

Министерство науки и высшего образования Российской Федерации
ФЕДЕРАЛЬНОЕ ГОСУДАРСТВЕННОЕ АВТОНОМНОЕ ОБРАЗОВАТЕЛЬНОЕ
УЧРЕЖДЕНИЕ ВЫСШЕГО ОБРАЗОВАНИЯ
НАЦИОНАЛЬНЫЙ ИССЛЕДОВАТЕЛЬСКИЙ УНИВЕРСИТЕТ ИТМО
ITMO University

ВЫПУСКНАЯ КВАЛИФИКАЦИОННАЯ РАБОТА/GRADUATION THESIS

Применение алгоритмов решения задачи о булевой выполнимости (SAT) к поиску комбинаторных структур, связанных с латинскими квадратами

Автор/ Author

Устинов Артем Павлович

Направленность (профиль) образовательной программы/Major

Информатика и программирование 2017

Квалификация/ Degree level

Бакалавр

Руководитель ВКР/ Thesis supervisor

Ульянцев Владимир Игоревич, кандидат технических наук, Университет ИТМО, факультет информационных технологий и программирования, доцент (квалификационная категория "ординарный доцент")

Группа/Group

М3436

Факультет/институт/кластер/ Faculty/Institute/Cluster

факультет информационных технологий и программирования

Направление подготовки/ Subject area

01.03.02 Прикладная математика и информатика

Обучающийся/Student

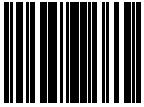
Документ подписан	
Устинов Артем Павлович	
16.05.2021	

(эл. подпись/ signature)

Устинов Артем
Павлович

(Фамилия И.О./ name
and surname)

Руководитель ВКР/ Head
of Graduate Project

Документ подписан	
Ульянцев Владимир Игоревич	
18.05.2021	

(эл. подпись/ signature)

Ульянцев
Владимир
Игоревич

(Фамилия И.О./ name
and surname)

**Министерство науки и высшего образования Российской Федерации
ФЕДЕРАЛЬНОЕ ГОСУДАРСТВЕННОЕ АВТОНОМНОЕ ОБРАЗОВАТЕЛЬНОЕ
УЧРЕЖДЕНИЕ ВЫСШЕГО ОБРАЗОВАНИЯ
НАЦИОНАЛЬНЫЙ ИССЛЕДОВАТЕЛЬСКИЙ УНИВЕРСИТЕТ ИТМО
ITMO University**

**ЗАДАНИЕ НА ВЫПУСКНУЮ КВАЛИФИКАЦИОННУЮ РАБОТУ /
OBJECTIVES FOR A GRADUATION THESIS**

Обучающийся / Student Устинов Артем Павлович

Группа/Group M3436

Факультет/институт/кластер/ Faculty/Institute/Cluster факультет информационных технологий и программирования

Квалификация/ Degree level Бакалавр

Направление подготовки/ Subject area 01.03.02 Прикладная математика и информатика

Направленность (профиль) образовательной программы/Major Информатика и программирование 2017

Специализация/ Specialization

Тема ВКР/ Thesis topic Применение алгоритмов решения задачи о булевой выполнимости (SAT) к поиску комбинаторных структур, связанных с латинскими квадратами

Руководитель ВКР/ Thesis supervisor Ульянцев Владимир Игоревич, кандидат технических наук, Университет ИТМО, факультет информационных технологий и программирования, доцент (квалификационная категория "ординарный доцент")

Срок сдачи студентом законченной работы до / Deadline for submission of complete thesis 31.05.2021

Техническое задание и исходные данные к работе/ Requirements and premise for the thesis

В рамках работы требуется исследовать вычислительные возможности алгоритмов решения проблемы булевой выполнимости (SAT) в применении к комбинаторным задачам, связанным с латинскими квадратами. Требуется изучить существующие алгоритмы и средства решения SAT, методы решения комбинаторных задач, связанных с латинскими квадратами, а также разработать новые техники, основанные на применении SAT решателей, повышающие эффективность решения задач из рассматриваемого класса.

Содержание выпускной квалификационной работы (перечень подлежащих разработке вопросов)/ Content of the thesis (list of key issues)

Изучить имеющиеся результаты по применению комбинаторных алгоритмов к решению ряда задач, связанных с латинскими квадратами. Конкретно, требуется изучить способы построения ортогональных и квазиортогональных систем латинских квадратов при помощи алгоритмов решения проблемы булевой выполнимости (SAT). Также требуется разработать новые техники, увеличивающие производительность SAT решателей на данном классе задач, и аргументировать эффективность разработанных алгоритмов посредством проведения вычислительных экспериментов.

Перечень графического материала (с указанием обязательного материала) / List of


graphic materials (with a list of required material)

Графические материалы и чертёжи в работе не предусмотрены.

Исходные материалы и пособия / Source materials and publications

1. Холл М. Комбинаторика. М.: “Мир”, 1970.
2. Белей Е.Г., Семенов А.А. О способах пропозиционального кодирования различимости объектов в конечных множествах. Известия Иркутского гос. ун.-та. Серия: математика. 2019. Т. 28. С. 3-20.
3. Белей Е.Г., Семенов А.А. О вычислительном поиске квазиортогональных систем латинских квадратов, близких к ортогональным системам. International Journal of Open Information Technologies. 2018. Т.6, No2, с. 22-30.
4. Zhang H. Combinatorial Designs by SAT solvers. Handbook of Satisfiability. 2009. Pp. 533-568.
5. Egan J., Wanless I. Enumeration of MOLS of Small Order// Mathematics of Computation. 2016 Vol. 85 N 298 799-824.
6. Kochemazov S., Zaikin O., Vatutin E., Belyshev A. Enumerating Diagonal Latin Squares of Order Up to 9 Journal of Integer Sequences. 2020 Vol. 23 Article 20.1.2.
7. Colbourn C.J., Dinitz J.H. Handbook of Combinatorial Designs. 2007.

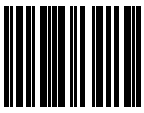
Дата выдачи задания/ Objectives issued on 19.05.2021**СОГЛАСОВАНО / AGREED:**Руководитель ВКР/
Thesis supervisor

Документ подписан	
Ульянцев Владимир Игоревич	
19.05.2021	

Ульянцев
Владимир
Игоревич

(эл. подпись)

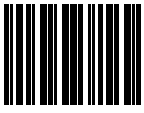
Задание принял к
исполнению/ Objectives
assumed by

Документ подписан	
Устинов Артем Павлович	
07.06.2021	

Устинов Артем
Павлович

(эл. подпись)

Руководитель ОП/ Head
of educational program

Документ подписан	
Станкевич Андрей Сергеевич	
07.06.2021	

Станкевич
Андрей
Сергеевич

(эл. подпись)

**Министерство науки и высшего образования Российской Федерации
ФЕДЕРАЛЬНОЕ ГОСУДАРСТВЕННОЕ АВТОНОМНОЕ ОБРАЗОВАТЕЛЬНОЕ
УЧРЕЖДЕНИЕ ВЫСШЕГО ОБРАЗОВАНИЯ
НАЦИОНАЛЬНЫЙ ИССЛЕДОВАТЕЛЬСКИЙ УНИВЕРСИТЕТ ИТМО
ITMO University**

**АННОТАЦИЯ
ВЫПУСКНОЙ КВАЛИФИКАЦИОННОЙ РАБОТЫ /
SUMMARY OF A GRADUATION THESIS**

Обучающийся/ Student

Устинов Артем Павлович

Наименование темы ВКР / Title of the thesis

Применение алгоритмов решения задачи о булевой выполнимости (SAT) к поиску комбинаторных структур, связанных с латинскими квадратами

Наименование организации, где выполнена ВКР/ Name of organization

Университет ИТМО

**ХАРАКТЕРИСТИКА ВЫПУСКНОЙ КВАЛИФИКАЦИОННОЙ РАБОТЫ/
DESCRIPTION OF THE GRADUATION THESIS**

1. Цель исследования / Research objective

Разработка новых алгоритмов и техник, повышающих эффективность применения современных SAT-решателей к задачам построения ортогональных и квазиортогональных систем латинских квадратов.

2. Задачи, решаемые в ВКР / Research tasks

а) разработка алгоритмов пропозиционального кодирования комбинаторных задач, связанных с латинскими квадратами; б) увеличение производительности SAT решателей на рассматриваемом классе задач за счет использования техник “Symmetry breaking”, “Incremental SAT”; в) построение квазиортогональных систем латинских квадратов при помощи комбинации SAT и специализированных алгоритмов локального поиска; г) аргументация эффективности разработанных алгоритмов в серии вычислительных экспериментов.

3. Краткая характеристика полученных результатов / Short summary of results/conclusions

В ходе выполнения ВКР были разработаны несколько алгоритмов и техник, позволяющих строить ортогональные и квазиортогональные системы латинских квадратов. В основе таких алгоритмов лежит использование SAT решателей и сопутствующих техник, таких как “Symmetry Breaking” и “Incremental SAT”. Основной результат работы состоит в комбинированном использовании алгоритмов решения SAT и специальных методов локального поиска. Все разработанные алгоритмы программно реализованы и их эффективность аргументирована вычислительными экспериментами.

4. Наличие публикаций по теме выпускной работы/ Have you produced any publications on the topic of the thesis

5. Наличие выступлений на конференциях по теме выпускной работы/ Have you

produced any conference reports on the topic of the thesis

6. Полученные гранты, при выполнении работы/ Grants received while working on the thesis

7. Дополнительные сведения/ Additional information

Обучающийся/Student

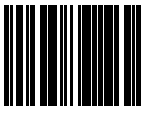
Документ подписан	
Устинов Артем Павлович	
16.05.2021	

(эл. подпись/ signature)

Устинов Артем
Павлович

(Фамилия И.О./ name
and surname)

Руководитель ВКР/ Head
of Graduate Project

Документ подписан	
Ульянцев Владимир Игоревич	
18.05.2021	

(эл. подпись/ signature)

Ульянцев
Владимир
Игоревич

(Фамилия И.О./ name
and surname)

СОДЕРЖАНИЕ

ВВЕДЕНИЕ	5
1. БАЗОВЫЕ ОПРЕДЕЛЕНИЯ И ПОСТАНОВКИ ИССЛЕДУЕМЫХ ЗАДАЧ.....	8
1.1. Латинские квадраты	8
1.2. Открытые проблемы конечной комбинаторики, связанные с латинскими квадратами	9
1.3. Методы решения SAT	10
1.4. Способы пропозиционального кодирования комбинаторных структур, связанных с латинскими квадратами	12
1.4.1. «One-hot» кодировка.....	12
1.4.2. Логарифмические кодировки	14
1.4.3. Кодирование ортогональных и квазиортогональных систем	15
1.5. Существующие методы решения задач поиска комбинаторных структур, связанных с латинскими квадратами	21
Выводы по главе 1	23
2. ОПИСАНИЕ РАЗРАБОТАННЫХ ТЕХНИК.....	24
2.1. Техника итеративного увеличения индекса ортогональности при помощи инкрементальности.....	24
2.2. Условия ограничения симметрии	26
2.3. Комбинированный подход (SAT + локальный поиск) для построения квазиортогональных систем.....	29
Выводы по главе 2	32
3. ЭКСПЕРИМЕНТАЛЬНЫЕ РЕЗУЛЬТАТЫ	33
3.1. Результаты по технике ограничения симметрии	33
3.2. Результаты по итеративной технике	35
3.3. Применение техники склеивания переменных.....	36
3.4. Результаты для комбинированного подхода.....	37
Выводы по главе 3	39
ЗАКЛЮЧЕНИЕ	40
СПИСОК ИСПОЛЬЗОВАННЫХ ИСТОЧНИКОВ	42
ПРИЛОЖЕНИЕ А. Квазиортогональные системы, полученные в экспериментах	45

ВВЕДЕНИЕ

Проблемы поиска комбинаторных структур с заданными свойствами возникают во многих практических областях: теория кодирования, криптография, планирование экспериментов, и другие. Латинские квадраты являются основой простого и естественного языка, на котором могут быть описаны свойства многих комбинаторных структур [1]. Произвольный латинский квадрат можно рассматривать как запись таблицы умножения в конечной группе (таблица Кэли). Таким образом, имеется глубокая связь между латинскими квадратами и различными объектами комбинаторно-алгебраической природы: ортогональные массивы, совершенные коды, схемы отношений, конечные проективные плоскости, графы специального вида и т.д.

Латинские квадраты порядка n существуют для любого натурального $n \geq 2$. Этот факт является простым следствием свойств таблицы Кэли аддитивной группы остатков от деления на n . Л. Эйлер определил т.н. греко-латинские квадраты, как произвольные квадраты, которые можно рассматривать как результат наложения друг на друга двух латинских квадратов при условии выполнения т.н. «свойства ортогональности». Общая задача существования греко-латинских квадратов уже оказалась слишком сложна для современной Эйлеру математики. Его гипотеза о несуществовании греко-латинских квадратов порядка n для всех $n = 2 \cdot k + 2, k \geq 2$ просуществовала недоказанной более 150 лет, и была опровергнута только в 1959 году.

В современной комбинаторике присутствует целый ряд открытых задач, связанных с латинскими квадратами. Эти задачи представляют своеобразный вызов для современных алгоритмических и вычислительных средств. Одной из таких задач является проблема существования тройки латинских квадратов порядка 10, любая пара которых дает греко-латинский квадрат. Данная задача является одной из самых известных открытых проблем конечной комбинаторики. Для такого рода задач естественным образом возникает идея применить к ним современные комбинаторные алгоритмы, востребованные в различных практических областях. Такие алгоритмы нацелены обычно на решение конкретных NP-трудных задач, имеющих важные промышленные приложения. Одной из наиболее известных задач такого рода является проблема булевой выполнимости (Boolean Satisfiability Problem), сокращенно обозначаемая как SAT.

Поскольку SAT NP-трудна, возможность решить ее в общем случае за полиномиальное время маловероятна. Однако современные SAT-решатели во многих индивидуальных случаях позволяют эффективно сокращать комбинаторную размерность соответствующих задач и, благодаря этому, используются в различных прикладных областях: в верификации, криптоанализе, биоинформатике, планировании и т.д. Применение комбинаторных алгоритмов, и SAT в частности, к новым трудным задачам имеет два позитивных аспекта. Во-первых, успешность такого опыта дает продвижение в решении рассматриваемой задачи. Во-вторых, адаптация алгоритма к поставленной задаче приводит к разработке новых техник, которые в дальнейшем, возможно, могут быть востребованы в других частных случаях базовой комбинаторной проблемы. В рамках данной дипломной работы предполагается исследовать возможности применения алгоритмов решения SAT к задачам существования конечных комбинаторных структур, связанных с латинскими квадратами, а именно, ортогональных и квазиортогональных систем латинских квадратов. Тематика исследований представляется актуальной, поскольку, с одной стороны, эти исследования способствуют продвижению в решении открытых проблем конечной комбинаторики, а с другой, их результатом могут стать новые алгоритмические техники решения SAT, которые в дальнейшем могут оказаться полезными в применении к другим трудным задачам. В результате были разработаны и реализованы техники, повышающие эффективность работы современных SAT-решателей в применении к задачам генерации греко-латинских квадратов (до порядка 10 включительно), а также к задачам построения квазиортогональных систем из трех латинских квадратов порядка 10 с итеративно увеличивающимся индексом ортогональности. К новым техникам, разработанным в рамках данной ВКР относятся:

- 1) инкрементальная техника увеличения индекса ортогональности для задачи поиска квазиортогональной тройки латинских квадратов порядка 10;
- 2) пропозициональная кодировка предиката нарушения симметрии (“Symmetry breaking predicate”), обеспечивающая дополнительные ограничения
- 3) техника, комбинирующая использование современных полных SAT-решателей и вариант локального поиска в специальном пространстве,

построенном с учетом особенностей задачи генерации квазиортогональной системы.

Работа состоит из введения, трех глав, заключения и списка литературы. Первая глава содержит краткий обзор предметной области, постановку исследуемой задачи, и анализ известных методов, применимых к этой задаче. Вторая глава содержит описание новых техник и алгоритмов, разработанных в процессе выполнения ВКР. В третьей главе приведены результаты вычислительных экспериментов, которые демонстрируют преимущества разработанных техник перед известными подходами. В заключении кратко подведены итоги исследований проведенных в настоящей работе.

ГЛАВА 1. БАЗОВЫЕ ОПРЕДЕЛЕНИЯ И ПОСТАНОВКИ ИССЛЕДУЕМЫХ ЗАДАЧ

1.1. Латинские квадраты

Определение 1. Латинский квадрат порядка n — это квадратная таблица размера $n \times n$, заполненная символами произвольного алфавита $\Sigma : |\Sigma| = n$ так, что в каждой строке и каждом столбце встречаются все символы этого алфавита [1]. Далее, без потери общности будем считать, что $\Sigma = \{1, \dots, n\}$.

Этот простой комбинаторный объект обладает массой глубоких математических свойств. Так, например, с помощью латинских квадратов можно описать таблицу умножения в произвольной конечной группе (т.н. таблицы Кэли). В свою очередь, конечные группы имеют очень широкое реальное практическое применение в теории кодирования и теории информации.

Еще в XVIII веке Л. Эйлер предложил строить так называемые греко-латинские квадраты на основе латинских. Мы можем наложить два латинских квадрата одного порядка один на другой. В итоге, мы получим квадрат того же порядка, но в его ячейках будут стоять не отдельные числа, а пары чисел из накладываемых квадратов. Про каждую пару будем говорить, что они состоят из двух компонент. Две пары будем считать различными, если они отличаются хотя бы по одной компоненте. Пусть A и B — два латинских квадрата, и $A|B$ — квадрат, полученный их наложением.

Определение 2. $A|B$ называется греко-латинским, если все пары, стоящие в его ячейках различны в указанном смысле, то есть встречаются все n^2 пар. Если квадраты A и B дают греко-латинский квадрат $A|B$, то говорят, что A и B ортогональны друг другу. Пару латинских квадратов, образующих греко-латинский квадрат, называют ортогональной системой мощности 2 или ортогональной парой.

Казалось бы, что вопрос о существовании греко-латинских квадратов произвольного порядка должен иметь какое-то относительно простое решение по аналогии с латинскими квадратами. Однако, легко заметить, что для $n = 2$ греко-латинских квадратов быть не может, а для $n = 3$ можно представить пример ортогональной пары (Рисунок 1).

Эйлер, пытаясь построить такие квадраты для $n = 6$ и $n = 10$, выдвинул гипотезу, что ортогональные пары латинских квадратов не существуют для любого $n = 4k + 2$. В 1900 году выяснилось, что для $n = 6$ греко-латинских

A =	<table border="1" style="display: inline-table; vertical-align: middle;"><tr><td>3</td><td>1</td><td>2</td></tr><tr><td>2</td><td>3</td><td>1</td></tr><tr><td>1</td><td>2</td><td>3</td></tr></table>	3	1	2	2	3	1	1	2	3
3	1	2								
2	3	1								
1	2	3								

B =	<table border="1" style="display: inline-table; vertical-align: middle;"><tr><td>2</td><td>1</td><td>3</td></tr><tr><td>1</td><td>3</td><td>2</td></tr><tr><td>3</td><td>2</td><td>1</td></tr></table>	2	1	3	1	3	2	3	2	1
2	1	3								
1	3	2								
3	2	1								

A B =	<table border="1" style="display: inline-table; vertical-align: middle;"><tr><td>3 2</td><td>1 1</td><td>2 3</td></tr><tr><td>2 1</td><td>3 3</td><td>1 2</td></tr><tr><td>1 3</td><td>2 2</td><td>3 1</td></tr></table>	3 2	1 1	2 3	2 1	3 3	1 2	1 3	2 2	3 1
3 2	1 1	2 3								
2 1	3 3	1 2								
1 3	2 2	3 1								

Рисунок 1 – Ортогональная пара порядка 3

квадратов действительно нет [2]. Позже, в 1959 году, общая гипотеза была опровергнута. Р. Ч. Боуз и С. С. Шрикханде нашли ортогональную пару латинских квадратов порядка 22 [3]. Через год, они же, вместе с Е. Т. Паркером, используя для решения этой задачи компьютеры, представили греко-латинский квадрат порядка 10 [4].

1.2. Открытые проблемы конечной комбинаторики, связанные с латинскими квадратами

Задачу поиска ортогональных пар латинских квадратов можно обобщить на ортогональные системы.

Определение 3. Ортогональной системой порядка n и мощности k назовем множество из k латинских квадратов порядка n . При этом любые два квадрата из этого множества образуют ортогональную пару. Обозначим через $N(n)$ максимальную мощность существующей ортогональной системы порядка n .

В 1896 году Э. Мур доказал, что $N(q) \geq q-1$ для любого простого числа, и что если $N(n) \geq k$ и $N(m) \geq k$, то $N(nm) \geq k$ [5]. Из этого следует, что $N(n) \geq q-1$, где q — минимальный простой делитель n . Можно показать, что при фиксированном n , $N(n) \leq n-1$. Ортогональная система из $n-1$ латинских квадратов порядка n , если она существует, дает так называемую «проективную плоскость порядка n ». [1].

Таким образом, мы можем обобщить задачу о существовании греко-латинских квадратов поиском числа $N(n)$ для ортогональной системы порядка n . Вопрос о существовании системы порядка 10 и мощности 3 является открытой проблемой конечной комбинаторики.

Эта задача интересна тем, что у нее относительно небольшая комбинаторная размерность. Но в то же время, для ее решения полным перебором она

требует нереалистично огромных вычислительных ресурсов. Исходя из этого, к данной задаче можно применять современные комбинаторные алгоритмы, которые в последствии могут иметь большие перспективы в ее решении.

В рамках настоящей работы, мы будем сводить задачу поиска ортогональных систем латинских квадратов к проблеме булевой выполнимости (SAT — Boolean SATisfiability), и соответственно использовать современные алгоритмы решения SAT.

1.3. Методы решения SAT

Большинство современных аппаратных средств, решающих проблему булевой выполнимости основаны на алгоритме CDCL (conflict-driven clause learning) [6]. Данный алгоритм решает более узкую задачу: выполнимость выражения, представленного в конъюнктивно нормальной форме (КНФ). Выражения в КНФ представляют из себя набор дизъюнкций литералов (пропозициональных переменных, либо их отрицания), объединенных операцией конъюнкции. Примером выражения в КНФ может послужить:

$$(x_0) \wedge (\neg x_0 \vee x_1) \wedge (\neg x_1) \wedge (x_1 \vee \neg x_3 \vee x_2),$$

где x_0, x_1, x_2, x_3 — некоторые пропозициональные переменные.

Известно, что любое булево выражение может быть сведено к КНФ, и существуют алгоритмы сведения произвольного выражения в КНФ за полиномиальное время [7].

Также, SAT можно представить в виде задачи псевдобулевой оптимизации (MaxSAT) функции определенного вида. Псевдобулевой [8] называется любая функция следующего вида:

$$f : \{0, 1\}^n \longrightarrow R \quad (1)$$

Требуется найти глобальный экстремум f на «булевом гиперкубе» $\{0, 1\}^n$. Можно заметить, что данная задача NP-трудна. Действительно, хорошо известную проблему MaxSAT можно поставить как задачу максимизации функции вида (1). Пусть C — произвольная КНФ над множеством булевых переменных X , $|X| = n$, которая состоит из m дизъюнктов. Сопоставим каждому вектору $\alpha \in \{0, 1\}^n$ значение функции f_C , равное числу дизъюнктов в C ,

которые истинны на векторе α . Очевидно, что

$$f_C : \{0, 1\}^n \longrightarrow \{0, 1, \dots, m\},$$

и C выполнима тогда и только тогда, когда существует такой $\alpha \in \{0, 1\}^n$, что $f_C(\alpha) = m$. Соответственно, зная максимум функции f_C на $\{0, 1\}^n$, мы знаем и ответ на вопрос о выполнимости КНФ C .

Далее мы разберем несколько алгоритмов псевдобулевой оптимизации, базирующихся на идеях локального поиска (т.н. local search-алгоритмы). В такого рода алгоритмах базовым является понятие системы окрестностей (Neighborhood structure) в $\{0, 1\}^n$. Система окрестностей задается при помощи функции окрестностей (см., например, [9]), определенной всюду на $\{0, 1\}^n$:

$$\aleph : \{0, 1\}^n \longrightarrow 2^{\{0,1\}^n} \quad (2)$$

Чаще всего для задания (2) используется метрика Хэмминга, в рамках которой для произвольной точки $\alpha \in \{0, 1\}^n$ ее окрестность Хэмминга радиуса $r, r \geq 1$, определяется следующим образом:

$$\aleph_r(\alpha) = \{\alpha' \in \{0, 1\}^n : d_H(\alpha, \alpha') \leq r\},$$

где $d_H(\alpha, \alpha')$ — расстояние Хэмминга между словами α и α' .

Базовой local search схемой псевдобулевой оптимизации является следующий алгоритм, известный как Hill Climbing (HC) [10]:

- 1) выбираем произвольным образом, например, случайно в соответствии с равномерным распределением на $\{0, 1\}^n$, стартовую точку $\alpha_0 \in \{0, 1\}^n$, вычисляем $f(\alpha_0)$; считаем α_0 текущей точкой, а $f(\alpha_0)$ текущим значением f ;
- 2) пусть $\alpha \in \{0, 1\}^n$ — текущая точка; обходим в некотором порядке $\aleph_1(\alpha) \setminus \{\alpha\}$, вычисляя для каждой точки α' из данного множества $f(\alpha')$. Если найдена такая точка $\alpha' \in \aleph_1(\alpha) \setminus \{\alpha\}$, что $f(\alpha') \geq f(\alpha)$, то перейти на шаг 3, в противном случае перейти на шаг 4;
- 3) $\alpha \leftarrow \alpha', f(\alpha) \leftarrow f(\alpha')$, перейти на шаг 1;
- 4) $(\alpha, f(\alpha))$ — локальный максимум функции f на $\{0, 1\}^n$ (поскольку для любой $\alpha' \in \aleph_1(\alpha) \setminus \{\alpha\}$ имеет место $f(\alpha') < f(\alpha)$). В этом случае ал-

горитм либо останавливается и выдает в качестве ответа $(\alpha, f(\alpha))$, либо запускает некоторую процедуру выхода из локального максимума.

Для выхода из локальных экстремумов можно дополнять приведенный выше базовый алгоритм различными техниками, основанными на эвристических и метаэвристических соображениях. Зачастую, выйдя из точки локального экстремума, можно оказаться в точке с худшим значением f . Такого сорта «выпрыгивания» из локальных экстремумов могут осуществляться в процессе поиска неоднократно. В этом случае обычно хранится точка с самым лучшим значением функции f , достигнутым за всю историю поиска. Такое значение называется рекордом (в англоязычной литературе Best known value, BKV). Современные техники выхода из локальных экстремумов позволяют даже при решении весьма трудных задач многократно улучшать рекорд в процессе поиска. Если некоторое число попыток выхода из локальных экстремумов не дает улучшения текущего рекорда f^* , достигнутого в точке α^* , то алгоритм останавливается и выдает в качестве ответа пару (α^*, f^*) .

Известно множество способов выхода из точек локальных экстремумов. Простейшими и весьма популярными являются метод имитации отжига (simulated annealing) [11], поиск с запретами (tabu search) [12], поиск с переменными окрестностями (variable neighborhood search) [13] и другие.

1.4. Способы пропозиционального кодирования комбинаторных структур, связанных с латинскими квадратами

Для того, чтобы свести к SAT задачу о поиске ортогональных систем латинских квадратов, необходимо научиться представлять наши искомые объекты булевыми векторами. Если исходный объект существует, то соответствующий ему булев вектор должен выполнять некоторую формулу. Если же комбинаторный объект с заданными свойствами отсутствует, то эта булева формула должна быть не выполнима в принципе на любом наборе векторов. Существует несколько способов построения булевых векторов и булевых формул, удовлетворяющим таким условиям.

1.4.1. «One-hot» кодировка

Самый простой из них — это так называемая «one-hot» кодировка [14]. Пусть, у нас есть алфавит Σ мощности n . Без потери общности можно считать, что это алфавит натуральных чисел от 1 до n ($\Sigma = \{1, \dots, n\}$). Свяжем с каж-

мым натуральным числом $i \in \{1, \dots, n\}$ булев вектор длины n , у которого в i -й позиции стоит 1, а в остальных компонентах — нули.

Например, в такой кодировке значения из алфавита $\{1, 2, 3\}$ будут иметь вид:

$$1 \longrightarrow \{1, 0, 0\}, 2 \longrightarrow \{0, 1, 0\}, 3 \longrightarrow \{0, 0, 1\}$$

В задачах поиска конечных комбинаторных структур мы не знаем, какое значение стоит на месте некоторого символа из конечного алфавита. Для того, чтобы записать условие, что «конкретный символ в “one-hot” кодировке может быть любым числом из алфавита $\Sigma = \{1, \dots, n\}$ » нам необходимо использовать некоторую булеву формулу, которая выполнима на любом векторе с весом Хэмминга 1, и невыполнима на всех других векторах из $\{0, 1\}^n$.

Чтобы построить такой предикат, свяжем с векторами значений из $\{0, 1\}^n$ вектор пропозициональных переменных $X = \{x_1, \dots, x_n\}$. Рассмотрим выражение:

$$\text{AtLeastOne}(X) = x_1 \vee x_2 \vee \dots \vee x_n \quad (3)$$

Оно будет выполняться для всех булевых векторов, у которых вес Хэмминга не меньше 1. Также рассмотрим выражение:

$$\text{AtMostOne}(X) = \bigwedge_{1 \leq i < j \leq n} (\neg x_i \vee \neg x_j) \quad (4)$$

Оно будет выполняться для всех булевых векторов, у которых вес Хэмминга меньше 2. Действительно, пусть у нас есть две пропозициональные переменные под индексами $k, j : k \neq l$ со значением 1, тогда дизъюнкт $\neg x_k \vee \neg x_j$ не выполним, и следовательно не выполнимо все выражение.

Таким образом совместим выражения (3) и (4), и запишем предикат:

$$\text{OnlyOneCardinality}(X) = (x_1 \vee x_2 \vee \dots \vee x_n) \wedge \bigwedge_{1 \leq i < j \leq n} (\neg x_i \vee \neg x_j) \quad (5)$$

Очевидно, что эта формула (5) выполнима для любого вектора $\{x_1, \dots, x_n\}$, в котором присутствует ровно одна единица, и не выполняется на всех векторах, в которых число единиц отличается. Таким образом, эта формула — в точности предикат, кодирующий векторы с весом Хэмминга 1 (всего n векторов).

Итак, для того, чтобы закодировать условие, что в ячейке квадрата с индексом i, j , находящейся на пересечении i -й строки и j -го столбца, может стоять любое число из $1, \dots, n$, нам понадобится множество булевых переменных $\{x_{i,j}^1, \dots, x_{i,j}^n\}$ и формула (5).

Для формулировки предиката, который кодирует условие, что в строке под некоторым номером i латинского квадрата стоят различные числа, мы рассмотрим векторы булевых переменных, соответствующие элементам этой строки: $X_{i,1} = \{x_{i,1}^1, \dots, x_{i,1}^n\}$, $X_{i,2} = \{x_{i,2}^1, \dots, x_{i,2}^n\}$, \dots , $X_{i,n} = \{x_{i,n}^1, \dots, x_{i,n}^n\}$. При этом, если в ячейке под индексом i, j записано число k , то $x_{i,j}^k = 1$ и все остальные компоненты вектора $X_{i,j}$ равны 0. Если выполняется условие того, что, например, значение 1 появляется в строке ровно один раз, то в векторе $\{x_{i,1}^1, \dots, x_{i,n}^1\}$ ровно одна компонента $x_{i,j}^1 = 1$, которая соответствует ячейке со значением 1.

Таким образом, если выражение $OOC(x_{i,1}^1, \dots, x_{i,n}^1)$ выполнено, то в i -й строке присутствует ровно один элемент с значением 1. Из полученного несложно записать выражение для условия того, что в i -й строке встречается каждое значение от 1 до n ровно один раз:

$$OOC(x_{i,1}^1, \dots, x_{i,n}^1) \wedge \dots \wedge OOC(x_{i,1}^n, \dots, x_{i,n}^n) \quad (6)$$

Построив подобным образом предикат для столбцов, и совместив все предикаты конъюнкцией, мы получим булево выражение (обозначим его за LS) в конъюнктивной нормальной форме, которое кодирует латинский квадрат. Решением LS будет являться некоторый булев вектор, из которого можно извлечь всю информацию, определяющую латинский квадрат.

1.4.2. Логарифмические кодировки

Другим, более экономным в плане числа кодирующих переменных, способом является логарифмическая кодировка [15]. В этом случае мы связываем с переменной, которая может принимать любое значение i из $\{1, \dots, n\}$, вектор булевых переменных длины $\lceil \log(n) \rceil$, являющийся двоичным представлением числа $i - 1$. Более точно, произвольному значению $m \in \{1, \dots, n\}$ сопоставляется вектор $\alpha = \{\alpha_1, \dots, \alpha_{\lceil \log(n) \rceil}\}$, $\alpha \in \{0, 1\}^{\lceil \log(n) \rceil}$, который определяет

коэффициенты двоичного представления $m - 1$:

$$m - 1 = \alpha_1 \cdot 2^0 + \alpha_2 \cdot 2^1 + \dots + \alpha_s \cdot 2^{\lceil \log(n) \rceil - 1}$$

Итак, обозначим, что $s = \lceil \log(n) \rceil$, и свяжем с латинским квадратом A множество булевых переменных X размера $n^2 \cdot s$. При этом разобьем X на n^2 непересекающихся подмножеств размера s , которые будут кодировать ячейки данного квадрата. Назовем такие подмножества $X_{i,j} = \{x_{i,j}^0, \dots, x_{i,j}^s\}$, и $X_{i,j}$ будет соответствовать непосредственно ячейке, находящейся на пересечении строки под номером i и столбца под номером j .

Для того, чтобы закодировать условие, что не существует повторяющихся элементов в строке 1, введем следующие обозначения. $\alpha_i(m)$ — значения i -го бита в битовом представлении числа. Запись x^λ означает литерал $\neg x$, если $\lambda = 0$, и литерал x , если $\lambda = 1$. Построим следующий предикат:

$$\Phi_1 = \bigwedge_{m=0}^{n-1} \bigvee_{i=1}^n \left((x_{1,i}^0)^{\alpha_0(m)} \wedge \dots \wedge (x_{1,i}^s)^{\alpha_s(m)} \right) \quad (7)$$

Он будет выполняться, если в множествах вида $X_{1,j}$ для j от 1 до n присутствуют все битовые представления чисел от 0 до $n - 1$. Если же в таких множествах нет хотя бы одного битового представления числа, то он будет не выполнен. Таким образом, связав значение числа k , битовое представление которого представляет множество $X_{1,j}$ со значением $k + 1$ в ячейке квадрата в первой строке и j -м столбце, мы получим необходимое нам свойство.

Построим аналогичным образом предикаты для оставшихся строк Φ_2, \dots, Φ_n , а также для всех столбцов Ψ_1, \dots, Ψ_n , и, соединив их все конъюнкцией, мы получим предикат, который выполняется тогда и только тогда, когда связанный с множеством X квадрат A — латинский.

В настоящей работе данная кодировка показала себя на практике хуже, чем кодировка «one-hot», поэтому далее будем считать, что если не сказано обратное, то используется «one-hot» кодировка.

1.4.3. Кодирование ортогональных и квазиортогональных систем

После введения способов кодирования квадратов в булевы выражения, разберем как закодировать ортогональные пары и рассмотрим более общую

задачу кодирования так называемых «квазиортогональных» систем латинских квадратов.

Итак, пусть A, B — произвольная пара латинских квадратов порядка n . Рассмотрим квадрат $A|B$, полученный наложением A и B . Пронумеруем ячейки первого и второго квадратов в некотором порядке: например, нумеруем ячейки первой строки слева направо, потом продолжаем нумерацию, проходя вторую строку слева направо и т.д. Будем обозначать содержимое ячеек первого квадрата как $a_i, i \in \{1, \dots, n^2\}$, а содержимое ячеек второго квадрата как $b_i, i \in \{1, \dots, n^2\}$, подразумевая, что i пробегает ячейки квадратов в соответствии с фиксированным порядком. Будем считать для $i, j \in \{1, \dots, n^2\}, i \neq j$, пары (a_i, b_i) и (a_j, b_j) различными, если $a_i \neq a_j$ или $b_i \neq b_j$.

Далее нам понадобится, в некотором роде, «эталонное» множество всех различных пар вида $(a_i, b_i), i \in \{1, \dots, n^2\}$, которое мы будем использовать для кодирования условий, связанных с ортогональностью. Фактически нам надо перечислить все различные пары в некотором фиксированном порядке, который можно считать «удобным», например, с точки зрения некоторой простой закономерности. Обозначим через P^* следующее множество пар, упорядоченное, по сути, в соответствии с лексикографическим порядком:

$$P^* = (1, 1), (1, 2), \dots, (1, n), (2, 1), \dots, (2, n), \dots, (n, 1), \dots, (n, n) \quad (8)$$

Свойство ортогональности пары квадратов A и B означает, что не существует такой пары индексов $i, j \in \{1, \dots, n^2\}$, что $(a_i, b_i) = (a_j, b_j)$. Как было указано выше, условие того, что в строке, закодированной способом «one-hot», присутствует не больше одного конкретного значения, можно выразить предикатом (4). Такой предикат легко переформулировать для пар значений. Пусть есть два множества $A^* = \{a_1, \dots, a_{n^2}\}$ и $B^* = \{b_1, \dots, b_{n^2}\}$, элементы которых $a_i = \{a_i^1, \dots, a_i^n\}$, $b_i = \{b_i^1, \dots, b_i^n\}$ представляют собой векторы булевых переменных веса Хэмминга 1. Сопоставим с множествами A^*, B^* пару латинских квадратов A, B , ячейки которых занумерованы, а a_i, b_i — соответствуют ячейке под номером i у A и B соответственно. Если значение ячейки равно $k \in \{1, \dots, n\}$, то $a_i^k = 1$, а все остальные компоненты a_i равны. Аналогично для квадрата B . Тот факт, что пара (k_1, k_2) не встречается в квадрате более од-

ного раза можно закодировать следующим выражением, находящемся в КНФ:

$$\bigwedge_{i,j \in \{1, \dots, n^2\}, i \neq j} \left(\neg a_i^{k_1} \vee \neg b_i^{k_2} \vee \neg a_j^{k_1} \vee \neg b_j^{k_2} \right) \quad (9)$$

Данный предикат будет выполняться на таких квадратах A и B , в которых пара (k_1, k_2) встречается в квадрате $A|B$ не более одного раза. Перечислив все возможные значения таких пар из (8) и объединив их конъюнкцией, мы получим предикат, кодирующий условие ортогональности пары A и B . С помощью данной кодировки современные SAT-решатели могут найти ортогональную пару порядка 10 за время от 20 до 30 минут, что означает, что задача достаточно трудная. Но каждая такая пара опровергает гипотезу Эйлера, что само по себе достижение.

Однако, вычисление попарно ортогональных троек, начиная с порядка 9, занимает существенное количество времени. Поэтому, попробуем ослабить условие ортогональности пары квадратов и обобщим задачу поиска ортогональных систем к поиску так называемых квазиортогональных систем.

Другое свойство, которое следует из условия ортогональности пары квадратов A и B , означает, что в квадрате $A|B$ встречается каждая пара из (8). Для того, чтобы закодировать тот факт, что, например, первая пара из (8) будет встречаться среди пар вида (a_i, b_i) , $i \in \{1, \dots, n^2\}$, отметим, что произвольная конъюнкция литералов выполнима на одном единственном наборе значений входящих в нее переменных. Например, конъюнкция $x_1 \wedge \neg x_2 \wedge \neg x_3$ принимает значение 1 только на наборе $x_1 = 1, x_2 = 0, x_3 = 0$. По сути, мы видим, что формула выполнима только на векторе, который кодирует (в нашей кодировке) число 1 из алфавита $\{1, 2, 3\}$. Тогда тот факт, что, скажем, пара $(1, 1)$ встречается в ячейках квадратов A и B с номером i (притом что общее число ячеек в квадрате есть n^2) будет равносильно истинности следующей формулы:

$$(a_i^1 \wedge \neg a_i^2 \wedge \dots \wedge \neg a_i^n) \wedge (b_i^1 \wedge \neg b_i^2 \wedge \dots \wedge \neg b_i^n) \quad (10)$$

Здесь используются те же обозначения, что и в (9).

Тот факт, что пара $(1, 1)$ может иметь любой номер $i \in \{1, \dots, n^2\}$, означает, что для кодирования этого мы должны рассмотреть n^2 формул вида (10)

и связать все такие формулы дизъюнкцией:

$$\bigvee_{i \in \{1, \dots, n^2\}} (a_i^1 \wedge \neg a_i^2 \wedge \dots \wedge \neg a_i^n) \wedge (b_i^1 \wedge \neg b_i^2 \wedge \dots \wedge \neg b_i^n) \quad (11)$$

Построим аналогичные формулы для всех пар из (8). Обозначим эти формулы через $\Phi_j, j \in \{1, \dots, n^2\}$. Свяжем с каждым $j \in \{1, \dots, n^2\}$, рассматривая его как номер пары в P^* , булеву переменную, которую обозначим через χ_j . Будем считать, что χ_j кодирует значение предиката, который определяется следующим образом: $\chi_j = 1$, если пара из P^* с номером j встречается в некоторой ячейке квадрата $A|B$, в противном случае $\chi_j = 0$. Очевидно, что $\chi_j = 1$ тогда и только тогда, когда формула Φ_j принимает значение 1. Таким образом, истинна следующая булева формула:

$$\chi_j \equiv \Phi_j$$

где через \equiv обозначена логическая эквивалентность. $\chi_{A|B} = (\chi_1, \dots, \chi_{n^2})$ Вектор $\chi_{A|B}$ везде далее называем вектором маркировки.

Определение 4. Для двух заданных латинских квадратов A и B число различных пар из P^* , присутствующих в ячейках квадрата $A|B$, назовем индексом ортогональности пары A, B . Если r — индекс ортогональности пары A, B , то будем также говорить, что A и B образуют квазиортогональную пару индекса r .

Из определений, очевидным образом следует справедливость следующих утверждений:

Утверждение 1. Квазиортогональная пара латинских квадратов с индексом ортогональности $r = n^2$ является ортогональной парой, другими словами дает греко-латинский квадрат.

Утверждение 2. Индекс ортогональности произвольной пары латинских квадратов A и B равен весу Хэмминга вектора маркировки.

Таким образом квадраты A и B образуют ортогональную пару тогда и только тогда, когда вес Хэмминга вектора маркировки равен n^2 . Научимся строить формулы в КНФ, кодирующие факт существования латинских квадратов порядка n , которые образуют квазиортогональную пару индекса не менее r для фиксированного $r : 1 \leq r \leq n^2$.

Итак, предположим, что у нас есть формулы, кодирующие условие, что A и B латинские квадраты порядка n . Пусть X и Y — множества булевых переменных, использованных в квадратах A и B соответственно. Обозначим через $C(X)$ и $C(Y)$ — булевы формулы в КНФ, кодирующие квадраты A и B . Тогда, для фиксированного $r \in \{1, \dots, n^2\}$ квадраты A и B образуют квазиортогональную систему индекса $\geq r$ тогда и только тогда, когда выполнена следующая формула:

$$C(X) \vee C(Y) \wedge (\Phi^1 \equiv \chi_1) \wedge \dots \wedge (\Phi^{n^2} \equiv \chi_{n^2}) \wedge (w_H(\chi) \geq r) \quad (12)$$

В (12) через « $w_H(\chi) \geq r$ » обозначена булева формула от переменных $\chi_1, \dots, \chi_{n^2}$, которая истинна тогда и только тогда, когда вектор $\chi_{A|B}$ содержит $\geq r$ единиц. Ее зададим немного позже. Рассмотрим сперва формулы вида $\Phi^1 \equiv \chi$. Они не находятся в КНФ и более того, формула Φ^1 — это выражение в дизъюнктивно нормальной форме (ДНФ). Для перевода произвольной булевой формулы в КНФ за полиномиальное, и даже за линейное от длины формулы, время широко используются преобразования Цейтина [7]. Это достигается при помощи введения дополнительных переменных. Общий смысл заключается в следующем. Рассмотрим произвольную формулу вида $f(G, H)$, где G, H — произвольные булевы формулы, а f — некоторая «внешняя» булева операция над формулами G, H . Предположим, что $f(G, H)$ — формула над множеством булевых переменных X . Введем новую булеву переменную $u, u \notin X$ и рассмотрим следующую формулу

$$(G \equiv u) \wedge f(u, H) \quad (13)$$

Формула $f(u, H)$ в (13) получена из $f(G, H)$ в результате замены вхождений подформулы G новой переменной u . Формула $G \equiv u$ в большинстве практических случаев оказывается такой размерности по числу переменных, что ее можно эффективно представить в КНФ, используя обычную таблицу истинности.

Вернемся к формулам вида $\Phi^i \equiv \chi_i$. Конкретно рассмотрим формулу $\Phi^1 \equiv \chi_1$, расписав левую часть с учетом (11):

$$\bigvee_{i \in \{1, \dots, n^2\}} (a_i^1 \wedge \neg a_i^2 \wedge \dots \wedge \neg a_i^n) \wedge (b_i^1 \wedge \neg b_i^2 \wedge \dots \wedge \neg b_i^n) \equiv \chi_1 \quad (14)$$

Теперь перейдем от (14) к КНФ, используя преобразования Цейтина. Для начала введем n^2 новых булевых переменных:

$$u_1 \equiv (a_1^1 \wedge \neg a_1^2 \wedge \dots \wedge \neg a_1^n) \wedge (b_1^1 \wedge \neg b_1^2 \wedge \dots \wedge \neg b_1^n) \quad (15)$$

и переменные u_2, \dots, u_{n^2} по аналогии. Для того, чтобы записать (15) в КНФ необходимо использовать КНФ представление логической эквивалентности: $F \equiv G$ в КНФ — это $(F \vee \neg G) \wedge (\neg F \vee G)$. Используя правило де Моргана и дистрибутивность дизъюнкции относительно конъюнкции, получим КНФ для (15):

$$(u_1 \vee \neg a_1^1 \vee a_1^2 \vee \dots \vee a_1^n \vee \neg b_1^1 \vee b_1^2 \vee \dots \vee b_1^n) \wedge \\ \wedge (\neg u_1 \vee a_1^1) \wedge (\neg u_1 \vee b_1^1) \wedge \bigwedge_{i=2}^n (\neg u_1 \vee \neg a_1^i) \wedge (\neg u_1 \vee \neg b_1^i)$$

Обозначим эту формулу через ψ_1 . Аналогичным образом поступаем относительно всех других формул вида $\psi^i, i \in \{2, \dots, n^2\}$. Как итог, имеем формулы $\psi_1, \dots, \psi_{n^2}$. Заменим формулу (14) на следующее выражение:

$$\psi_1 \wedge \dots \wedge \psi_{n^2} \wedge ((u_1 \vee \dots \vee u_{n^2}) \equiv \chi_1) \quad (16)$$

В (16) остается только преобразовать $((u_1 \vee \dots \vee u_{n^2}) \equiv \chi_1)$ в КНФ, что также легко делается с помощью формулы для логической эквивалентности, правил де Моргана и дистрибутивности дизъюнкции относительно конъюнкции. Как итог имеем n^2 формул в КНФ, которые мы обозначим как $C_{\chi_1}, \dots, C_{\chi_{n^2}}$. С учетом этого подставим их в (12):

$$C(X) \wedge C(Y) \wedge C_{\chi_1} \wedge \dots \wedge C_{\chi_{n^2}} \wedge (w_H(\chi) \geq r) \quad (17)$$

Для получения итоговой формулы в представлении КНФ нам необходимо закодировать в КНФ условие « $W_H(\chi) \geq r$ ». Существует множество способов это сделать. Для этого будем использовать кодировку, предложенную К. Синцем [16]. Для кодирования условия $w_H(x_1, \dots, x_n) \geq k$ в КНФ, предлагается построить схему из вычисления частичных сумм нулевых значений в векторе x , $s_i = \sum_{j=1}^i \neg x_j$ для всех i от 1 до n . Значения s_i представлены, как унарные числа, и соответствуют векторам булевых переменных $\{s_{i,1}, \dots, s_{i,k}\}$.

Конечная КНФ счетчика представляет собой конъюнкцию следующих выражений:

$$\left. \begin{array}{l} (x_1 \vee s_{1,1}) \\ (\neg s_{1,j}) \quad \text{для } 1 < j \leq k \\ (x_i \vee s_{i,1}) \\ (x_i \vee \neg s_{i-1,j-1} \vee s_{i,j}) \\ (\neg s_{i-1,j} \vee s_{i,j}) \\ (x_i \vee \neg s_{i-1,k}) \\ (x_n \vee \neg s_{n-1,k}) \end{array} \right\} \text{ для } 1 < j \leq k \left. \vphantom{\begin{array}{l} (x_1 \vee s_{1,1}) \\ (\neg s_{1,j}) \\ (x_i \vee s_{i,1}) \\ (x_i \vee \neg s_{i-1,j-1} \vee s_{i,j}) \\ (\neg s_{i-1,j} \vee s_{i,j}) \\ (x_i \vee \neg s_{i-1,k}) \\ (x_n \vee \neg s_{n-1,k}) \end{array}} \right\} \text{ для } 1 < i < n \quad (18)$$

Подставив формулу вида (18) вместо $w_H \chi_{A|B} \geq r$ в (17) получим предикат, выполнимый для квазиортогональных пар с индексом ортогональности $\geq r$.

1.5. Существующие методы решения задач поиска комбинаторных структур, связанных с латинскими квадратами

На текущий момент, существует множество вычислительных способов решения задачи поиска ортогональных и квазиортогональных систем. Самый успешный результат был достигнут в работе Д. Иган и Я. Уонлесса [17]. В ней они нашли тройку квадратов порядка 10, где две пары квадратов были ортогональны и третья пара имела индекс ортогональности 92. Для этого они использовали эффективный переборный алгоритм с отсечением изоморфных групп и построения всех ортогональных пар по заданным квадратам. Как показано в работе [18], число различных ортогональных пар латинских квадратов порядка 10 равно примерно 10^{15} . Из этого можно сделать вывод, что подход полного перебора ортогональных пар для поиска ортогональной тройки не подходит из-за слишком большой вычислительной стоимости. С другой стороны, непонятно,

в каком направлении можно развить подход из работы [17], чтобы увеличить размерность индекса ортогональности, или доказать, что квазиортогональной тройки с большим индексом ортогональности не существует.

Поиск ортогональных систем также можно свести к проблемам целочисленного линейного программирования (ILP — Integer Linear Programming) и линейного программирования на ограничениях (CLP — Constraint Linear Programming). Данные задачи являются NP-трудными, и также, как и для SAT проблемы имеют ряд алгоритмов, которые находят решения применимых на практике задач за весьма эффективное время. Данный подход использовали [19, 20]. В своей работе, они свели задачу перебора латинских квадратов заданного порядка n к задаче CLP, а задачу поиска ортогональных систем к задаче ILP. Так, перебирая квадраты порядка n , они искали ортогональные пары и ортогональные системы с помощью ILP-решателя с фиксацией одного из квадратов в системе.

Также, они в своей работе предложили предикат ограничения симметрии, основанный на приведении ортогональных систем латинских квадратов к каноничной форме с накладыванием условия на первый столбец второго квадрата в системе. Это позволяет значительно ограничить число перестановок этого столбца. Подобный предикат ограничения симметрии был представлен в работе [21]. Они использовали аналогичное ограничивающее условие на перестановки, но для второго столбца первого квадрата. Данные условия рассмотрены в разделе 2.2, и там же представлен способ их кодирования в задачу SAT, представимой в КНФ.

Сведение задачи поиска квазиортогональных систем к проблеме булевой выполнимости рассматривалась в работах [15, 22]. В них описаны способы кодирования данной проблемы в КНФ с помощью логарифмических и «one-hot» кодировок, и также представлены вычислительные результаты работы различных SAT-решателей на данных кодировках. Так, в результате экспериментов на мощной вычислительной машине была найдена квазиортогональная тройка порядка 10 с индексом ортогональности 82.

Не смотря на то, что на данный момент SAT-подход не позволяет построить системы тройки латинских квадратов порядка 10, аналогичных по индексу ортогональности тройке Иган-Уонлесса. Однако быстрый прогресс в развитии SAT-решателей дает надежду на то, что в ближайшее время методы, основан-

ные на SAT, превзойдут известные подходы к решению таких задач. Такие выводы можно сделать ввиду того, что только за последние 3 года удалось повысить производительность SAT-решателей на задачах поиска греко-латинских квадратов 10 порядка в десятки раз. Данный факт следует из результатов вычислительных экспериментов, приведенных в главе 3 настоящей работы.

Выводы по главе 1

В данной главе была представлена задача поиска систем попарно-ортогональных латинских квадратов, а также была рассмотрена более общая задача поиска квазиортогональных систем латинских квадратов с заданным минимальным индексом ортогональности. Сведение данных задач к проблеме булевой выполнимости формулы в КНФ можно произвести с помощью способа описанного в разделе 1.4.3, и итоговая формула имеет вид (15). К данной формуле можно применить алгоритмы решения SAT и извлечь всю необходимую информацию для построения систем. В разделе 1.5 рассмотрены существующие методы решения представленных задач, связанных с латинскими квадратами.

ГЛАВА 2. ОПИСАНИЕ РАЗРАБОТАННЫХ ТЕХНИК

В данной главе приводятся основные теоретические выкладки, и дается описание методов и техник, применимых для задач поиска ортогональных и квазиортогональных систем латинских квадратов, которые были разработаны в рамках настоящей работы.

2.1. Техника итеративного увеличения индекса ортогональности при помощи инкрементальности

Некоторые современные SAT-решатели на основе CDCL (например `cryptominisat` [23]) после нахождения выполняющего набора некоторого выражения, которое представлено в КНФ, могут сохранять информацию о полученных на данный момент литералах и дизъюнктах. Такая техника называется инкрементальной и она позволяет эффективно решать задачу инкрементальной булевой выполнимости (incremental SAT): Если известно, что выражение C в КНФ выполнимо, то будет ли выполнима функция $C \wedge a$, где a — некоторый дизъюнкт.

Рассмотрим, как можно использовать данную технику в формулировке задачи поиска квазиортогональных систем порядка n мощности k и индекса ортогональности r . Вместо того, чтобы решать с «нуля» соответствующее этой задаче булево выражение, можно решить эту же задачу для меньших индексов и «переиспользовать» решение и информацию об обученных дизъюнктах инкрементального SAT-решателя. Так, решение задачи поиска квазиортогональной системы некоторого индекса с использованием инкрементального подхода может занимать суммарно меньше времени, чем решение формулы, соответствующей системе с данным индексом.

В формуле (17), для кодирования условия, что индекс ортогональности пары латинских квадратов $\geq r$, мы использовали счетчик Синца на векторе маркировок (18). Обозначим $q = n^2 - r$, и напомним, что такой счетчик строится из $n^2 - 1$ частичных сумм нулевых значений в векторе маркировок $s_i = \sum_{j=1}^{n^2-1} \neg \chi_{ij}$. Каждая из таких сумм представляет собой унарное число и связана с битовыми векторами $s_i = \{s_{i,1}, \dots, s_{i,q}\}$. То есть s_i имеет такой вид, что все $s_{i,1}, \dots, s_{i,k}$ имеют значение 1 для некоторого k от 1 до некоторого k , а все последующие $s_{i,k+1}, \dots, s_{i,q} = 0$. Сумма s_{n^2} представляет общее число нулевых бит вектора маркировок и может быть опущена. Для поддержания всех этих свойств используется формула (18).

Мы можем научиться итеративно увеличивать индекс ортогональности пары, инкрементально добавляя новые дизъюнкты на формулу счетчика. Для того, чтобы перейти от условия $w_H(\chi) \geq r$ к $w_H(\chi) \geq r + 1$, нам необходимо усилить ограничение на сумму s_{n^2} представленного счетчика Синца. Рассмотрим последний дизъюнкт в (18). Он задает условие, что сумма $s_{n^2-1} \leq q$, либо $s_{n^2-1} = q$ и $\chi_{n^2} = 1$. Добавим в формулу следующий дизъюнкт:

$$(s_{n^2-1, q-1} \vee \neg \chi_{n^2}) \wedge (\neg s_{n^2-1, q}).$$

Добавив его в выражение, мы ограничиваем значение суммы $s_{n^2-1} \leq q - 1$, а также задаем условие, что если $s_{n^2-1} = q - 1$, то $\chi_{n^2} = 1$. Таким образом, итоговая сумма s_{n^2} не может быть больше $q - 1$, и соответственно $w_H(\chi) \geq r + 1$.

Итак, мы можем построить следующую схему. Кодировем квазиортогональную систему из k латинских квадратов порядка n с индексом ортогональности 1 и получаем формулу C_1 . В формуле будет присутствовать $k \cdot (k - 1) / 2$ формул вида (18). Обозначим их за $S_1, \dots, S_{k \cdot (k-1)/2}$, а переменные, отвечающие за частичные суммы обозначим за $s_{i,j}^1, \dots, s_{i,j}^{k \cdot (k-1)/2}$. Также вектора маркировок, используемые в счетчике S_j пометим как χ^j . Далее приступаем к итеративному увеличению индекса. На каждой i -й итерации мы будем решать текущую формулу C_i , а затем строить C_{i+1} следующим образом:

$$C_{i+1} = C_i \wedge \bigwedge_{j=1}^{k \cdot (k-1)/2} \left(s_{n^2-i, q-i}^j \vee \neg \chi_{n^2}^j \right) \wedge \left(\neg s_{n^2-i, q-i-1}^j \right) \quad (19)$$

В итоге, на i -й итерации будет решаться формула, задающая квазиортогональную систему порядка n с индексом $\geq i$. На последней n^2 -й итерации мы строим систему C_{n^2} немного отличающимся от (19) образом:

$$C_{n^2} = C_{n^2} \wedge \bigwedge_{j=1}^{k \cdot (k-1)/2} \left(\neg s_{n^2-1, 1}^j \right)$$

Последняя формула C_{n^2} будет кодировать ортогональные системы латинских квадратов порядка n , и решив это булево выражение мы сможем найти пример такой ортогональной системы.

2.2. Условия ограничения симметрии

Во многих комбинаторных задачах при их сведении к SAT в случае существования хотя бы одного набора, выполняющего получаемую формулу, существует огромное число других выполняющих наборов, которые соответствуют тому же решению исходной задачи. Считается [24], что такая ситуация снижает эффективность современных SAT-решателей (по крайней мере тех, которые основаны на алгоритме CDCL). Техники, уменьшающие число решений-дубликатов в выполнимых формулах, называются техниками ограничения симметрии (Symmetry Breaking Techniques).

При сведении к SAT задач поиска латинских квадратов описанная проблема возникает явным образом: из-за того, что существует большое число изотопных квадратов с одинаковыми параметрами, число выполняющих наборов получаемой КНФ может быть колоссальным. Известен ряд техник ограничения симметрии, увеличивающих эффективность поиска в задачах, связанных с латинскими квадратами: [20, 21] и ряд других.

Для задач построения ортогональных пар и троек латинских квадратов различных порядков были построены и протестированы SAT кодировки условий нарушения симметрии из работ [20] и [21]. Лучшие экспериментальные результаты удалось получить для предикатов, описанных в [21]. В основе соответствующих условий лежит следующий теоретический факт.

Теорема 1 ([21]). Произвольная ортогональная система, образованная k , $2 \leq k \leq n-1$, латинскими квадратами $LS_i, i \in \{1, \dots, k\}$, порядка n может быть приведена к следующей канонической форме:

- 1) элементы первой строки в каждом квадрате $LS_i, i \in \{1, \dots, k\}$, упорядочены в лексикографическом порядке;
- 2) элементы первого столбца в LS_1 упорядочены в лексикографическом порядке;
- 3) первые две строки LS_1 задают перестановку на множестве $1, \dots, n$, имеющую следующее циклическое представление:

$$(1 \ 2 \ \dots \ k_1) (k_1 + 1 \ \dots \ k_2) \dots (k_{i-1} + 1 \ \dots \ k_i) (k_i + 1 \ \dots \ n) \quad (20)$$

Длины циклов (20) подчинены следующему условию:
 $2 \leq k_1 \leq k_2 - k_1 \leq \dots \leq k_i - k_{i-1} \leq n - k_i$.

Дополнительно прокомментируем, что для получения вариантов заполнения второго столбца квадрата LS_1 необходимо по конкретному циклическому представлению (20) построить перестановку в виде обычной таблицы из двух строк. Тогда первую строку этой таблицы можно рассматривать как первую строку LS_1 , а вторая строка даст вариант заполнения второй строки LS_1 .

Пример: произведение циклов $(1\ 2)(3\ 4)(5\ 6)(7\ 8)(9\ 10)$ задает следующую перестановку:

$$\begin{array}{cccccccccc} 1 & 2 & 3 & 4 & 5 & 6 & 7 & 8 & 9 & 10 \\ 2 & 1 & 4 & 3 & 6 & 5 & 8 & 7 & 10 & 9 \end{array}$$

Также можно заметить связь между числом перестановок размера n с циклической записью такого вида с числом разбиений числа n на суммы чисел, больших 1. Действительно, элементы перестановки в циклической записи фиксированы, и могут меняться лишь количество и длины циклов. Мы можем представить, что длины циклов — это слагаемые в разбиении числа, которые в сумме будут давать n . Последовательность из чисел, равных числу таких разбиений растет медленнее, чем числа Фибоначчи [25] и для рассматриваемых задач, связанных с латинскими квадратами небольших порядков не будет превосходить нескольких десятков.

Как пример, на Рисунке 2 показаны все возможные циклические представления перестановок, связывающих первые две строки LS_1 для $n = 10$, а также показана их связь с разбиением числа 10 на числа, отличные от 1. Заметим, что при фиксации первой строки LS_1 в виде $(1\ 2\ \dots\ 10)$ имеется 12 различных вариантов для второй строки. Все эти варианты могут быть закодированы в SAT с использованием техники преобразований Цейтина [7]. Для этой цели введем n множеств $S_i, i \in \{1, \dots, n\}$, каждое из которых содержит n булевых переменных. Каждое множество $S_i, i \in \{1, \dots, n\}$ отвечает за кодирование ячейки с номером i второй строки LS_1 . Таким образом, имеем n^2 булевых переменных, значения которых определяют конкретные варианты вида второй строки LS_1 с учетом Теоремы 1. Так, для $n = 10$ кодирование условия нарушения симметрии для второй строки потребует 100 переменных. Каждое их

1. (1 2) (3 4) (5 6) (7 8) (9 10)	\longleftrightarrow	$10 = 2 + 2 + 2 + 2 + 2$
2. (1 2) (3 4) (5 6) (7 8 9 10)	\longleftrightarrow	$10 = 2 + 2 + 2 + 4$
3. (1 2) (3 4) (5 6 7) (8 9 10)	\longleftrightarrow	$10 = 2 + 2 + 3 + 3$
4. (1 2) (3 4) (5 6 7 8 9 10)	\longleftrightarrow	$10 = 2 + 2 + 6$
5. (1 2) (3 4 5) (6 7 8 9 10)	\longleftrightarrow	$10 = 2 + 3 + 5$
6. (1 2) (3 4 5 6) (7 8 9 10)	\longleftrightarrow	$10 = 2 + 4 + 4$
7. (1 2) (3 4 5 6 7 8 9 10)	\longleftrightarrow	$10 = 2 + 8$
8. (1 2 3) (4 5 6) (7 8 9 10)	\longleftrightarrow	$10 = 3 + 3 + 4$
9. (1 2 3) (4 5 6 7 8 9 10)	\longleftrightarrow	$10 = 3 + 7$
10. (1 2 3 4) (5 6 7 8 9 10)	\longleftrightarrow	$10 = 4 + 6$
11. (1 2 3 4 5) (6 7 8 9 10)	\longleftrightarrow	$10 = 5 + 5$
12. (1 2 3 4 5 6 7 8 9 10)	\longleftrightarrow	$10 = 10$

Рисунок 2 – Циклические представления перестановок размера 10 и их связь с разбиениями на сумму чисел, отличных от 1

значение из 12 возможных перечисленных выше кодируется конъюнкцией 100 литералов, и как итог получим набор конъюнкций: C_1, \dots, C_{12} .

Пусть теперь C^* — исходная КНФ кодировка системы из k ортогональных квадратов 10-го порядка (например, $k = 3$). Тогда формула, кодирующая существование ортогональной системы латинских квадратов 10-порядка, в которое учтено нарушение симметрии, обеспечиваемое Теоремой 1, выглядит следующим образом:

$$C^* \wedge (C_1 \vee \dots \vee C_{12}) \quad (21)$$

Для перехода от (21) к КНФ необходимо использовать преобразования Цейтина и ввести 12 переменных по следующему правилу:

$$\phi_j \equiv C_j, j \in \{1, \dots, 12\}$$

Добавив эти выражения в (21) и сделав преобразования Цейтина, получим предикат, кодирующий представленное условие нарушения симметрии для квадратов порядка 10.

2.3. Комбинированный подход (SAT + локальный поиск) для построения квазиортогональных систем

Базовая идея описываемого далее подхода состоит в следующем. Мы можем, используя, например, какой-либо CDCL решатель, построить некоторую квазиортогональную систему латинских квадратов, а затем попытаться найти систему с большим индексом ортогональности при помощи алгоритмов локального поиска. Для этой цели мы рассматриваем задачу поиска квазиортогональной системы латинских квадратов как задачу псевдоболевой оптимизации. Далее будут рассмотрены два варианта такой постановки. В первом варианте задача нахождения квазиортогональной системы сводится к MaxSAT, а во втором — рассматривается, по сути, как задача в ограничениях без перехода к какой-либо унифицированной форме.

Начнем с MaxSAT постановки. Пусть C — КНФ, кодирующая задачу поиска системы k квазиортогональных латинских квадратов порядка n , и пусть X — множество булевых переменных, которые встречаются в C . Выделим в X подмножество \tilde{X} , которое образовано переменными, кодирующими строки и столбцы квадратов LS_1, \dots, LS_k ($2 \leq k \leq n - 1$). Все остальные переменные в C функционально зависят от переменных из \tilde{X} , поскольку вводятся в результате перехода от выражений вида

$$y \equiv g(x_{i_1}, \dots, x_{i_s}), \{x_{i_1}, \dots, x_{i_s}, y\} \subseteq X, \quad (22)$$

к КНФ при помощи преобразований Цейтина. При этом переменные из \tilde{X} не могут фигурировать в левых частях соотношений вида (22).

Зависимости вида (22) можно представить специальным направленным ациклическим графом $G(C)$, часть вершин которого не имеют родителей. Все такие вершины соответствуют входным переменным из \tilde{X} . Любая другая вершина имеет как минимум одного родителя и соответствует переменной в левой части соотношения вида (22). Конкретно соотношению (22) в графе $G(C)$ отвечает вершина $v(y)$ с приписанной ей переменной y , родители которой — это вершины $v(x_{i_j}), j \in \{1, \dots, s\}$.

Пусть $\alpha \in \{0, 1\}^{|\tilde{X}|}$ — произвольный набор значений переменных из \tilde{X} . Если теперь, используя набор α вычислить значения всех переменных типа y из (22) в графе $G(C)$, то будут получены значения всех переменных из X .

Обозначим такой набор значений переменных из X через $\gamma(\alpha)$ и назовем его набором, индуцированным α .

Рассмотрим теперь КНФ C , кодирующую задачу существования квазиортогональной системы латинских квадратов, граф $G(C)$ и соответствующий данному графу набор соотношений вида (22). Зададим следующую псевдодобулевую функцию:

$$\Phi_C : \{0, 1\}^{|\tilde{X}|} \longrightarrow \{0, 1, \dots, m\}, \quad (23)$$

где m — число дизъюнктов в C : для произвольного $\alpha \in \{0, 1\}^{|\tilde{X}|}$ значение (23) равно числу дизъюнктов в C , выполненных вектором $\gamma(\alpha)$. Несложно понять, что если $\Phi_C(\alpha) = m$, то α определяет квазиортогональную систему латинских квадратов, проблему существования которой кодирует КНФ C . Если же $\max_{\alpha \in \{0, 1\}^{|\tilde{X}|}} \Phi_C < m$, то система с требуемыми свойствами не существует. Таким образом, задача поиска квазиортогональной системы с заданным индексом ортогональности сведена к проблеме максимизации функции (23) на булевом гиперкубе $\{0, 1\}^l$, $l = |\tilde{X}|$.

Предположим, что при построении C использовалось «one-hot» кодирование натуральных чисел. В этих предположениях задача максимизации (23) обладает одной интересной особенностью: для многих векторов из $\{0, 1\}^l$ вывод о том, что значение функции (23) на них $< m$ делается тривиально: действительно, если, например, в векторе $\alpha \in \{0, 1\}^l$ в координатах, кодирующих некоторое число $i \in \{1, \dots, n\}$ число единиц превосходит 1, то понятно, что все условия, определяемые графом $G(C)$ заведомо не выполняются и, как следствие, $\Phi_C(\alpha) < 0$.

Исходя из сказанного, возникает следующая идея. Для «one-hot» кодировок определить локальный поиск не на всем $\{0, 1\}^l$, где l — общее число переменных, кодирующих квадраты, а на специальном образом построенном подмножестве $\{0, 1\}^l$. Поясним сказанное более конкретными рассуждениями. Предположим, что рассматривается задача поиска системы из k латинских квадратов порядка n , удовлетворяющей тем или иным ограничениям, например, образующих квазиортогональную систему некоторого индекса. Каждая ячейка квадрата кодируется n булевыми переменными. Всего, таким образом, мы имеем $l = k \cdot n^2$ булевых переменных, от которых функционально зависят все остальные переменные в кодировке. Обозначим это множество переменных через X^{in} . Заметим, что мы можем разбить X^{in} на

$L = k \cdot n^2$ подмножеств переменных: каждое такое подмножество кодирует заполнение конкретной ячейки рассматриваемых квадратов. Про любое такое подмножество (обозначим его через X') мы знаем, что в наборе, выполняющем нашу итоговую КНФ, набор значений переменных из X' — это вектор веса Хэмминга 1 длины n . Таким образом, мы можем связать с каждым множеством переменных вида X' новую мета-переменную y' , которая может принимать n возможных значений: мы можем обозначить их натуральными числами от 1 до n . Также мы знаем, что за произвольным значением $q \in \{1, 2, \dots, n\}$ такой переменной «стоит» булев вектор длины n с единицей в q -й координате, притом что все остальные координаты равны 0. Пусть теперь X^1, \dots, X^L — множества булевых переменных, кодирующих ячейки квадратов и y^1, \dots, y^L — соответствующие им, в указанном смысле, мета-переменные с доменами $D^j = \{1, 2, \dots, n\}, j \in \{1, \dots, L\}$. Рассмотрим следующее множество:

$$D = D^1 \times \dots \times D^L \quad (24)$$

Несложно понять, что (24) — метрическое пространство, на котором может быть задана метрика Хэмминга, и, соответственно, определена система окрестностей (neighborhood structure), концептуально аналогичная системе окрестностей Хэмминга на $\{0, 1\}^n$. Более конкретно, мы можем рассмотреть на D систему, образованную окрестностями с радиусом Хэмминга $r = 1$. Отметим, что каждая такая окрестность произвольной точки $\alpha \in D$ будет содержать $L \cdot (n - 1) + 1$ точек (для фиксированной α и координаты с номером $j \in \{1, \dots, L\}$ существует $n - 1$ точек, отличающихся от α в координате с номером j , все эти точки находятся на расстоянии Хэмминга 1 от α в D). Тогда с задачей поиска системы латинских квадратов с фиксированным индексом ортогональности можно связать задачу максимизации следующей непсевдобулевой функции

$$\Psi_C : D \longrightarrow \{0, 1, \dots, m\}, \quad (25)$$

заданной на множестве D . Функция (25) задается по аналогии с (23): произвольному вектору $\beta \in D : \beta = (\beta_1, \dots, \beta_L), \beta_j \in \{1, \dots, n\}$ однозначно сопоставляется булев вектор $\alpha \in \{0, 1\}^l$ — набор значений переменных из множества $X = \cup_{j=1}^L X^j$. При этом набор значений переменных из X^j — вектор веса Хэмминга 1 длины n с единицей в позиции с номером β_j . По вектору α и

графу $G(C)$ вычисляется функция Φ_C . Полученное значение — это значение функции (25).

Помимо простого алгоритма Hill Climbing, использующего окрестности радиуса 1 в D был реализован также один вариант поиска с переменными окрестностями, основанный на т.н. принципе склеивания переменных (Merging Variables Principle, MVP [26]). В MVP рассматриваются расширенные окрестности, которые порождаются группами базовых переменных — это могут быть либо булевы переменные, либо переменные с доменами $D^j, j \in \{1, \dots, L\}$. Более конкретно, например, результатом склеивания двух переменных y^1, y^2 является переменная z^{12} , домен которой — это $D^1 \times D^2$. Как показано в [26] для построенных таким способом склеенных переменных можно определить декартово произведение их доменов, на котором можно задать структуру окрестностей Хэмминга радиуса 1. Дальнейший поиск — это Hill Climbing в построенном таким способом пространстве с периодическим обновлением множества склеенных переменных.

Выводы по главе 2

В данной главе были описаны основные методы и техники решения ортогональных и квазиортогональных систем латинских квадратов, разработанных в настоящей работе. Так в разделе 2.1 описана схема построения квазиортогональных систем с итеративно увеличивающимся индексом, использующая инкрементальную технику. В разделе 2.2 представлен способ кодирования предиката ограничения симметрии из [21] в выражение в КНФ представления. В разделе 2.3 был представлен комбинированный способ решения задачи поиска квазиортогональных систем с использованием локального поиска по мета-переменным, соответствующим ячейкам квадратов.

ГЛАВА 3. ЭКСПЕРИМЕНТАЛЬНЫЕ РЕЗУЛЬТАТЫ

В данной главе указаны детали реализации описанных во второй главе методов. Также описаны проведенные вычислительные эксперименты, и представлены полученные результаты. Произведено сравнение с существующими работами и представлена квазиортогональная тройка латинских квадратов порядка 10 с индексом ортогональности 82.

3.1. Результаты по технике ограничения симметрии

Для кодирования задачи поиска ортогональных и квазиортогональных систем латинских квадратов была реализована программа, которая позволяет выбирать порядок системы n , ее мощность k и индекс ортогональности r . Для ортогональных систем ($r = n^2$) используется формула (9), которая на практике показывает лучшие результаты по сравнению с (15).

Итак, пусть мы закодировали задачу поиска квазиортогональной системы латинских квадратов LS_1, \dots, LS_k , и получили булеву формулу в КНФ C .

Чтобы построить предикат ограничения симметрии, фиксируем первый столбец LS_1 и первую строку LS_1, \dots, LS_k лексикографически отсортированной последовательностью. Так, конъюнкция, фиксирующая какую-либо ячейку квадрата числом m будет иметь вид

$$\neg x_1 \wedge \dots \wedge x_m \wedge \dots \wedge x_n,$$

где $\{x_1, \dots, x_n\}$ множество переменных, соответствующее этой ячейке в «one-hot» кодировке.

Для кодирования того, что первые две строки LS_1 образуют перестановку с заданным циклическим представлением, описанным в Теореме 1, воспользуемся связью перестановок такого вида с разбиением чисел на слагаемые, отличные от 1. Так, мы можем перебрать все такие разбиения в силу их небольшого числа для рассматриваемого класса задач с $n \leq 10$, и получить по ним все возможные перестановки второй строки квадрата LS_1 . Такие перестановки кодируем аналогично (21). В итоге получим предикат, ограничивающий симметрию системы, который соединим конъюнкцией с C и получим C^* .

Заметим, что переменные в C^* , соответствующие ячейкам в первом столбце LS_1 , имеют фиксированное значение, а также ограничивают часть

других переменных, кодирующих LS_1 . Так, например, в ячейке LS_1 на пересечении строки i и столбца $j : 2 \leq j \leq k$ не может стоять значение равное i . Аналогично, в первых строках LS_1, \dots, LS_k также фиксируется значение соответствующих булевых переменных и ограничивает значения в других строках. Таким образом, мы можем сократить число переменных, непосредственно кодирующих ячейки квадратов в выражении C^* , произведя операцию Unit Propagation [6], с $n^3 \cdot k$ до $(n-2) \cdot (n-1)^2 + (n-1) + (n \cdot (n-1)^2) \cdot (k-1)$. Например, для троек порядка 10 число переменных уменьшится с 3000 до 2277.

В вычислительных экспериментах, представленных в Таблице 1, исследовались решения задачи поиска ортогональных систем порядка n от 7 до 10 и мощности k от 2 до 3. Вычисления проводились с помощью SAT-решателя `rainless-mcomsps` [27], который победил в соревнованиях SAT competition 2020 в номинации параллельных SAT-решателей. Для каждой системы было произведено по 10 запусков с ограничением по времени в 1 час. В таблице представлены следующие результаты: «one-hot» кодировки без предиката ограничения симметрии; с предикатом ограничения симметрии, использующий теорему из [21]; с предикатом ограничения симметрии, описанным в работе [20]; результаты, полученные в работе [20] при аналогичных экспериментах.

Вычисления производились на платформе AMD Opteron 6378 @ 2.4 ГГц с использованием 32 логических ядер на 32 потоках и 50 Гб оперативной памяти.

Эксперименты из работы [20] проводились на платформе Intel Core i9-9900K @ 3.6 ГГц с 32 Гб оперативной памяти с использованием ILP-решателя Gurobi [28]. При этом, для перестановки, образованной первыми столбцами первого и второго квадрата в системе, фиксировалось конкретное представление в циклической записи, и, соответственно, фиксировались значения этих столбцов. Для каждого варианта циклической записи такой перестановки проводилась отдельная серия экспериментов. В Таблице 1 представлены лучшие результаты из всех таких серий. Результаты из работы по тройкам латинских квадратов несравнимы, потому что в работе [20] использовался совершенно другой метод их поиска: с помощью CLP-решателя OR TOOLS итеративно перебирался третий квадрат в лексикографическом порядке. Так, на каждой итерации производился поиск ортогональной тройки с фиксированным третьим квадратом, и, если система не была найдена, то фиксировался следую-

щий квадрат в порядке перебора. Такой подход не сравним с предложенным в данной работе из-за строгой привязке к существованию ортогональных систем к лексикографическому порядку одного из квадратов в системе.

Таблица 1 – Сравнение времени решения кодировок задачи поиска ортогональной системы порядка n и мощности k

(n, k)	Время решения (с)			
	Без ограничения симметрии	Ограничение симметрии из [21]	Ограничение симметрии из [20]	Результаты из [20]
(7, 2)	0,277	0,159	0,182	0
(8, 2)	0,612	0,315	3,115	4
(9, 2)	49,537	15,882	152,689	299
(10, 2)	505,981	91,632	более 3600	14256
(7, 3)	17,804	0,551	6,054	—
(8, 3)	1816,84	70,749	более 3600	—
(9, 3)	более 3600	более 3600	более 3600	—
(10, 3)	более 3600	более 3600	более 3600	—

Из полученных результатов можем сделать вывод, что кодировка с описанным в разделе 2.2 предикатом ограничения симметрии решается за существенно меньшее время относительно других сравниваемых кодировок. Сравнение с результатом работы [20] показывает, что время на решение данной задачи с использованием SAT, а также подход решения полной кодировки полной ортогональной системы на порядок эффективнее, чем сведение той же задачи к проблемам ILP и CLP.

3.2. Результаты по итеративной технике

Для реализации метода итеративного увеличения индекса, описанного в разделе 2.1 использовалась библиотека `cryptominisat` [23] для C++, с помощью которой можно использовать функционал этого SAT-решателя и инкрементально добавлять дизъюнкты в решаемую КНФ формулу.

Для исследования производительности данной техники проводились эксперименты по поиску квазиортогональных систем порядка 10 мощности 3 и индекса ортогональности r . Для каждого индекса было сделано по 5 запусков с ограничением по времени в 5 часов. Решения сравнивались со стандартным подходом поиска таких систем с помощью SAT-решателя `painless-mcomsps`. Результаты экспериментов представлены в Таблице 2. В таблице указано мини-

мальное, среднее и максимальное время решения системы в серии экспериментов для индексов ортогональности 75, 77 и 79.

Таблица 2 – Сравнение инкрементального и стандартного подхода к решению задачи поиска квазиортогональной тройки латинских квадратов порядка 10 с различным индексом ортогональности r .

Индекс ортогональности r	Время решения (с)					
	Инкрементальный подход			Стандартный подход		
	Мин.	Среднее	Макс.	Мин.	Среднее	Макс.
75	55,95	117,20	213,16	49,44	167,86	275,53
77	672,76	829,59	1089,11	12,82	2182,97	3595,01
79	11490,8	> 18000	> 18000	11016,74	> 18000	> 18000

По результатам из таблицы можем сделать вывод, что стандартный подход в лучшем случае может быть эффективнее, чем инкрементальный. Так минимальное время, затраченное решателем `rainless-mscomsprs` во всех проведенных сериях экспериментов меньше, чем у инкрементального подхода с решателем `cryptominsat5`. Однако, среднее и максимальное время в проведенных сериях экспериментов у инкрементального подхода лучше.

Также из достоинств инкрементального подхода можно отметить то, что в процессе итеративного решения системы мы можем сохранять значения для систем меньшего индекса ортогональности, и при достижении условия ограничения по времени использовать его как ответ. Так, задача поиска квазиортогональной системы с конкретным индексом ортогональности сводится к задаче поиска системы с наибольшим индексом за определенное время.

3.3. Применение техники склеивания переменных

Алгоритм локального поиска по окрестностям вида, описанного в разделе 2.3 для систем латинских квадратов вместе с техникой склеивания переменных был реализован на языке C++.

Для повышения эффективности перебора была также реализована следующая эвристика. На очередной итерации перебора окрестностей вектора мета-переменных $\beta = \{\beta_1, \dots, \beta_L\}$, мы рассматриваем вектора вида $\beta^* = \{\beta_1^*, \dots, \beta_L^*\}$, которые отличаются от β в компонентах i_1, \dots, i_n . То есть

$\beta_i \neq \beta_i^* \iff i \in \{i_1, \dots, i_n\}$. Мы можем отбросить такие вектора β^* , в которых вектор булевых переменных α' , порожденный мета-переменными $\beta_{i_1}, \dots, \beta_{i_n}$ не находятся в невыполнимых дизъюнктах с C .

Метод склеивания переменных применялся тогда, когда алгоритм приходил в локальный максимум на окрестности. Тогда, число мета-переменных в одной «склейке» увеличивалось на 1 и следовательно увеличивался размер окрестности. Когда размер окрестности доходил до слишком больших значений для полного перебора, то генерировалось конечное число точек из окрестности, на которых потом и запускался перебор.

Условием остановки алгоритма является достижения некоторого порога числа мета-переменных в одной «склейке».

Перебор окрестностей мета-переменных можно вычислять в несколько потоков. Для этого каждому из потоков выделялось равное количество сгенерированных точек для перебора. Это позволяет значительно ускорить алгоритм на многопроцессорных системах, а для итераций с большим числом «склеенных» мета-переменных алгоритм можно адаптировать для вычисления на кластере из нескольких вычислительных машин.

В результате работы алгоритма для кодировок поиска ортогональных систем со случайной начальной точкой выдавались вектора булевых значений, соответствующие квазиортогональным системам из «псевдолатинских» квадратов, в которых некоторые элементы нарушали свойство «латинскости» квадрата. Например, такими элементами могут быть ячейки с одинаковым значением, находящиеся в одной строке определенного квадрата. Из наблюдений за улучшением системы в ходе работы алгоритма было замечено, что на каждой итерации оптимизировалось как число элементов, нарушающих свойство «латинскости» квадратов, так и индекс ортогональности системы. Пример системы порядка 10 и мощности 3, полученной в результате локального поиска из случайной точки указан на Рисунке А.2 в приложении А.

3.4. Результаты для комбинированного подхода

Комбинированный подход использования SAT-решателя наряду с реализованным в настоящей работе алгоритмом локального поиска применялся на задаче поиска квазиортогональной тройки порядка 10 с максимальным индексом ортогональности.

Можно предложить несколько вариантов комбинации данных подходов:

Первый — запуск алгоритма локального поиска с произвольной точки. Таким образом, мы получаем систему из псевдолатинских квадратов, в которых некоторые элементы нарушают свойство «латинскости» квадрата, т.е. элементы, которые находятся в одной строке или одном столбце одного квадрата и имеют одинаковые значения. Такая система на практике проведенных экспериментов оказывается с индексом ортогональности (который также можем определить на псевдолатинских квадратах как вес Хэмминга вектора маркировок пары квадратов), приближенным к 90. Далее мы можем построить булево выражение, зафиксировав некоторые элементы. В настоящей работе фиксировались элементы в ячейках, которые не находятся в одной строке или одном столбце с элементами, нарушающими свойство «латинскости». В итоге, мы получим булево выражение с меньшим числом пропозициональных переменных и с потенциально большим возможным индексом ортогональности системы. На такой кодировке мы можем запустить поиск квазиортогональной системы с помощью инкрементального подхода, и существенно быстрее находить примеры систем с большим индексом ортогональности.

Второй — использование примера квазиортогональной тройки, найденной SAT-решателем с помощью обычного или инкрементального подхода. Далее, мы можем использовать это решение как начальную точку в алгоритме локального поиска на задаче построения ортогональной системы. При использовании такого подхода, локальный поиск с техникой склеивания переменных может прийти к системе с большим минимальным индексом ортогональности.

Описанные подходы можно повторять несколько раз, чередуя. В результате нескольких попыток проведения таких серий запусков в настоящей работе была найдена система из трех латинских квадратов с индексом ортогональности 82, представленная в приложении А на Рисунке А.1. Вычисления проводились на платформе Intel Core i7 8665U @ 1.9 ГГц с использованием 4 физических ядер и 16 Гб оперативной памяти. Работа инкрементального SAT-решателя длилась 20 часов. Квазиортогональная система того же индекса была найдена работе [22], но в вычислениях использовалась платформа Intel Xeon 2695 @ 2.4 ГГц с 36 физическими ядрами, и поиск такой системы занял 72 часа. Таким образом, удалось повторить существующие результаты, но на значительно менее производительной платформе за меньшее количество времени.

Выводы по главе 3

В данной главе приведены реализации разработанных методов и техник, и описаны условия проведенных вычислительных эксперименты. Представленные результаты этих экспериментов показывают, что разработанные техники существенно повышают эффективность алгоритмов решения SAT на приведенном классе задач, а также превосходят по эффективности сравниваемые работы. Комбинация всех техник позволила получить квазиортогональную систему из трех латинских квадратов порядка 10 с индексом ортогональности 82.

ЗАКЛЮЧЕНИЕ

В настоящей работе были рассмотрены способы сведения задач, связанных с латинскими квадратами, к проблеме SAT, предложенные в [15, 22]. Также были исследованы существующие методы решения задач из данного класса. Так, в работе [17] использовался алгоритм эффективного перебора латинских квадратов, основанный на их математических свойствах. В результате данной работы была найдена квазиортогональная тройка порядка 10 с индексом ортогональности 92. Но в этой работе не использовалась задача булевой выполнимости. В работах [19, 20] было предложено сведение задачи построения ортогональных систем к проблеме ILP и CLP. В работе [15, 22] была представлена задача поиска квазиортогональных систем и предложен способ кодирования данной задачи в SAT.

Также в рамках исследовательской работы были разработаны и реализованы следующие техники:

- 1) Способ кодирования условия нарушения симметрии в SAT из [21]. В результате вычислительных экспериментов было произведено сравнение эффективности современных многопоточных SAT-решателей с другим предикатом нарушения симметрии из [20], а также с вычислительными результатами экспериментов из [20]. Было установлено, что реализованный предикат существенно выигрывает по времени работы на задачах поиска ортогональных пар относительно сравниваемой работы и другого предиката.
- 2) Метод, позволяющий итеративно увеличивать индекса ортогональности квазиортогональных систем с применением инкрементальной техники. Вычислительные эксперименты показали, что в среднем и худшем случае время работы современных SAT-решателей улучшилось.
- 3) Алгоритм локального поиска, специализированного на задачах поиска ортогональных и квазиортогональных систем латинских квадратов. Комбинация использования данного алгоритма с CDCL-решателем и применением разработанных техник позволила найти квазиортогональную тройку порядка 10 с индексом ортогональности 82.

Подводя итог настоящей работы, можно утверждать, что разработанные техники существенно повышают эффективность алгоритмов решения SAT на

задачах построения ортогональных и квазиортогональных систем латинских квадратов.

СПИСОК ИСПОЛЬЗОВАННЫХ ИСТОЧНИКОВ

- 1 *Holl M.* Комбинаторика. — М.: «Мир», 1970. — С. 261–283.
- 2 *Tarry G.* Le problème de 36 officiers // *Compte Rendu de la Association Française pour l'Avancement des Sciences.* — 1900. — Т. 29, n° 1. — P. 122-123.
- 3 *Bose R., Shrikhande S.* On the falsity of Euler's conjecture about the nonexistence of two orthogonal Latin squares of order $4t + 2$ // *Proceedings of the National Academy of Sciences of the United States of America.* — 1959. — P. 734–737.
- 4 *Bose R., Shrikhande S., Parker E.* Further results on the construction of mutually orthogonal Latin squares and the falsity of Euler's conjecture // *Canadian Journal of Mathematics.* — 1960. — Vol. 12. — P. 189–203.
- 5 *Moore E.* Tactical Memoranda I–III // *American Journal of Mathematics.* — 1896. — Vol. 18, no. 3/4. — P. 264–303.
- 6 *Marques-Silva J. P., Lynce I., Malik S.* Conflict-driven clause learning solvers // *Handbook of Satisfiability* / ed. by A. Biere [et al.]. — IOS Press, 2009. — P. 131–153.
- 7 *Цейтин Г. С.* О сложности вывода в исчислении высказываний // *Записки научных семинаров ЛОМИ.* — 1968. — Т. 8. — С. 234–259.
- 8 *Boros E., Hammer P.* Pseudo-Boolean optimization // *Discrete Appl. Math.* — 2002. — Vol. 123. — P. 155–225.
- 9 *Burke E., Kendall G.* *Search Methodologies. Second Edition.* — Springer, 2014.
- 10 *Russell S., Norvig P.* *Artificial Intelligence: A modern approach. Third edition.* — Pearson, 2010. — P. 120–161.
- 11 *Kirkpatrick S., Gelatt C., Vecchi M.* Optimization by simulated annealing // *Science.* — 1983. — Vol. 220, no. 4598. — P. 671–680.
- 12 *Glover F., Laguna M.* Tabu Search // *Handbook of Combinatorial Optimization* / ed. by D. Z. Du, P. P. M. — Kluwer Academic Publishers, Norwell, Ma, USA, 1997. — P. 2093–2229.
- 13 *Mladenovic N., Hansen P.* Variable neighborhood search // *Computers and Operations Research.* — 1997. — Vol. 24, no. 11. — P. 1097–1100.

- 14 *Zhang H.* Combinatorial Design by SAT Solvers // Handbook of Satisfiability / ed. by A. Biere [et al.]. — IOS Press, 2009. — Chap. 17. P. 553–568.
- 15 *Белей Е. Г., Семенов А. А.* О способах пропозиционального кодирования различимости объектов в конечных множествах // Известия Иркутского государственного университета, Серия «Математика». — 2019. — С. 3–20.
- 16 *Sinz C.* Towards an Optimal CNF Encoding of Boolean Cardinality Constraints // Lecture notes in computer science. — 2007. — Vol. 3709. — P. 827–831.
- 17 *Egan J., Wanless I. M.* Enumeration of MOLES of small order // Mathematics of Computation. — 2016. — P. 799–824.
- 18 *McKay B., Meynert A., Myrvold W.* Small Latin squares, quasigroups, and loops // Journal of Combinatorial Designs. — 2007. — Т. 15. — С. 98–119.
- 19 *Appa G., Magos D., Mourtos I.* Searching for mutually orthogonal latin squares via integer and constraint programming // European journal of operational research 173(2). — 2006. — P. 519–530.
- 20 Integer and Constraint Programming Revisited for Mutually Orthogonal Latin Squares [Электронный ресурс] / N. Rubin [et al.]. — 03/2021. — URL: <https://arxiv.org/pdf/2103.11018.pdf>.
- 21 *Ma F., Zhang J.* Finding orthogonal latin squares using finite model searching tools // Science China Information Sciences. — 2013. — Vol. 56. — P. 1–9.
- 22 *Белей Е. Г., Семенов А. А.* О вычислительном поиске квазиортогональных систем латинских квадратов, близких к ортогональным системам // International Journal of Open Information Technologies ISSN. — 2018. — № 2. — С. 22–30.
- 23 CryptoMiniSat with CCAr at the SAT Competition 2020 / M. Soos [et al.] // Proceedings of SAT Competition 2020: Solver and Benchmark Descriptions. — 2020. — Vol. B-2020–1. — P. 27–28.
- 24 *Sakallah K.* Symmetry and Satisfiability // Handbook of Satisfiability / ed. by A. Biere [et al.]. — IOS Press, 2009. — P. 289–338.
- 25 The on-line encyclopedia of Integer Numbers. Sequence A002865 [Электронный ресурс]. — 2021. — URL: <https://oeis.org/A002865>.

- 26 *Semenov A.* Merging Variables: One Technique of Search in Pseudo-Boolean Optimization // *Communication in Computer and Information Science*. — 2019. — Vol. 1090. — P. 86–102.
- 27 P-MCOMSPS-STR: a Painless-based Portfolio of MapleCOMSPS with Clause Strengthening / V. Vallade [et al.] // *Proceedings of SAT Competition 2020: Solver and Benchmark Descriptions*. — 2020. — Vol. B-2020–1. — P. 56–57.
- 28 *Gurobi Optimization, LLC.* Gurobi Optimizer Reference Manual [Электронный ресурс]. — 2021. — URL: <http://www.gurobi.com>.
- 29 *Handbook of Satisfiability* / ed. by A. Biere [et al.]. — IOS Press, 2009. — P. 980.

**ПРИЛОЖЕНИЕ А. КВАЗИОРТОГОНАЛЬНЫЕ СИСТЕМЫ,
ПОЛУЧЕННЫЕ В ЭКСПЕРИМЕНТАХ**

1 7 6 4 3 10 2 9 8 5	1 5 3 10 4 8 6 7 9 2	1 9 4 5 7 2 8 6 10 3
7 2 1 10 5 6 9 4 3 8	2 7 9 8 1 10 4 3 6 5	2 4 8 3 9 7 10 5 6 1
9 3 2 8 4 1 10 6 5 7	3 6 8 4 2 7 1 5 10 9	3 1 7 2 6 9 5 8 4 10
4 5 10 9 8 3 1 7 6 2	4 8 2 7 3 9 5 6 1 10	4 10 1 7 8 5 2 3 9 6
6 1 4 5 7 2 3 8 9 10	6 9 10 5 7 4 3 2 8 1	5 7 9 6 10 1 3 4 2 8
10 6 5 2 9 7 8 3 1 4	9 2 5 3 8 1 7 10 4 6	6 8 5 10 1 4 9 2 3 7
5 9 7 1 6 8 4 10 2 3	7 3 4 1 5 6 10 9 2 8	7 2 6 8 3 10 1 9 5 4
8 10 9 3 1 4 5 2 7 6	8 4 6 2 10 5 9 1 3 7	8 6 10 9 5 3 4 7 1 2
2 8 3 6 10 5 7 1 4 9	5 10 1 9 6 3 2 8 7 4	9 3 2 1 4 6 7 10 8 5
3 4 8 7 2 9 6 5 10 1	10 1 7 6 9 2 8 4 5 3	10 5 3 4 2 8 6 1 7 9

Рисунок А.1 – Квазиортогональная тройка латинских квадратов порядка 10. Индексы ортогональности пар квадратов 1–2, 1–3, 2–3 соответственно 82, 83, 83

1 2 3 4 5 6 7 8 9 10	1 4 9 8 10 7 6 2 3 5	1 3 2 5 10 9 8 7 4 6
2 1 4 3 6 5 8 7 10 9	2 8 6 10 9 3 1 5 4 7	2 3 6 9 7 4 10 1 5 8
3 6 8 1 9 4 10 5 7 2	3 10 4 7 1 5 2 6 9 8	3 8 5 2 6 7 1 9 10 4
4 5 9 8 7 2 1 10 6 3	4 1 5 9 8 7 10 3 2 6	4 7 10 6 9 1 9 2 3 5
5 7 10 6 2 1 4 9 3 6	5 7 8 2 6 9 3 10 1 4	5 4 7 10 1 6 3 6 8 9
6 8 7 10 4 9 3 2 8 1	6 3 10 1 5 8 4 9 7 2	6 9 3 4 8 2 7 5 1 10
7 4 2 9 1 10 6 4 5 8	7 9 1 6 3 2 5 4 8 10	7 1 9 3 5 8 2 10 6 4
8 10 1 5 3 7 9 6 2 4	8 6 7 4 2 10 9 1 5 3	8 2 10 1 6 5 4 3 9 7
9 3 5 2 10 8 6 1 4 7	9 5 2 3 7 6 8 4 10 1	9 6 4 7 3 10 5 8 2 1
10 9 6 7 8 4 2 3 1 5	10 2 3 5 4 1 7 8 6 9	10 5 1 8 2 9 6 4 7 3

Рисунок А.2 – Квазиортогональная тройка «псевдолатинских» квадратов порядка 10, полученная в результате локального поиска. Жирным шрифтом отмечены элементы, нарушающие свойство «латинскости» квадратов. Индексы ортогональности пар квадратов 1–2, 1–3, 2–3 соответственно 88, 86, 84