системы латинских квадратов.

Определение 1.

Латинский квадрат порядка n – это таблица $n \times n$, заполненная элементами из множества $A = \{\alpha_1, \dots, \alpha_n\}$, $\alpha_i \neq \alpha_j, i, j = 1, .., n, i \neq j$, такая, что в каждой строке и каждом столбце встречаются все n символов. Множество A далее называем алфавитом.

$$\begin{bmatrix} 1 & 2 & 3 & 4 & 5 \\ 5 & 3 & 4 & 1 & 2 \\ 4 & 5 & 2 & 3 & 1 \\ 2 & 4 & 1 & 5 & 3 \\ 3 & 1 & 5 & 2 & 4 \end{bmatrix}$$

Рис. 1. Латинский квадрат порядка 5, используется алфавит $A = \{1,2,3,4,5\}$.

Как уже было сказано выше, латинские квадраты любого порядка $n, n \in \mathrm{N}$, существуют. Однако многие вопросы, касающиеся латинских квадратов, остаются открытыми: не известно даже точное их число уже при $n > 11$. Также остается открытым вопрос о точном числе трансверсалей [3] в латинском квадрате произвольного порядка. В настоящей работе подобные вопросы не рассматриваются. Основным объектом наших исследований являются греко-латинские квадраты и более общие структуры – ортогональные и квазиортогональные системы латинских квадратов.

Определение 2.

Рассмотрим два латинских квадрата A, B произвольного порядка n. Занумеруем ячейки первого и второго квадратов в одинаковой последовательности τ, используя для этой цели числа от 1 до n^2. Рассмотрим квадрат $A|B$, элементами которого являются все возможные

пары (a,b), $a \in A, b \in B$, *выписываемые в последовательности* τ. *Пары* (a_i,b_i) *и* (a_j,b_j) *различны по определению тогда и только тогда, когда* $a_i \neq a_j$ *или* $b_i \neq b_j, i,j \in \{1,\dots,n^2\}, i \neq j$. *Квадрат* $A|B$ *называется греко-латинским, если все составляющие его пары различны. Если два латинских квадрата* A,B *образуют греко-латинский квадрат, то говорят, что* A *и* B *ортогональны (обозначение «*$A \perp B$*»).*

$$A = \begin{bmatrix} 1 & 2 & 3 \\ 2 & 3 & 1 \\ 3 & 1 & 2 \end{bmatrix} \quad B = \begin{bmatrix} 1 & 2 & 3 \\ 3 & 1 & 2 \\ 2 & 3 & 1 \end{bmatrix}$$

$$A|B = \begin{bmatrix} 11 & 22 & 33 \\ 23 & 31 & 12 \\ 32 & 13 & 21 \end{bmatrix}$$

Рис. 2. Пара ортогональных латинских квадратов порядка 3
и образованный ими греко-латинский квадрат

Определение 3.

Набор из K *латинских квадратов порядка* n, *в котором любые два квадрата ортогональны, называется системой из* K *ортогональных квадратов порядка* n *или ортогональной системой мощности* K *и порядка* n. *Кратко будем говорить о такой системе как об ортогональной* (K,n) *системе.*

Известно (см., например, [2]), что для фиксированного n мощность ортогональной системы порядка n не может превосходить $n-1$.

$$A = \begin{bmatrix} 2 & 1 & 4 & 3 \\ 3 & 4 & 1 & 2 \\ 1 & 2 & 3 & 4 \\ 4 & 3 & 2 & 1 \end{bmatrix} \quad B = \begin{bmatrix} 3 & 4 & 1 & 2 \\ 1 & 2 & 3 & 4 \\ 2 & 1 & 4 & 3 \\ 4 & 3 & 2 & 1 \end{bmatrix}$$

$$C = \begin{bmatrix} 1 & 2 & 3 & 4 \\ 2 & 1 & 4 & 3 \\ 3 & 4 & 1 & 2 \\ 4 & 3 & 2 & 1 \end{bmatrix}$$

Рис. 3. Ортогональная система мощности 3 и порядка 4.

Один из самых известных открытых вопросов, относящийся к латинским квадратам, заключается в следующем: существует ли ортогональная система мощности три и порядка 10?

Главная привлекательность этой задачи в ее, казалось бы, относительно небольшой комбинаторной размерности, что может быть основанием для применения к ней современных комбинаторных алгоритмов, использующих различные техники сокращения перебора. Основная идея, которая эксплуатируется далее, заключается в рассмотрении ортогональной $(3,10)$ системы как предельного случая для комбинаторных конфигураций с менее жесткими ограничениями на структуру. В частности, мы можем рассматривать, например, такие 3 латинских квадрата, два из которых не образуют греко-латинский: скажем, для квадратов A,B,C может выполняться $A \perp B, B \perp C$, но при этом A и C не ортогональны. Можно даже допустить, чтобы любая пара квадратов из трех не была ортогональной. Однако в этом случае необходимо предложить какую-либо разумную характеристику, которая позволяла бы сравнивать такие системы между собой, и наилучшая в смысле этой характеристики система давала бы ортогональную $(3,10)$ систему. Ниже мы рассматриваем пример такой характеристики.

Определение 4.

Пусть A, B – произвольные латинские квадраты порядка n. Рассмотрим квадрат $A|B$ и выпишем все пары вида (a_i, b_i), $i \in \{1, \dots, n^2\}$, в некоторой фиксированной последовательности. Пусть r– число различных пар в полученной последовательности (в указанном выше смысле). Назовем число $r = r(A, B)$, $1 \le r \le n^2$, индексом ортогональности, а пару квадратов A, B будем называть квазиортогональной парой индекса r. Пусть A, B – латинские квадраты порядка n. Очевидно, что $A \perp B$ тогда и только тогда, когда A, B – квазиортогональная пара индекса n^2.

$$A = \begin{bmatrix} 1 & 2 & 3 \\ 2 & 3 & 1 \\ 3 & 1 & 2 \end{bmatrix} \quad B = \begin{bmatrix} 1 & 2 & 3 \\ 2 & 3 & 1 \\ 3 & 1 & 2 \end{bmatrix}$$

$$A|B = \begin{bmatrix} 11 & 22 & 33 \\ 22 & 33 & 11 \\ 33 & 11 & 22 \end{bmatrix}$$

Рис. 4 Квазиортогональная пара порядка 3 индекса 3.

В силу всего сказанного выше очевидным образом справедливо следующее утверждение.

Утверждение 1.

Рассмотрим набор из K латинских квадратов порядка n: $A_1, \dots, A_K$, $K \le n - 1$. Для каждой пары чисел вида $i, j \in \{1, \dots, K\}$, $i \ne j$, найдем индекс ортогональности пары квадратов A_i, A_j, который обозначим через r_{ij}. Введем обозначение: $r_* = \min\limits_{i,j \in \{1,\dots,K\}: i \ne j} \{r_{ij}\}$. Число r_* назовем индексом ортогональности системы $A_1, \dots, A_K$. Система $A_1, \dots, A_K$ является ортогональной (K, n) системой тогда и только тогда, когда $r_* = n^2$.

Данное утверждение, несмотря на его простоту, позволяет сформулировать общую стратегию построения ортогональной (K, n) системы как стратегию последовательного улучшения квазиортогональных систем. Более точно, далее мы будем рассматривать задачу существования квазиортогональной системы с индексом ортогональности $r_* < n^2$ как задачу поиска выполняющего набора булевой формулы F специального вида. Если для какого-либо значения r_* нам удается найти квазиортогональную систему индекса r_*, то взяв за основу соответствующий набор, выполняющий F, мы можем подставить в формулу F часть этой информации и перестроить полученную формулу так, чтобы она кодировала существование квазиортогональной системы большего индекса. После чего снова пытаться решить задачу о выполнимости полученной формулы.

3. Задача о булевой выполнимости и некоторые алгоритмы ее решения.

Далее рассматриваются переменные, принимающие значения в множестве символов $\{0,1\}$. Такие переменные называются булевыми. Пусть $X = \{x_1, \dots, x_n\}$ – множество булевых переменных и $F(x_1, \dots, x_n)$ – выражение над данным множеством, а также множеством специальных символов, задающих булевы операции, построенное по определенным правилам. Выражение $F(x_1, \dots, x_n)$ в этом случае называется булевой формулой. Правила построения булевых формул можно найти в [9] (где они называются формулами исчисления высказываний) или в [10] (где они называются формулами алгебры логики).

Задача о булевой выполнимости (Boolean Satisfiability, SAT) заключается в следующем: дана произвольная булева формула $F = F(x_1, \dots, x_n)$. Требуется ответить на вопрос о существовании такого набора значений переменных из X, подстановка которого в F обращает ее в 1. Если такой набор существует, он называется набором, выполняющим F, а сама формула в этом случае

называется выполнимой. В противном случае формула F называется невыполнимой. SAT является исторически первой задачей, для которой была установлена ее NP-полнота [11]. Соответственно, поисковый вариант SAT (найти выполняющий набор либо доказать невыполнимость F) NP-трудна [12]. Можно показать, что задача о выполнимости произвольной булевой формулы F эффективно (за полиномиальное в общем случае время) сводится к задаче о выполнимости булевой формулы в конъюнктивной нормальной форме (КНФ). Результирующая КНФ может содержать, правда, больше переменных, чем исходная формула, однако входящие в КНФ ограничения (дизъюнкты) имеют простую природу и могут быть весьма эффективно представлены в памяти ЭВМ. Соответственно, под SAT чаще всего понимается задача о выполнимости произвольной КНФ, и именно в этом смысле SAT рассматривается в настоящей работе.

Несмотря на NP-трудность, во многих частных случаях SAT весьма эффективно решается для существенных размерностей (десятки тысяч переменных и сотни тысяч дизъюнктов). Соответствующие алгоритмы решения SAT эксплуатируют различные техники сокращения перебора. Мы выделим здесь две концепции решения SAT, получившие, пожалуй, наиболее широкое распространение: SLS (Stochastic Local Search) [13-16] и CDCL (Conflict Driven Clause Learning) [17-18]. SLS-алгоритмы основаны на идеологии локального поиска – итеративной процедуре, в каждой итерации которой осуществляется проверка окрестности рассматриваемого набора значений булевых переменных в надежде улучшить значение целевой функции (обычно требуется максимизировать число выполненных дизъюнктов в рассматриваемой КНФ). В ситуации, когда набор является локальным максимумом в рассматриваемой окрестности, используются различные способы выхода из таких точек (главным образом, за счет процедур, имеющих природу случайных блужданий). CDCL-алгоритмы сочетают полную стратегию обхода дерева поиска с использованием памяти для записи конфликтной информации. Последнее позволяет во многих случаях осуществлять глубокие откаты (т.н. “Backjumping”), что является одним из главных факторов эффективности CDCL. CDCL-алгоритмы с успехом используются для решения различных индустриальных задач большой размерности (символьная верификация, биоинформатика, криптография и др.).

4. Булево кодирование проблем поиска квазиортогональных систем латинских квадратов.

Булево или пропозициональное кодирование комбинаторной задачи P заключается в построении булевой формулы F_P, обладающей следующими свойствами:

1. если формула F_P невыполнима, то задача P не имеет решений;
2. если F_P выполнима, то из любого выполняющего набора данной формулы можно эффективно извлечь двоичное описание некоторого решения задачи P.

Для весьма широкого класса комбинаторных задач возможно эффективное (полиномиальное от длины описания исходной задачи) булево кодирование (по сути, все задачи из класса NP являются таковыми).

Процесс пропозиционального кодирования проблемы существования квазиортогональной системы латинских квадратов можно разбить на два этапов. На первом этапе строятся формулы, кодирующие тот факт, что все рассматриваемые квадратные матрицы являются латинскими квадратами. На втором этапе кодируется условие частичной ортогональности для всех пар квадратов.

Разберем процесс кодирования того факта, что рассматриваемая квадратная матрица порядка n является латинским квадратом. Сразу оговоримся, что используемая далее кодировка известна [19-20]. Это, пожалуй, самая простая кодировка, она самая эффективная по числу переменных, но при этом сильно проигрывает другим способам кодирования латинских квадратов по числу дизъюнктов и числу литералов [22]. Мы будем называть данную кодировку

наивной. Что удивительно, именно наивная кодировка оказывается наиболее простой для SAT-решателей, причем как для SLS, так и для CDCL (это было выяснено в результате большого числа вычислительных экспериментов).

Основные дальнейшие построения базируются на свойстве, которое в некоторых источниках называется Only One Cardinality (сокращенно OOC). Пусть $X = \{x_1, \dots, x_m\}$ – множество булевых переменных. Рассмотрим следующую формулу:

$$OOC(x_1, \dots, x_m) = (x_1 \vee \dots \vee x_m) \wedge \bigwedge_{1 \le i < j \le m} (\neg x_i \vee \neg x_j). \qquad (1)$$

Несложно убедиться, что (1) выполнима на всех таких наборах значений переменных из X, в которых одна переменная принимает значение 1, а все остальные переменные принимают значение 0. Пусть S – множество всех наборов, выполняющих (1). С произвольным $v \in S$ ассоциируем натуральное число, равное номеру координаты в v, значение которой есть 1. Тем самым булевы векторы из S могут рассматриваться как кодировки натуральных чисел из $\{1, \dots, m\}$.

Рассмотрим квадратную матрицу L порядка n. Будем считать, что элементы матрицы – это числа из множества $A = \{1,2, \dots, n\}$. Пронумеруем все ячейки матрицы числами от 1 до n^2 в некотором фиксированном порядке. Каждой ячейке с номером $q \in \{1, \dots, n^2\}$ сопоставим множество булевых переменных $X^q = \{x_1^q, \dots, x_n^q\}$. Для наглядного представления процесса пропозиционального кодирования условия, что матрица L является латинским квадратом, рассмотрим следующий рисунок ($n = 3$).

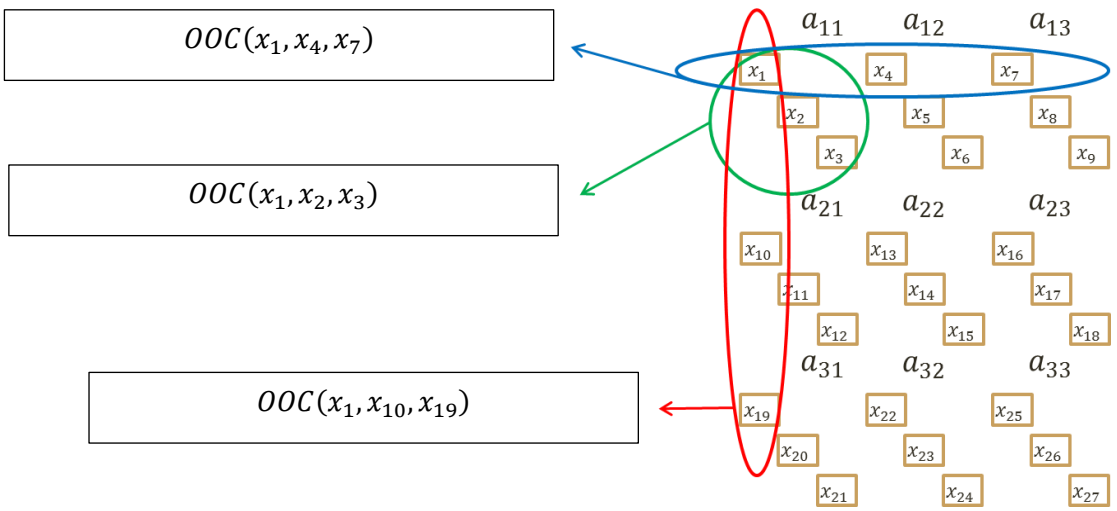

Рис. 5. Пояснение к процедуре пропозиционального кодирования латинского квадрата.

Прокомментируем рисунок 5. Для удобства используется сквозная нумерация переменных. Как уже было сказано выше, мы используем для представления натуральных чисел из $\{1, \dots, n\}$ единичные булевы векторы длины n. Соответственно, в рассматриваемом примере 100 кодирует 1, 010 кодирует 2, 001 кодирует 3. Тогда на переменные, сопоставленные, например, первой ячейке матрицы, накладывается дополнительно ограничение $OOC(x_1, x_2, x_3)$, которое означает, что набор значений переменных из $X = \{x_1, x_2, x_3\}$ должен выполнять соответствующую формулу вида (1), то есть должен принадлежать множеству $\{100,010,001\}$. Это соответствует тому факту, что в первой ячейке матрицы может стоять любое число из множества $\{1,2,3\}$. Аналогичные ограничения накладываются на переменные, кодирующие содержимое всех остальных ячеек матрицы. Также мы вводим дополнительные ограничения типа $OOC(x_1, x_4, x_7)$. Очевидно, что одновременное выполнение $OOC(x_1, x_4, x_7)$, $OOC(x_2, x_5, x_8)$ и $OOC(x_3, x_6, x_9)$ при действующих аналогичных ограничениях на ячейки означает, что в первой

строке матрицы находятся все числа из множества $\{1,2,3\}$. Аналогично, одновременное выполнение условий $OOC(x_1, x_{10}, x_{19})$, $OOC(x_2, x_{11}, x_{20})$, $OOC(x_3, x_{11}, x_{21})$ при действующих ограничениях на содержимое ячеек означает, что в первом столбце матрицы находятся все числа из $\{1,2,3\}$. Тот факт, что одновременно выполняется несколько OOC- условий эквивалентен истинности конъюнкции соответствующего множества формул вида (1).

Несложно построить оценки числа переменных и дизъюнктов в описанной пропозициональной кодировке латинских квадратов. Очевидно, что для кодирования одного латинского квадрата порядка n потребуется n^3 булевых переменных. Число дизъюнктов в формуле (1), выражающей $OOC(x_1, \dots, x_n)$, есть $1 + \frac{n(n-1)}{2}$. Всего потребуется:

1. n^2 условий OOC от n переменных, соответствующих ячейкам матрицы;
2. n^2 условий OOC, необходимых для кодирования того факта, что в каждой строке матрицы присутствуют все числа от 1 до n;
3. n^2 условий OOC, необходимых для кодирования того факта, что в каждом столбце матрицы присутствуют все числа от 1 до n.

С учетом сказанного асимптотика по числу дизъюнктов в описанной кодировке выглядит как $O(n^4)$.

Отметим здесь, что для представления свойства Only One Cardinality можно использовать различные способы кодирования (см., например, [23-24]), однако, как уже было сказано выше, в сравнении с наивной все другие кодировки оказываются более сложными для SAT-решателей.

Перейдем теперь к кодированию условия квазиортогональности. Предположим, что рассматривается система из K латинских квадратов порядка n. Пусть A, B – произвольная пара квадратов из данной системы и пусть все ячейки этих квадратов, а также квадрата $A|B$ занумерованы в одинаковой последовательности числами от 1 до n^2. Каждой ячейке квадрата $A|B$ с номером $q \in \{1, \dots, n^2\}$ сопоставим пару множеств булевых переменных $\left(X_A^q, X_B^q\right)$. Каждое из множеств X_A^q, X_B^q состоит из n булевых переменных, кодирующих содержимое соответствующих ячеек в квадратах A и B в соответствии с описанной выше схемой.

Теперь рассмотрим множество всех возможных пар чисел, которые в принципе могут появиться в квадрате $A|B$ (неважно в каких ячейках):

$$(1,1), (1,2), \dots, (1,n), (2,1), \dots, (2,n), \dots, (n,1), \dots, (n,n). \qquad (2)$$

Будем рассматривать (2) как упорядоченное множество, считая первой пару $(1,1)$, второй – пару $(1,2)$ и т.д., последней – пару (n,n) (по сути, лексикографический порядок). Отметим, что для любого $q \in \{1, \dots, n^2\}$ переменные X_A^q, X_B^q в принципе могут кодировать любую из пар чисел, перечисленных в (2).

Свяжем с парой квадратов A, B булев вектор длины n^2 $\chi(A|B) = (\chi_1, \dots, \chi_{n^2})$, компоненты которого определим следующим образом:

$$\chi_i = \begin{cases} 1, \text{если пара чисел из (2) с номером } i \text{ встречается в } A|B \\ 0, \text{в противном случае} \end{cases}$$

Назовем заданный так вектор $\chi(A|B)$ вектором маркировки пары квадратов A, B. В силу всего сказанного выше справедливо следующее утверждение.

Утверждение 2.

Для пары латинских квадратов A, B индекс их ортогональности равен весу Хэмминга вектора маркировки: $r(A,B) = wt_H\left(\chi(A|B)\right)$.

Доказательство.

Обозначим через P_n множество пар чисел вида (2). Пусть $r = r(A,B)$ – индекс ортогональности пары квадратов A, B. В силу определения r равен числу различных пар из множества P_n, встречающихся в $A|B$. Пусть $p_{t_1}, \dots, p_{t_r}$ – все различные такие пары, тогда в векторе $\chi(A|B)$ этим парам соответствуют единицы. Любая пара из $P_n \backslash \{p_{t_1}, \dots, p_{t_r}\}$ не встречается в $A|B$, поэтому всем таким парам в $\chi(A|B)$ соответствуют нули. Как итог,

$wt_H\big(\chi(A|B)\big) = r$. Утверждение 2 доказано.

Для кодирования того факта, что квадраты A, B квазиортогональны с индексом ортогональности не меньше r, мы должны сначала определить условия, при которых булевы переменные χ_i, $i \in \{1, \dots, n^2\}$, принимают значение 1. Для этой цели мы учитываем, что пара $p_i \in P_n$ может стоять в $A|B$ на любом месте. Скажем, для пары $(1,1)$ в таком случае имеем следующее условие:

$$\big((X_A^1, X_B^1) = (1,1)\big) \vee \big((X_A^2, X_B^2) = (1,1)\big) \vee \dots \vee \Big(\big(X_A^{n^2}, X_B^{n^2}\big) = (1,1)\Big) \qquad (3)$$

В записи (3), например, выражение $(X_A^1, X_B^1) = (1,1)$ подразумевает, что значение переменных из множества X_A^1 – это булев вектор $(10 \dots 0)$ и то же самое имеет место в отношении переменных из множества X_B^1. С учетом сказанного, несложно переписать (3) в виде булевой формулы, которую мы обозначим через $F_{(1,1)}(X)$, где $X = \bigcup_{q=1}^{n^2} X^q$. Тот факт, что χ_1 принимает значение 1 тогда и только тогда, когда $F_{(1,1)}(X)$ принимает значение 1, записывается в виде следующей формулы:

$$\chi_1 \equiv F_{(1,1)}(X), \qquad (4)$$

где через $\equiv$ обозначена логическая эквивалентность. По аналогии с (4) зададим все остальные компоненты вектора $\chi(A|B)$.

Теперь нам остается записать в виде булевой формулы тот факт, что в векторе маркировки присутствует $\geq r$ единиц. Для этой цели мы используем специальный предикат, который носит название $AL_k(x_1, \dots, x_m)$. Данный предикат принимает значение 1 на булевых векторах длины m, вес Хемминга которых не меньше некоторого натурального k. Соответственно, в нашем случае потребуется булева формула, задающая предикат $AL_r(\chi_1, \dots, \chi_{n^2})$, где r– произвольное значение индекса ортогональности для рассматриваемой пары квадратов.

Предикат $AL_k(x_1, \dots, x_m)$ можно задать булевой формулой в КНФ при помощи схемы кодирования т.н. последовательного счетчика (Sequential Counter), предложенной К. Синцем в [24]. В данной работе описывается предикат $AM_k(x_1, \dots, x_m)$, принимающий значение 1 на любом булевом векторе длины m, вес Хэмминга которого $\leq k$, и принимающий значение 0 на остальных булевых векторах длины m.

Пусть имеется m булевых переменных $x_1, \dots, x_m$. Тогда $AM_k(x_1, \dots, x_m)$ задается следующей формулой в КНФ:

$$\left.\begin{array}{c} (\neg x_1 \vee s_{1,1}) \\ (\neg s_{1,j}), \text{для } 1 < j \leq k \\ (\neg x_i \vee s_{i,1}) \\ (\neg s_{i-1,1} \vee s_{i,1}) \\ \left.\begin{array}{c} (\neg x_i \vee \neg s_{i-1,j-1} \vee s_{i,j}) \\ (\neg s_{i-1,j} \vee s_{i,j}) \end{array}\right\}, \text{для } 1 < j \leq k \\ (\neg x_i \vee \neg s_{i-1,k}) \\ (\neg x_m \vee \neg s_{m-1,k}) \end{array}\right\}, \text{для } 1 < i < m \qquad (5)$$

В формуле (5) фигурируют km дополнительных переменных $s_{11}, \dots, s_{km}$. Подставив вместо вхождений литералов $x_1, \dots, x_m$ литералы $\neg x_1, \dots, \neg x_m$ и сделав элементарные преобразования, получим КНФ, задающую предикат $AL_{m-k}(x_1, \dots, x_m)$.

Итак, подведем итог процессу построения пропозициональной кодировки условия квазиортогональности. Пропозициональная кодировка системы из K квазиортогональных латинских квадратов порядка n это конъюнкция следующих формул:

1. K КНФ, кодирующих латинские квадраты (конъюнкции соответствующих множеств формул вида (1));

2. $\frac{K(K-1)}{2}$ формул, задающих векторы маркировки для соответствующих пар квадратов (каждая такая формула есть конъюнкция n^2 формул вида (4));
3. $\frac{K(K-1)}{2}$ формул, задающих предикаты $AL_{r_{ij}}(\chi_1, \dots, \chi_{n^2})$, $1 \le i < j \le K$.

Формулы в пункте 2 могут быть превращены в КНФ с использованием преобразований Цейтина [25]. Как итог, имеем эффективную процедуру сведения проблемы поиска квазиортогональной системы латинских квадратов к SAT.

5. Вычислительные эксперименты.

Во всех вычислительных экспериментах исследовались задачи поиска квазиортогональных систем, состоящих их 3 латинских квадратов 10 порядка (далее кратко «квазиортогональные $(3,10)$ системы»). В экспериментах использовались два SLS SAT-решателя CSCCSat [26], DCCalm [27], а также CDCL-решатели lingeling, plingeling [28], cryptominisat [29]. Вычисления проводились на платформах Intel Core i7, Opteron 6276.

В ходе первого эксперимента решались задачи поиска квазиортогональной $(3,10)$ системы с индексами ортогональности от 70 до 80. При этом использовались пропозициональные кодировки соответствующих задач, построенных согласно схеме кодирования, описанной в предыдущем пункте. Результаты этой стадии экспериментов отражены в таблице 1. Некоторые эксперименты были прерваны по лимиту времени в 6 часов (прерванным тестам соответствуют строки в таблице, выделенные жирным шрифтом).

Таблица 1.

Индекс ортогональности	Время решения	SAT-решатель
70	13 секунд	CSCCSat
	20 секунд	DCCalm
	40 секунд	plingeling
	5 секунд	cryptominisat 5.0
72	13 минут	CSCCSat
	более 6 часов	DCCalm
	2 секунды	plingeling
	30 секунд	cryptominisat 5.0
75	**более 6 часов**	CSCCSat
	более 6 часов	DCCalm
	18 минут	plingeling
	5 секунд	cryptominisat 5.0
78	**более 6 часов**	CSCCSat
	более 6 часов	DCCalm
	30 минут	plingeling
	более 6 часов	cryptominisat 5.0
80	**более 6 часов**	CSCCSat
	более 6 часов	DCCalm
	более 6 часов	plingeling
	6 часов	cryptominisat 5.0

На второй стадии экспериментов часть информации из найденных систем подставлялась в КНФ, кодирующую проблему поиска квазиортогональной системы большего индекса

ортогональности. В роли такой информации использовалось содержимое некоторых ячеек найденных квадратов: конкретно, в каждом квадрате выбирались (как правило, случайным образом) от 20 до 50 процентов пар вида (a_i, b_i), $i \in \{1, \dots, n^2\}$, встречающихся ровно один раз. Значения переменных, кодирующих ячейки латинских квадратов, содержащих выбранные пары, а также соответствующие им значения переменных, кодирующих векторы маркировок, приписывались в виде однолитеральных дизъюнктов к КНФ, кодирующей задачу поиска квазиортогональной системы с большим индексом ортогональности. Результаты данной стадии экспериментов приведены в таблице 2. В первом столбце данной таблицы содержится информация об исходной квазиортогональной системе, найденной на первой стадии эксперимента (указан индекс ортогональности, решатель и время решения соответствующей SAT-задачи). Во втором столбце указана новая граница для индекса ортогональности, решатель и время решения задачи (поиска системы с указанным индексом), в которую подставлена часть информации из системы меньшего индекса, найденной на предыдущей стадии эксперимента. В третьем столбце указано суммарное время вычислений.

Таблица 2.

Параметры исходной задачи	Новая граница для индекса ортогональности	Суммарное время решения
72 CSCCSat 753 секунды	75 lingeling 10 секунд	12,7 минут
72 CSCCSat 753 секунды	78 lingeling 128 секунд	14,7 минут
72 CSCCSat 753 секунды	80 cryptominisat 5.0 936 секунд	28,15 минут
72 cryptominisat 5.0 2 секунды	75 plingeling 44 секунды	46 секунд
72 cryptominisat 5.0 2 секунды	78 plingeling 131 секунда	2,2 минуты
72 cryptominisat 5.0 2 секунды	80 treengeling 5 121 секунда	1,42 часа

Комментарии к таблице 2. Из результатов, приведенных в таблице 2, можно сделать вывод, что итеративная схема построения квазиортогональных систем существенно эффективнее «прямой» схемы: в первой части экспериментов единственному решателю (cryptominisat5.0) удалось найти квазиортогональную $(3{,}10)$ систему индекса 80, затратив на решение данной задачи более 6 часов. Используя итеративную схему, данный решатель тратит на поиск аналогичной системы менее получаса с использованием информации, найденной SLS-решателем, и около полутора часов с использованием информации, найденной CDCL-решателем. На наш взгляд особенно интересен тот факт, что учет информации, найденной на первой стадии экспериментов SLS-решателями, позволяет более эффективно выполнять вторую стадию эксперимента (в сравнении с ситуацией, когда используется информация, найденная CDCL-решателями).

6. Близкие по тематике работы.

Нам известно относительно немного работ, в которых SAT-подход использовался применительно к задачам, связанным с латинскими квадратами. Среди наиболее интересных стоит отметить статью [20], где анализировались возможности применения SAT к исследованию различных задач, связанных с квадратами Sudoku. Большой обзор Х. Чжана [19] содержит исчерпывающую на 2009 год информацию по задачам построения ортогональных систем латинских квадратов с использованием SAT. Насколько нам известно, похожие по постановкам задачи рассматривались в работах [30]-[31], где с использованием SAT исследовались задачи построения систем диагональных ортогональных латинских квадратов. В настоящей статье используются пропозициональные кодировки и техники поиска, основанные на иных принципах.

Наибольшая по индексу ортогональности квазиортогональная $(3,10)$-система, известная на данный момент, приведена в работе [30], ее индекс ортогональности равен 91. Однако при построении данной системы SAT никак не задействовался.

7. Ближайшие планы.

В ближайшем будущем предполагается развить описанный в работе подход, прежде всего, в направлении автоматизации итеративной схемы, в которой поиск квазиортогональной системы большего индекса учитывает информацию от квазиортогональной системы меньшего индекса. На данном этапе подставляемые данные выбираются фактически случайным образом из множества альтернатив. Планируется реализовать соответствующие процедуры в стиле алгоритмов локального поиска. С применением этих процедур планируется построить квазиортогональные системы с индексом ортогональности, существенно большим 80. В идеале хотелось бы улучшить результат работы [30].

Также планируется детально исследовать возможность использования для поиска квазиортогональных систем базовых принципов пропозиционального кодирования, отличных от рассмотренных выше. В частности, на данном этапе неплохие результаты показывает кодировка, в которой для представления содержимого ячеек квадрата используются векторы двоичных представлений натуральных чисел. Именно с использованием этой кодировки совсем недавно удалось получить квазиортогональную $(3,10)$ систему с индексом ортогональности 82[1]. Использование в перспективе различных техник распараллеливания SAT также позволяет надеяться на улучшение достигнутых результатов. Предполагается, что работа в данном направлении будет осуществляться в самое ближайшее время.

Благодарности. Работа выполнена при частичной поддержке Российского Научного Фонда, грант №16-11-10046. Автор благодарит своего научного руководителя Семенова Александра Анатольевича за внимание к работе и ценные замечания, способствовавшие улучшению качества представления результатов.

Список цитированных источников

[1] Кострикин А.И. Введение в алгебру. М.: «Наука». 1977. 495 С.
[2] Холл М. Комбинаторика. М.: «Мир». 1970. 425 С.
[3] Colbourn C.J., Dinitz J.H. Handbook of Combinatorial Designs. Second Edition. Chapman&Hall, 2006. 984 p.
[4] Lin, Keh Ying (2004), "Number Of Sudokus", Journal of Recreational Mathematics, 33 (2): 120–24.

[1]Данный результат был получен уже после отправки настоящей работы на рецензирование.

[5] Схрейвер А. Теория линейного и целочисленного программирования (в двух томах). М.: «Мир». 1991.
[6] Biere, A., Heule, M., van Maaren, H., Walsh, T. (eds.): Handbook of Satisfiability, Frontiers in Artificial Intelligence and Applications, vol. 185. IOS Press (2009)
[7] Mironov I., Zhang L. Applications of SAT solvers to cryptanalysis of hash functions // Lecture Notes in Computer Science. 2006. Vol. 4121, pp. 102–115.
[8] 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, pp. 473–483.
[9] Мендельсон Э. Введение в математическую логику. М.: «Наука». 1971. 320 С.
[10] Яблонский С.В. Введение в дискретную математику. М.: «Наука». 1986. 384 С.
[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] Гэри М., Джонсон Д. Вычислительные машины и труднорешаемые задачи. М.: «Мир». 1982. 416 С.
[13] Selman B., Kautz H., Cohen B. Noise strategies for local search. In Proc. of AAA I-94, pp. 337-343. AAAI Press/The MIT Press, 1994.
[14] Schoning U. A probabilistic algorithm for k-SAT and constraint satisfaction problems. In 40th FOCS, pages 410‑414, New York, NY, October 1999. IEEE.
[15] Hirsch E., Kojevnikov A. UnitWalk: A new SAT solver that uses local search guided by unit clause elimination. Annals Math. and AI, 43(1):91-111, 2005.
[16] Kautz H., Sabharwal A., Selman B. Incomplete Algorithms. In [6], Chapter 6. Pp. 185-203.
[17] Marques-Silva J.P., Sakallah K.A. GRASP: A search algorithm for propositional satisfiability. IEEE Trans. Computers, 48(5):506–521, 1999.
[18] Marques-Silva J.P., Lynce I., Malik S. Conflict-Driven Clause Learning SAT solvers. In [6], Chapter 4. Pp. 131–153.
[19] Zhang H. Combinatorial Design by SAT Solvers. In [6], Chapter 17. Pp. 533- 568.
[20] Lynce I. Ouaknine J. Sudoku as a sat problem. In 9th International Symposium on Artificial Intelligence and Mathematics, 2006.
[21] Steffen Hölldobler and Van Hau Nguyen. An Efficient Encoding of the At-Most-One Constraint.//Technical report, Knowledge Representation and Reasoning Group 2013-04, Technische Universität Dresden, 01062 Dresden, Germany, 2013.
[22] Кочемазов С.Е., Семенов А.А. О различных подходах к булевому кодированию задач поиска систем ортогональных латинских квадратов // Труды XVIII Байкальской конференции «Информационные и математические технологии в науке и управлении». Иркутск. 2013. Т.3. С. 40–44.
[23] Asin R, Nieuwenhuis R, Oliveras A, Rodriguez-Carbonell E (2011) Cardinality networks: a theoretical and empirical study. Constraints 16:195–221.
[24] Sinz C. Towards an Optimal Encoding of Boolean Cardinality Constraints // Lecture Notes in Computer Science. 2005. Vol. 3709. Pp. 827-831.
[25] Цейтин Г.С. О сложности вывода в исчислении высказываний // Записки научных семинаров ЛОМИ АН СССР. 1968. Т.8. С. 234–259.
[26] Shaowei Cai, Chuan Luo, Kaile Su: Scoring Functions Based on Second Level Score for k-SAT with Long Clauses // J. Artif. Intell. Res. 51, pp. 413-441 (2014)
[27] Shaowei Cai, Kaile Su: Local search for Boolean Satisfiability with configuration checking and subscore // Artif. Intell. 204, pp. 75-98 (2013)
[28] Biere, A.: Splatz, Lingeling, Plingeling, Treengeling, YalSAT Entering the SAT Competition 2016. In: Balyo, T., Heule, M., Jarvisalo, M. (eds.) Proc. of SAT Competition 2016 – Solver and Benchmark Descriptions. Department of Computer Science Series of Publications B, vol. B-2016-1, pp. 44–45. University of Helsinki (2016)

[29] Soos, M., Nohl, K., Castelluccia, C.: Extending SAT Solvers to Cryptographic Problems. In: Kullmann, O. (ed.) SAT. Lecture Notes in Computer Science, vol. 5584, pp. 244–257. Springer (2009)
[30] Zaikin, O., Kochemazov, S., Semenov, A.: SAT-based search for systems of diagonal latin squares in volunteer computing project sat@home // In Proc. of 39th International Convention on Information and Communication Technology, Electronics and Microelectronics, MIPRO 2016, Opatija, Croatia, May 30 - June 3, 2016. pp. 277–281.
[31] Zaikin, O., Zhuravlev, A., Kochemazov, S., Vatutin, E. On the construction of triples of diagonal Latin squares of order 10 // Electronic Notes in Discrete Mathematics. 2016. Vol. 54. pp. 307–312.
[32] Egan J., Wanless I. Enumeration of MOLS of Small Order// Mathematics of Computation. 2016. Vol. 85. N 298. 799-824.