Федеральное государственное бюджетное учреждение науки Вычислительный центр им. А.А. Дородницына Российской академии наук

На правах рукописи

ПОСЫПКИН Михаил Анатольевич

МЕТОДЫ ГЛОБАЛЬНОЙ И МНОГОКРИТЕРИАЛЬНОЙ ОПТИМИЗАЦИИ НА БАЗЕ КОНЦЕПЦИЙ ВЕТВЕЙ И ГРАНИЦ И НЕРАВНОМЕРНЫХ ПОКРЫТИЙ

01.01.09 Дискретная математика и математическая кибернетика

ДИССЕРТАЦИЯ

на соискание ученой степени

доктора физико-математических наук

Научный консультант

академик РАН

Ю.Г. Евтушенко

Москва, 2014

Оглавление

Введение

Актуальность темы исследования

Задачи глобальной оптимизации часто возникают в различных областях современной науки и техники. В биологии и химии актуальны проблемы поиска конфигураций различных химических соединений, соответствующих минимуму энергии взаимодействия [1]. В частности, такие задачи возникают при разработке новых лекарственных препаратов [2]. В микроэлектронике необходимо оптимизировать размещение компонентов больших интегральных микросхем [3]. Оптимизация часто применяется в промышленном дизайне для улучшения параметров различных конструкций и изделий [4, 5]. Также к оптимизационным относятся различные логистические задачи, управление транспортными потоками [6–8], размещение предприятий [9], планирование производства [10]. Большое практическое значение имеют задачи дискретного программирования, связанные с поиском оптимального размещения различных объектов (телекоммуникационных вышек, заправочных станций и т.п.). Много внимания последнее время уделяется задачам теории расписаний [11, 12]. Классическим стало применение методов оптимизации для решения обратных задач идентификации параметров моделей по данным эксперимента [13]. Экстремальные задачи нередко возникают при решении задач распознавания [14]. Есть много и других областей, в которых оптимизация находит свое применение.

Можно с уверенностью утверждать, что задачи прикладной оптимизации являются чрезвычайно важными для развития различных отраслей науки и тех-

нологий. Поэтому разработка эффективных методов решения оптимизационных задач и их реализация в виде прикладных программных комплексов представляется актуальной научной и практической проблемой. Нередко при решении прикладных задач ограничиваются поиском локальных минимумов, т.е. точек, являющихся минимумом в некоторой своей окрестности. Теория таких методов хорошо развита как для непрерывных оптимизационных задач [15–18], так и для дискретных задач [19–21].

Интерес к решению задач глобальной оптимизации возник достаточно давно, а активное развитие эти методы получили во второй половине XX века. Методы решения задач глобальной оптимизации можно условно разделить на два класса: *эвристические* и *детерминированные*. Эвристические методы основаны на правдоподобных, но не всегда строго обоснованных, предположениях о природе решаемой задачи и позволяют найти некоторое допустимое решение, не давая при этом каких-либо гарантированных оценок на величину отклонения найденного приближенного решения от оптимума. Эвристические методы нередко применяются для решения задач оптимизации "черного ящика", когда аналитическое выражение для целевой функции неизвестно. Большой вклад в создание, развитие и исследование эвристических алгоритмов был внесен представителями российской и зарубежных школ оптимизации Ю.И. Журавлевым [22], К.В. Рудаковым [23], Ю.Ю. Финкельштейном [24], И.Х. Сигалом [25], А.А. Корбутом [26], Ю.А. Кочетовым [19], Б.Т. Поляком [27], Плясуновым А.В. [20], А.П. Карпенко [28], Курейчиком В.В. [29], F. Glover [30], G. Kochenberger [31] , C. Cotta [32], R. Marti [33] и многими другими. Для таких алгоритмов иногда удается получить вероятностные оценки точности получаемых решения [34].

Основное внимание в диссертационной работе уделено детерминированным методам глобальной оптимизации. В отличие от эвристических, такие методы позволяют не только найти решение, но и дают оценку отклонения от оп-

тимума найденного значения целевой функции. Различные детерминированные методы решения дискретных задач создавались и развивались И.Х. Сигалом [35], Ю.Ю. Финкельштейном [36], В.П. Черениным [37], В.Р. Хачатуровым [38], А.А. Лазаревым [39], А. А. Колоколовым [40], G. Danzig [41], E. Balas [42], D. Pisinger [43], S. Martello [44], P. Toth [45], и др. Детерминированные методы решения непрерывных задач глобальной оптимизации были разработаны Ю.Г. Евтушенко [46–48], С.А. Пиявским [49], Р.Г. Стронгиным [50], В.П. Гергелем [51], Я.Д. Сергеевым [52], А.С. Стрекаловским [53], Е.С. Левитиным [54], В.П. Булатовым [55], О. В. Хамисовым [56], E. Hansen [57], R. Kearfott [58], J. Pinter [59], H. Tuy [60] и другими исследователями. Данная диссертационная работа во-многом посвящена развитию метода неравномерных покрытий [46], который был предложен Ю.Г. Евтушенко в 1971 году для решения задач оптимизации с простыми (параллелепипедными) ограничениями и в дальнейшем был обобщен на более широкий класс задач, включая задачи математического программирования и многокритериальной оптимизации.

Большое внимание в современных исследованиях уделяется многокритериальной оптимизации (МКО). Значительный вклад в развитие теории и методов решения таких задач внесли А.В. Лотов [61, 62], В.В. Подиновский [63], В.А. Бушенков [62, 64], В.Д. Ногин [65], Г.К. Каменев [61], И.М. Соболь [66], Р.Б. Статников [67, 68], K. Deb [69], M. Ehrgott [70], E. Zitzler [71], H.P. Benson [72], A. Pascoletti [73], P. Serafini [74], K. Miettinen [75] и многие другие. Детерминированные методы построение приближенных решений многокритериальных задач были предложены Ю.Г. Евтушенко и М.А. Потаповым [76]. Дискретные многокритериальные задачи рассматривались в работах И.Х. Сигала [77], И.И. Меламеда [78], Д.И. Когана [79].

Очень важным представляется вопрос об оценках вычислительной сложности рассматриваемых алгоритмов. Определение класса сложности и получение

оценка на число шагов. Подобные вопросы сложности в работах А.В. Кельманова и А.В. Пяткина [80, 81], М.Ю. Хачая [82, 83], Ю.А. Кочетова, А.В. Плясунова [84, 85], Л.А. Заозерской, А.А. Колоколова, А.В. Еремеева [86, 87] и многих других.

Помимо теоретических основ разрабатывались также программные комплексы для решения задач оптимизации, объединяющие различные методы решения задач глобальной оптимизации. Следует отметить такие разработки, как программный комплекс ДИСО "Диалоговая Система Оптимизации" (ВЦ РАН), пакеты BARON, BONMIN, KNITRO, CBC и другие. Объединение различных методов оптимизации в один программный комплекс является вполне оправданным подходом, так как многие методы решения оптимизационных задач взаимосвязаны. Например, локальные методы поиска экстремума используются для нахождения рекордов в методах типа ветвей и границ. Методы линейного программирования применяются для нахождения оценок при организации ветвлений.

Несмотря на существенный прогресс, достигнутый в области глобальной оптимизации, возможностей существующих подходов нередко не достаточно для решения практических задач оптимизации за приемлемое время. Поэтому актуальна задача разработки новых методов глобальной оптимизации и их эффективной реализации на современных вычислительных платформах.

Цель работы. Основной целью диссертационной работы является разработка и реализация методов решения задач оптимизации, а также всестороннее теоретическое и экспериментальное исследование их эффективности. Для достижения этой цели были поставлены и решены следующие задачи:

- разработка новых и развитие существующих алгоритмов глобальной и дискретной оптимизации с одним и несколькими критериями, основанных на идеях неравномерных покрытий;

- разработка и реализация детерминированных методов решения задач многокритериальной оптимизации;
- теоретическое и экспериментальное исследование эффективности методов решения задач глобальной оптимизации, получение оценок сложности оптимизационных алгоритмов;
- создание мультиплатформенного расширяемого программного комплекса, предназначенного для решения задач оптимизации на многопроцессорных вычислительных системах.

На защиту выносятся следующие результаты:

1. Получены новые миноранты и нижние оценки для минимизации скалярных функций. Предложены новые миноранты, основанные на оценке спектра матрицы Гессе, разработан метод вычисления таких оценок. Для задач оптимизации с параллелепипедными ограничениями предложены новые миноранты, учитывающие необходимые условия оптимальности.
2. В методе неравномерных покрытий предложены способы сокращения области поиска, основанные на построении дополнения покрывающего множества до параллелепипеда, позволяющие существенно повысить быстродействие метода.
3. Метод неравномерных покрытий перенесен на задачи частично-целочисленного программирования, доказана корректность алгоритма.
4. Для задачи многокритериальной оптимизации с параллелепипедными ограничениями доказаны новые свойства ε-Парето множества, устанавливающие связь между этим множеством, границей Парето и оболочкой Эджворта-Парето.

5. Метод неравномерных покрытий для задач многокритериальной оптимизации обобщен на случай произвольных минорант, доказана сходимость метода.

6. Для задач многокритериальной оптимизации с функциональными ограничениями введено новое понятие приближенного решения — множество ε, δ-Парето и предложен алгоритм для его построения, доказана сходимость метода.

7. На основе идей многокритериальной оптимизации введено понятие эффективной оболочки множества, показано, что эта оболочка содержится в его выпуклой оболочке. Введены понятия ε-эффективной границы и ε-оболочки множества, предложены методы их численного построения. Разработанные понятия и методы применены для задачи построения внешней границы рабочей области многосекционного робота-манипулятора с заданной точностью.

8. Разработан подход к оценке сложности метода неравномерных покрытий, с помощью которого получены верхние оценки сложности различных вариантов этого метода для различных классов задач.

9. Получены оценки сложности метода ветвей и границ для задачи о булевом ранце. Проведено экспериментальное исследование и сопоставление их точности.

10. Исследован вопрос параллельной вычислительной сложности одной из возможных реализаций метода ветвей и границ. Получена общая оценка вычислительной сложности, не зависящая от конкретной задачи. Для задачи о сумме подмножеств получены асимптотические формулы для минимальной параллельной сложности, максимального параллельного ускорения и эффективности.

11. Предложен подход к реализации детерминированных и гибридных методов глобальной оптимизации на многопроцессорных вычислительных комплексах, основанный на разделении вычислительной, коммуникационной и управляющей функциональности в программе. Разработан язык описания управления параллельным процессом поиска глобального экстремума, основанный на формализме конечных автоматов. Этот язык позволяет описывать как управление процессом оптимизации, так и балансировку вычислительной нагрузки между процессорами параллельной системы. Такой подход позволяет независимым образом реализовывать и отлаживать соответствующие компоненты, а также, облегчать процесс расширения программного комплекса за счет повторного использования ранее разработанных компонент.

12. На основе разработанного подхода реализован программный комплекс для решения задач глобальной оптимизации на последовательных, параллельных и распределенных системах. В рамках этого программного комплекса были реализованы различные алгоритмы для решения задач конечномерной оптимизации с одним и несколькими критериями, проведены многочисленные эксперименты на различных вычислительных платформах. В частности, разработана и выполнена эффективная параллельная реализация метода ветвей и границ для задачи о булевом ранце с одним ограничением, превосходящая по скорости работы известные аналоги.

Методы исследований. В работе используются методы глобальной оптимизации, методы дискретной математики и математического анализа, методы параллельных и распределенных вычислений.

Научная новизна. Все основные результаты диссертации являются новыми. Разработаны новые алгоритмы решения задач оптимизации с заданной точностью на основе концепции неравномерных покрытий. В частности, в диссер-

тации даны новые миноранты для скалярных функций, оригинальные способы сокращения области поиска, основанные на построении дополнения покрывающего множества до параллелепипеда. Предложенная техника позволила существенно повысить быстродействие метода неравномерных покрытий.

Метод неравномерных покрытий для задач многокритериальной оптимизации обобщен на случай произвольных минорант и перенесен на задачи с функциональными ограничениями.

Введено новое понятие эффективной оболочки множества, обобщающее понятие оболочки Эджворта-Парето для многокритериальных задач. Показано, что эта оболочка содержится в его выпуклой оболочке. Введены понятия ε-эффективной границы и ε-оболочки множества, предложены методы их численного построения.

Получены новые оценки сложности различных вариантов метода неравномерных покрытий. Также в диссертации предложен новый подход к оценке вычислительной сложности метода ветвей и границ в задаче о сумме подмножеств, с помощью которого получены оценки для конкретных вариантов метода и серий задач.

Предложен оригинальный подход к реализации детерминированных и гибридных методов глобальной оптимизации на многопроцессорных вычислительных комплексах, основанный на разделении вычислительной, коммуникационной и управляющей функциональности в программе. На базе предложенного подхода создан программный комплекс для решения задач оптимизации.

Теоретическая и практическая ценность. Работа носит теоретический и прикладной характер. Полученные результаты могут найти применение в теории глобальной, дискретной и непрерывной оптимизации с одним и несколькими критериями. В частности, техника получения оценок числа шагов методов, изложенная в диссертационной работе, может быть применена для теорети-

ческого исследования сложности детерминированных методов решения задач оптимизации. Подход, предложенный в диссертационной работе, может использоваться при создании оптимизационных пакетов, ориентированных на многопроцессорные системы. Разработанный в результате работы над диссертацией комплекс программ применим при решении различных прикладных задач оптимизации, возникающих в вычислительной химии и биологии, при проектировании летательных аппаратов, машин и механизмов, в экономике и других областях. Разработанные в диссертации методы решения задач многокритериальной оптимизации были применены для практически важной задачи нахождения рабочей области робота-манипулятора.

Апробация работы. Результаты диссертационной работы докладывались на следующих конференциях:

- XII международная конференция "Математические методы распознавания образов", Москва, 2005 г.;
- VII Международная конференция "Дискретные модели в теории управляющих систем", Москва, 2006 г.;
- международная конференция "Тихонов и современная математика: Вычислительная математика и информатика", Москва, 2006 г.;
- вторая международная конференция "Распределенные вычисления и Грид-технологии в науке и образовании", GRID'2006, Дубна, 2006 г.;
- III международная конференция "Параллельные вычисления и задачи управления" PACO'2006 памяти И.В. Прангишвили, Москва, 2006 г.;
- II международная науч.-практ. конференция "Современные информационные технологии и ИТ-образование":, Москва, 2006 г.;
- V московская международная конференция по исследованию операций

ORM2007, Москва, 2007 г.;

- IX международный семинар “Дискретная математика и ее приложения”, Москва, 2007 г.;
- Parallel Computing Technologies 9th International Conference, PaCT 2007, Переславль-Залесский, 2007 г.;
- вторая международная конференция “Системный анализ и информационные технологии” САИТ-2007, Обнинск, 2007 г.;
- XV международная конференция “Проблемы теоретической кибернетики”, Москва, 2008 г.;
- XXI International Symposium on Nuclear Electonics and Computing, Варна, 2007 г.;
- Discrete and Global Optimization, International Conference in Honor of 50-th Anniversary of Glushkov Institute of Cybernetics, Ялта, 2008 г.;
- Научный сервис в сети Интернет’2008: решение больших задач, Абрау-Дюрсо, 2008 г.;
- четвертая Международная конференция “Параллельные вычисления и задачи управления” PACO’2008, Москва, 2008 г.;
- третья международная научно-практическая конференция “Современные информационные технологии и ИТ-образование”, Москва, 2008 г.;
- третья международная конференция “Распределенные вычисления и Грид-технологии в науке и образовании”, GRID’2008, Дубна, 2008 г.;
- XVII международная школа-семинар "Синтез и сложность управляющих систем"им. академика О.Б. Лупанова, Москва, 2008 г.;
- 4th EGEE User Forum, Catania, Sicily, Italy, 2009 г.;

- Параллельные вычислительные технологии (ПаВТ'2009), Нижний Новгород, 2009 г.;
- International Supercomputing Conference, Hamburg, Germany, 2009 г.;
- Optimization and applications (OPTIMA-2009), Petrovac, Montenegro, 2009 г.;
- третья международная конференция "Системный анализ и информационные технологии" САИТ - 2009, Звенигород, 2009 г.;
- четвертая конференция "Современные информационные технологии и ИТ-образование", Москва, 2009 г.;
- третья международная научная конференция "Суперкомпьютерные системы и их применение" (SSA'2010), Минск, Беларусь, 2010 г.;
- 8-я международная конференция "Интеллектуальная обработка информации", Пафос, Кипр, 2010 г.;
- V Международная конференция кПараллельные вычисления и задачи управлениянь PACO-2010, Москва, 2010 г.;
- VI Московская международная конференция по исследованию операций (ORM2010), Москва, 2010 г.;
- международная конференция по прикладной математике и информатике, посвященная 100-летию со дня рождения академика А.А. Дородницына, Москва, 2010 г.;
- 4-я Международная конференция "Распределенные вычисления и грид-технологии в науке и образовании Дубна, 2010 г.;
- XV Байкальская международная школа-семинар "Методы оптимизации и их приложения", Листвянка, 2011 г.;

- Eleventh International Conference on Parallel Computing Technologies (PaCT-2011), Казань, 2011 г.;
- II International conference “Optimization and applications” (OPTIMA-2011), Petrovac, Montenegro, 2011 г.;
- 3rd Int. Conf. on Optimization Methods and Software, Chania, Crete, Greece, 2012 г.;
- шестая международная конференция “Параллельные вычисления и задачи управления PACO’2012”, Москва, 2012 г.;
- международная молодежная конференция-школа “Современные проблемы прикладной математики и информатики”, Дубна, 2012 г.;
- 3rd International conference “Optimization and applications” (OPTIMA-2012), Petrovac, Montenegro, 2012 г.
- 4th International conference “Optimization and applications” (OPTIMA-2013), Petrovac, Montenegro, 2013 г.
- 20th Conference of the International Federation of Operational Research Societies (IFORS), Barcelona, Spain, 2014 г.
- V International Conference on Optimization Methods and Algorithms (OPTIMA-2014), Petrovac, Montenegro, 2014
- Международная молодежная конференция “Современные проблемы прикладной математики и информатики”, Дубна, 2014
- VIII Всероссийская научная конференция с международным участием “Математическое моделирование развивающейся экономики, экологии и технологий”, Москва, 2014

Также результаты диссертационной работы докладывались на научных семинарах ВЦ РАН, ИППИ РАН, ИСА РАН, ИПУ РАН.

Глава 1

Детерминированные методы решения задач глобальной оптимизации с одним критерием

1.1 Постановка задачи и обзор известных результатов

Рассматривается задача отыскания глобального минимума непрерывной функции $f : \mathbb{R}^n \to \mathbb{R}$ на компактном *допустимом множестве* $X \subseteq \mathbb{R}^n$:

$$f_* = glob \min_{x \in X} f(x) = f(x_*), \tag{1.1}$$

где x_* — любая точка, в которой достигается глобальный минимум, равный f_*.

Методы решения задач глобальной оптимизации можно условно разделить на две категории: *детерминированные методы* и *недетерминированные (эвристические) методы*. Эвристические методы основаны на правдоподобных, но не всегда строго обоснованных, предположениях о природе решаемой задачи и позволяют найти некоторое допустимое решение, не давая при этом каких-либо гарантированных оценок на величину отклонения найденного приближенного решения от оптимума. Эвристические методы нередко применяются для решения задач оптимизации "черного ящика", когда аналитическое выражение для целевой функции неизвестно.

Среди эвристических методов следует отметить различные варианты гене-

тических алгоритмов, которые разрабатывались в работах Eiben и Smith [88], Гладкова Л.А., Курейчика В.В., Курейчика В.М. [29], Michalewicz [89]. Более широким классом по отношению к генетическим являются популяционные алгоритмы, в которых набор точек допустимого множества моделируется как совокупность биологических объектов. Такие методы развивались в работах Карпенко А.П. [90–92], J. Kennedy [93], Clerc [94] и других авторов. Различные эвристические алгоритмы локального поиска рассматривались в работе [19].

В диссертационной работе основное внимание уделено детерминированным методам. Детерминированные методы глобальной оптимизации, основанные на интервальном анализе развивались в работах [95, 96]. В этих работах предлагаются различные способы определения диапазона изменения значений функции при заданных интервалах изменений ее параметров. Далее эти оценки используется при организации отсева в методах типа ветвей и границ. Методы, основанные на представлении функции в виде разности двух выпуклых, рассматриваются в работах Х. Туя [97] и Стрекаловского А.С. [53]. Широкий класс методов использует свойство Липшицевости функций. В первых работах по этой теме Евтушенко Ю.Г. [46] и Пиявский С.А. [49] предлагали методы, основанные на априорных оценках константы Липшица. Однако, такую оценку иногда достаточно затруднительно получить. В работах представителей нижегородской школы оптимизации Стронгина Р.Г., Сергеева Я.Д., Гергеля В.П. [50–52, 98–100] предлагаются различные методы вероятностного оценивания константы Липшица в процессе вычислений. Такой подход позволяет расширить спектр применения Липшицевой оптимизации на задачи, в которых аналитическое выражение целевой функции и ограничений неизвестно. В [101] В.П. Булатовым был предложен метод решения задач глобальной оптимизации, основанный на отсечениях второго порядка. Идеи отсечений получили дальнейшее развитие в работах В.П. Булатова и О.В. Хамисова [102, 103]. О.В. Хамисовым был изу-

чен класс функций для которых существуют вогнутые миноранты и предложены методы глобальной оптимизации таких функций [104]. В работе [105] А.Р. Ершовым и О.В. Хамисовым предложен способ автоматического получения кусочно-линейных минорант и мажорант функций, представленных в виде суперпозиции элементарных функций. Авторы демонстрирует эффективность предложенного подхода при решении ряда оптимизационных задач. Дальнейшее развитие эти результаты получили в работах [56, 106].

Также отметим и другие подходы к решению задач глобальной оптимизации [54, 107–109].

Одним из наиболее известных детерминированных методов глобальной оптимизации является *метод неравномерных покрытий* для поиска глобального экстремума функций многих переменных, предложенный в 1971 году [46]. Дальнейшее развитие этот подход нашел в многочисленных работах. Укажем лишь некоторые из них [47, 48, 110, 111]. Различные варианты метода были распараллелены, программно реализованы и использовались для расчетов на многопроцессорных системах [48, 111–113].

В данной главе дано более общее, чем в [46, 47], трактование метода неравномерных покрытий. Приведены новые модификации метода, повышающие его эффективность. Отдельно рассмотрены случай минимизации на простом допустимом множестве и случай функциональных ограничений. Предлагаемый метода применим, в частности, для поиска глобального экстремума в задачах нелинейного программирования, в которых допустимое множество может быть не односвязным и содержать изолированные точки. Метод расширен также на класс задач частично-целочисленного нелинейного программирования. Приведены новые миноранты, основанные на оценке спектра Гессиана целевых функций и ограничений. Получены новые формулы для покрывающих множеств, повышающие эффективность метода. Один из вариантов метода неравномер-

ных покрытий программно реализован на базе пакета BNB-Solver [112] и адаптирован к использованию на многопроцессорной вычислительной технике.

В качестве примера решена задача, в которой глобальный минимум достигается в изолированной точке допустимого множества. Показано, что при использовании предложенного метода введение дополнительного условия целочисленности существенно ускоряет расчеты. Метод также тестировался на задачах безусловной глобальной минимизации, в которых в качестве целевой функции брались полиномы с коэффициентами, сгенерированными случайным образом. Эти расчеты наглядно показывают эффективность предложенного подхода по сравнению с результатами, полученными программами поиска глобального экстремума BARON [114] и LINDOGLOBAL [115] из пакета GAMS [116].

Далее в главе используются следующие обозначения. Компоненты n-мерного вектора x обозначаются верхним индексом, заключенным в круглые скобки: $x = (x^{(1)}, .., x^{(n)})$. Через $\|x\|$ обозначается евклидова норма в пространстве $\mathbb{R}^n$: $\|x\| = \sqrt{\sum_{i=1}^{n}(x^{(i)})^2}$. Запись $|x|$ обозначает вектор $(|x^{(1)}|, .., |x^{(n)}|)$, составленный из абсолютных величин компонент x. Векторные неравенства и бинарные операции над векторами выполняются покомпонентно. Например запись $a \leq b$ означает, что $a_i^{(j)} \leq b_i^{(j)}$ для всех $j = 1, .., n$, а соотношение $c = a + b$ означает, что $c_i^{(j)} = a_i^{(j)} + b_i^{(j)}$ для всех $j = 1, .., n$. Градиент функции $f(x)$ обозначается через $f_x(x)$, а через $f_{xx}(x)$ — матрица Гессе. Для любого подмножества $A \subseteq \mathbb{R}^n$ определим его *диаметр* $d(A) = \sup\{\|x_1 - x_2\|, x_1, x_2 \in A\}$.

1.2 Случай множеств простой структуры

Для задачи (1.1) определим множество *глобальных решений* X_* и множество *ε-оптимальных решений* X_ε:

$$X_* = \{x \in X : f(x) = f_*\}, X_\varepsilon = \{x \in X : f(x) \leq f_* + \varepsilon\}, \varepsilon > 0. \tag{1.2}$$

Предполагается, что множество X_* не пусто. Требуется найти хотя бы одну точку из множества X_ε.

Для множества $Z \subseteq \mathbb{R}^n$, функции $f(x) : \mathbb{R}^n \to \mathbb{R}$ и числа $\lambda \in \mathbb{R}$ определим *Лебеговское множество* $\mathcal{L}(f(\cdot), Z, \lambda) = \{x \in Z : f(x) \geq \lambda\}$ и *открытое Лебеговское множество* $\mathcal{L}'(f(\cdot), Z, \lambda) = \{x \in Z : f(x) > \lambda\}$. Используя введенное понятие, можно определить необходимые и достаточные условия глобальной оптимальности для любой точки $x_* \in X$ следующим образом:

$$x_* \in X_* \Leftrightarrow \mathcal{L}(f(\cdot), X, f(x_*)) = X.$$

Аналогично записывается критерий глобальной ε-оптимальности. Пусть $x_\varepsilon \in X$, тогда

$$x_\varepsilon \in X_\varepsilon \Leftrightarrow \mathcal{L}(f(\cdot), X, f(x_\varepsilon) - \varepsilon) = X. \tag{1.3}$$

Построение лебеговского множества на всем допустимом множестве обычно бывает затруднительным. В таких случаях прибегают к дроблению множества X на подмножества и к построению на них минорант. Рассмотрим набор множеств $\{X_i\}$, $X_i \subseteq X$, $i = 1, \ldots, l$. На каждом множестве X_i определим миноранту $\mu_i(x)$ такую, что $f(x) \geq \mu_i(x)$ для всех $x \in X_i \cap X_*$. Пусть заданы совокупность допустимых точек $N_k = \{x_1, \ldots, x_k\}$ и совокупность множеств $M_k = \{\mathcal{L}_1, \ldots, \mathcal{L}_k\}$, удовлетворяющих условию

$$\mathcal{L}_i \subseteq \mathcal{L}(\mu_i(\cdot), X_i, f(x_i) - \varepsilon), i = 1, \ldots, k. \tag{1.4}$$

Будем говорить, что совокупность множеств M_k *покрывает множество* X_*, если

$$X_* \cap \cup_{i=1}^k \mathcal{L}_i \neq \emptyset. \tag{1.5}$$

Множества $\mathcal{L}_i$ будем называть *покрывающими.*

Теорема 1.1 *Если для наборов допустимых точек N_k и покрывающих множеств M_k выполнено условие (1.5), то точка $x_r = \arg\min_{x \in N_k} f(x)$ является ε-оптимальным решением задачи (1.1) и для него справедлива оценка:*

$$f(x_r) \geq f_* \geq f(x_r) - \varepsilon \tag{1.6}$$

Доказательство. Пусть x_* — одна из допустимых точек, в которых целевая функция принимает минимальное значение: $f_* = f(x_*)$. Из условия (1.5) следует, что найдется по крайней мере одно покрывающее множество $\mathcal{L}_i$ такое, что $x_* \in \mathcal{L}_i$. Тогда из (1.4) следует $f_* = f(x_*) \geq \mu_i(x_*) \geq f(x_i) - \varepsilon \geq f(x_r) - \varepsilon$. Тем самым показана справедливость правого неравенства в (1.6). Левое неравенство является следствием определения минимума функции $f(x)$ на X.□

Теорема 1.1 является теоретической основой для реализации различных вычислительных схем метода неравномерных покрытий. В данной главе рассматривается ситуация, когда множество X и его подмножества X_i являются *множествами простой структуры*, характеризуемыми следующими тремя свойствами:

1. минимум миноранты на множестве простой структуры легко определяется;
2. декомпозиция множества простой структуры на подмножества простой структуры легко реализуется алгоритмически;
3. нахождение допустимой точки внутри множества простой структуры представляет собой простую задачу.

Примером множеств простой структуры являются n-мерный параллелепипед или многогранник, задаваемый системой линейных неравенств. Ограничимся рассмотрением случая параллелепипеда. Рассмотрим одну из возможных реализаций метода неравномерных покрытий:

Алгоритм COVER

1. Поместить X в список S и положить $i := 1$. Выбрать в качестве x_0 произвольную точку из X.
2. Взять из списка S некоторое множество, которое обозначим через X_i.
3. Взять точку $c_i \in X_i$, вычислить $f(c_i)$ и положить
$$x_i = \begin{cases} c_i \text{ если } f(c_i) < f(x_{i-1}), \\ x_{i-1} \text{ в противном случае.} \end{cases}$$
4. Согласно (1.4) определить множество $\mathcal{L}_i$ и его дополнение $X_i' = X_i \setminus \mathcal{L}_i$.
5. Если $X_i' \neq \emptyset$, то разбить X_i' на p подмножеств $\mathcal{Y}_i = \{Y_1^i, \ldots, Y_p^i\}$ и добавить их в список S.
6. Удалить X_i из списка S.
7. Если список S пуст, то завершить работу алгоритма, в противном случае положить $i := i + 1$ и перейти к шагу 2.

При $p = 2$ алгоритм COVER называется также *методом половинных делений* (или бисекций) [47, 107] . Если алгоритм завершился за k итераций цикла 2-7, то найденная точка x_k будет ε-оптимальным решением задачи (1.1). Действительно, завершение работы алгоритма означает, что список S стал пустым, т.е. выполнено условие покрытия (1.5). По теореме 1.1 точка $x_r = \arg\min_{x \in N_k} f(x)$ будет ε-оптимальным решением задачи (1.1). Способ формирования последовательности точек x_i на шаге 3 обеспечивает выполнение равенства $x_k = x_r$. Точку x_i назовем *текущей рекордной точкой*, значение $f(x_i)$ — *текущим рекордом*, а множество X_i — *текущим подмножеством*.

Лебеговское множество $\mathcal{L}(f(\cdot), X, \lambda)$ расширяется при уменьшении λ:

$$\mathcal{L}(f(\cdot), \lambda_1, X) \subseteq \mathcal{L}(f(\cdot), \lambda_2, X),$$

если $\lambda_1 \geq \lambda_2$. Поэтому уменьшение текущего рекорда $f(x_i)$ расширяет множества $\mathcal{L}_i$, тем самым сокращая их количество, требуемое для выполнения условия покрытия (1.5). В процессе расчетов для уменьшения рекорда применяются методы локальной оптимизации, эвристические алгоритмы.

Стандартным способом формирования множества $\mathcal{L}_i$, используемым в работах [47, 48, 107, 110], является следующий:

$$\mathcal{L}_i = \begin{cases} X_i, \text{ если } \min_{x \in X_i} \mu_i(x) \geq f(x_i) - \varepsilon, \\ \emptyset, \text{ в противном случае.} \end{cases} \tag{1.7}$$

При таком способе множество X_i уменьшается только в случае, когда $\mathcal{L}(\mu_i(\cdot), X_i, f(x_i) - \varepsilon) = X_i$. Более эффективные способы построения покрывающего множества рассматриваются в разделе 1.5. Все они удовлетворяют условию:

$$\text{если } \mathcal{L}(\mu_i(\cdot), X_i, f(x_i) - \varepsilon) = X_i, \text{ то } \mathcal{L}_i = X_i, \tag{1.8}$$

т.е. дают не меньшее сокращение множества X_i по сравнению со стандартным способом.

1.3 Поиск глобального минимума в задаче НЛП

Рассмотрим случай, когда в задаче (1.1) допустимое множество задано с помощью функциональных ограничений:

$$X = \{x \in \mathbb{R}^n : g(x) \leq 0_m\}, \tag{1.9}$$

где $g(x) : \mathbb{R}^n \to \mathbb{R}^m$ — непрерывная вектор-функция. В этом случае разбиение множества X на подмножества X_i сложно алгоритмизируется. Поэтому предположим, что допустимое множество X содержится в некотором ограниченном множестве P из R^n простой структуры.

Для $\delta \in \mathbb{R}$ определим *δ-допустимое множество* следующим образом:

$$X^\delta = \{x \in \mathbb{R}^n : g^{(j)}(x) \leq \delta, j = 1, \ldots, m\}. \tag{1.10}$$

Множество X^δ может быть эквивалентным образом определено с использованием функции $\phi(x) = \max(g^{(1)}(x), \ldots, g^{(m)}(x))$: $X^\delta = \{x \in \mathbb{R}^n : \phi(x) \leq \delta\}$.

Так как $g(x)$ — непрерывная функция, то функция $\phi(x)$ также непрерывна. Положим $\underline{\delta} = \inf\{\delta : X^\delta \neq \emptyset\}$, $\overline{\delta} = \sup\{\delta : X^\delta \subseteq P\}$.

При $\underline{\delta} < \delta < \overline{\delta}$ положим

$$f_*^\delta = \min_{x \in X^\delta} f(x).$$

Функция f_*^δ называется *функцией чувствительности*. Хорошо известны следующие ее свойства [117, 118]:

1. $f_*^0 = f_*$; $X^\delta \subseteq X$ при $\underline{\delta} < \delta \leq 0$; $X \subseteq X^\delta$ при $0 \leq \delta < \overline{\delta}$.

2. Множество X^δ не сужается, а функция чувствительности монотонно не возрастает с увеличением δ:

$$X^{\delta_1} \subseteq X^{\delta_2},\ f_*^{\delta_1} \geq f_*^{\delta_2} \text{ для любых } \delta_1, \delta_2,\ \underline{\delta} < \delta_1 \leq \delta_2 < \overline{\delta}. \tag{1.11}$$

3. Для любых δ_0 таких, что $\underline{\delta} < \delta_0 < \overline{\delta}$ функция чувствительности непрерывна справа:

$$\lim_{\delta \to \delta_0 + 0} f_*^\delta = f_*^{\delta_0}. \tag{1.12}$$

Рассмотрим $\varepsilon \in \mathbb{R}_+$ и $\delta_1 \in \mathbb{R}, \delta_2 \in \mathbb{R},\ \underline{\delta} < \delta_1 \leq \delta_2 < \overline{\delta}$. Определим множество

$$X_\varepsilon^{\delta_1, \delta_2} = \{x \in X^{\delta_2} : f(x) \leq f_*^{\delta_1} + \varepsilon\}.$$

В частном случае, $\delta_1 = 0$, $\delta_2 = \delta$ это множество будем называть *множеством ε, δ-оптимальных решений* и обозначать через X_ε^δ:

$$X_\varepsilon^\delta = \{x \in X^\delta : f(x) \leq f_* + \varepsilon\}.$$

По аналогии с (1.3) можно сформулировать следующий критерий для точки $x_\varepsilon^{\delta_2}$ из X^{δ_2}:

$$x_\varepsilon^{\delta_2} \in X_\varepsilon^{\delta_1, \delta_2} \Leftrightarrow \mathcal{L}(f(\cdot), P, f(x_\varepsilon^{\delta_2}) - \varepsilon) \cup \mathcal{L}'(\phi(\cdot), P, \delta_1) = P. \tag{1.13}$$

Действительно, если $x_\varepsilon^{\delta_2} \in X_\varepsilon^{\delta_1, \delta_2}$, то $X^{\delta_1} \subseteq \mathcal{L}(f(\cdot), P, f(x_\varepsilon^{\delta_2}) - \varepsilon)$. Так как $X^{\delta_1} \cup \mathcal{L}'(\phi(\cdot), P, \delta_1) = P$, то $\mathcal{L}(f(\cdot), P, f(x_\varepsilon^{\delta_2}) - \varepsilon) \cup \mathcal{L}'(\phi(\cdot), P, \delta_1) = P$. Пусть правая часть утверждения (1.13) выполнена. Рассмотрим точку $x_*^{\delta_1} \in X^{\delta_1}$ такую, что

$f(x_*^{\delta_1}) = f_*^{\delta_1}$. Так как $X^{\delta_1} \cap \mathcal{L}'(\phi(\cdot), P, \delta_1) = \emptyset$, то $X^{\delta_1} \subseteq \mathcal{L}(f(\cdot), P, f(x_\varepsilon^{\delta_2}) - \varepsilon)$. Следовательно, $f_*^{\delta_1} = f(x_*^{\delta_1}) \geq f(x_\varepsilon^{\delta_2}) - \varepsilon$, т.е. $f(x_\varepsilon^{\delta_2}) \leq f_*^{\delta_1} + \varepsilon$.

Рассмотрим совокупность множеств $X_1, \ldots, X_k$, где все $X_i \subseteq P$. Определим миноранты $\mu_i(x)$ такие, что $f(x) \geq \mu_i(x)$ для всех x из X_i и миноранты $\nu_i(x)$ такие, что $\phi(x) \geq \nu_i(x)$ для всех x из X_i. Пусть заданы совокупность точек $N_k = \{x_1, \ldots, x_k\}$ из X^{δ_2} и совокупность множеств $M_k = \{\mathcal{L}_1, \ldots, \mathcal{L}_k\}$, удовлетворяющих условию

$$\mathcal{L}_i \subseteq \mathcal{L}(\mu_i(\cdot), X_i, f(x_i) - \varepsilon) \cup \mathcal{L}'(\nu_i(\cdot), X_i, \delta_1). \tag{1.14}$$

Условие покрытия в данном случае имеет вид:

$$P = \cup_{i=1}^{k} \mathcal{L}_i. \tag{1.15}$$

Справедлива следующая теорема.

Теорема 1.2 *Если для наборов точек N_k и покрывающих множеств M_k выполнено условие (1.15), то точка $x_r = \arg\min_{x \in N_k} f(x)$ принадлежит множеству $X_\varepsilon^{\delta_1, \delta_2}$ и справедлива оценка:*

$$f_*^{\delta_1} + \varepsilon \geq f(x_r) \geq f_*^{\delta_2}. \tag{1.16}$$

Доказательство. Неравенство $f(x_r) \geq f_*^{\delta_2}$ следует из определения $f_*^{\delta_2}$. Докажем левое неравенство. Пусть $f_*^{\delta_1} = f(x_1)$, где $x_1 \in X^{\delta_1}$. Из условия (1.15) следует что $x_1 \in \mathcal{L}_i$ для некоторого $i : 1 \leq i \leq k$, т.е. $x_1 \in \mathcal{L}(\mu_i(\cdot), X_i, f(x_i) - \varepsilon) \cup \mathcal{L}'(\nu_i(\cdot), X_i, \delta_1)$. Так как $x_1 \in X^{\delta_1}$, то $\nu_i(x_1) \leq \phi(x_1) \leq \delta_1$, т.е. $x_1 \notin \mathcal{L}'(X_i, \nu_i(x), \delta_1)$. Следовательно, $x_1 \in \mathcal{L}(\mu_i(\cdot), X_i, f(x_i) - \varepsilon)$ и $f_*^{\delta_1} = f(x_1) \geq \mu_i(x_1) \geq f(x_i) - \varepsilon \geq f(x_r) - \varepsilon$. Отсюда следует левое неравенство в (1.16). $\square$

Выделим два частных случая теоремы 2. В первом $\delta_1 < 0$, $\delta_2 = 0$, при этом точка x_r является приближенным допустимым решением, для которого справедливо неравенство: $f_*^{\delta_1} + \varepsilon \geq f(x_r) \geq f_*$. Из-за отсутствия непрерывности слева в этом случае приближенное и точное решения могут сколь угодно отли-

чаться. Кроме того, для задач где $\underline{\delta} = 0$ данный подход неприменим, т.к. здесь внутренность множества X пуста.

Во втором случае $\delta_1 = 0$, $\delta_2 > 0$. Положим $\delta = \delta_2$. Точка x_r будет приближенным, но возможно недопустимым решением, для которого справедливо неравенство: $f_* + \varepsilon \geq f(x_r) \geq f_*^{\delta}$. При стремлении ε и δ к нулю значение $f(x_r)$ будет стремиться к f_*. Рассмотрим модификацию базового алгоритма бисекций для задач нелинейного программирования в этом случае:

Алгоритм COVERNLP

1. Поместить P в список S и положить $i := 1$.
2. Взять из списка S некоторое множество, которое обозначим через X_i.
3. Взять точку $c_i \in X_i$, если $\phi(c_i) \leq \delta$, то вычислить $f(c_i)$ и положить
$$x_i = \begin{cases} c_i \text{ если } i = 1 \text{ или } f(c_i) < f(x_{i-1}), \\ x_{i-1} \text{ в противном случае.} \end{cases}$$
4. Определить множество $\mathcal{L}_i$ согласно (1.14) и его дополнение $X_i' = X_i \setminus \mathcal{L}_i$.
5. Если $X_i' \neq \emptyset$, то разбить X_i' на p подмножеств $\mathcal{Y}_i = \{Y_1^i, \ldots, Y_p^i\}$ и добавить их в список S.
6. Удалить X_i из списка S.
7. Если список S пуст, то завершить работу алгоритма, в противном случае положить $i := i + 1$ и перейти к шагу 2.

Если приведенный алгоритм завершился за k итераций цикла 2-7, то найденная точка x_k будет ε, δ-оптимальным решением задачи (1.1), где допустимое множество задается формулой (1.9). Если алгоритм COVERNLP завершился и не было найдено ни одной точки c_i такой, что $\phi(c_i) \leq \delta$, то значит множество X пусто.

Далее в работе будет предполагаться, что множество X (в алгоритме COVER), множества P, X_i, X_i' — n-мерные параллелепипеды.

1.4 Миноранты

Если функция $f(x)$ удовлетворяет условию Липшица с константой l_i на множестве X_i, то согласно [46] следующая функция будет минорантой для функции $f(x)$:

$$\mu_i(x) = f(c_i) - l_i\|x - c_i\|. \tag{1.17}$$

Если градиент функции $f(x)$ удовлетворяет условию Липшица, то согласно [48], миноранта для функции $f(x)$ имеет вид:

$$\mu_i(x) = f(c_i) + \langle f_x(c_i), x - c_i\rangle - \frac{L_i}{2}\|x - c_i\|^2, \tag{1.18}$$

где $\|f_x(x) - f_x(y)\| \le L_i\|x - y\|$ для любых $x, y \in X_i$.

Заметим, что $L_i \ge \sup_{x\in X_i}\|f_{xx}(x)\|$. Обычно принимают $L_i \ge \max(|k_i|, |K_i|)$, где k_i и K_i — такие числа, что спектр матрицы Гессе $f_{xx}(x)$ при $x \in X_i$ полностью лежит в пределах отрезка $[k_i, K_i]$. Использование чисел k_i, K_i позволяет строить более точные миноранту $\mu_i(x)$ и мажоранту $\mathcal{M}_i(x)$ для функции $f(x)$:

$$\mu_i(x) = f(c_i) + \langle f_x(c_i), x - c_i\rangle + \frac{k_i}{2}\|x - c_i\|^2, \tag{1.19}$$

$$\mathcal{M}_i(x) = f(c_i) + \langle f_x(c_i), x - c_i\rangle + \frac{K_i}{2}\|x - c_i\|^2. \tag{1.20}$$

Лебеговское множество $\mathcal{L}(\mu_i(x), X_i, f(x_i) - \varepsilon)$ состоит из точек X_i, удовлетворяющих условию:

$$\alpha(x) = f(c_i) + \langle f_x(c_i), x - c_i\rangle + \frac{k_i}{2}\|x - c_i\|^2 - f(x_i) + \varepsilon \ge 0. \tag{1.21}$$

Заметим, что $\alpha_x(x) = f_x(c_i) + k_i(x - c_i)$. При $k_i \ne 0$ уравнение $\alpha_x(x) = 0$ имеет единственное решение $z_i = c_i - \frac{1}{k_i}f_x(c_i)$. После замены переменных $x = y + z_i$ неравенство (1.21) примет вид:

$$\frac{k_i}{2}\|y\|^2 \ge \frac{1}{2k_i}\|f_x(c_i)\|^2 - f(c_i) + f(x_i) - \varepsilon.$$

Лебеговское множество имеет различный вид в зависимости от значения k_i. Рассмотрим три случая.

1. $k_i < 0$. Лебеговское множество является пересечением X_i и шара с центром в точке c_i с радиусом ρ_i, вычисляемым по формуле

$$\rho_i = \sqrt{\frac{2}{k_i}\left(\frac{1}{2k_i}\|f_x(c_i)\|^2 - f(c_i) + f(x_i) - \varepsilon\right)}. \tag{1.22}$$

2. $k_i = 0$. Лебеговское множество есть пересечение X_i и полупространства, определяемого неравенством

$$\langle f_x(c_i), x - c_i\rangle + f(c_i) - f(x_i) + \varepsilon \geq 0.$$

3. $k_i > 0$. Если $\frac{1}{2k_i}\|f_x(c_i)\|^2 - f(x_i) + f(x_r) - \varepsilon < 0$, то Лебеговское множество совпадает с X_i. В противном случае оно является пересечением X_i и внешности (с границей) шара радиуса, определяемого формулой (1.22).

Числа k_i и K_i можно оценить с помощью теоремы Гершгорина [119]. Согласно этой теореме для любого действительного собственного числа γ матрицы (a_{ij}) справедливы неравенства:

$$\min_{i=1,..,n}\left(a_{ii} - \sum_{j=1,j\neq i}^{n}|a_{ij}|\right) \leq \lambda \leq \max_{i=1,..,n}\left(a_{ii} + \sum_{j=1,j\neq i}^{n}|a_{ij}|\right)$$

Следовательно, для оценки границ спектра можно использовать неравенства

$$k_i \geq \min_{i=1,..,n}\left(\underline{u_{ii}} - \sum_{j=1,j\neq i}^{n} v_{ij}\right), K_i \leq \max_{i=1,..,n}\left(\overline{u_{ii}} + \sum_{j=1,j\neq i}^{n} v_{ij}\right), \tag{1.23}$$

где числа $\underline{u_{ij}}$, $\overline{u_{ij}}$ удовлетворяют соотношениям

$$\underline{u_{ij}} \leq \min_{x\in X_i}\frac{\partial f(x)}{\partial x^i \partial x^j}, \;\; \max_{x\in X_i}\frac{\partial f(x)}{\partial x^i \partial x^j} \leq \overline{u_{ij}}, \;\; v_{ij} = \max(|\underline{u_{ij}}|, |\overline{u_{ij}}|)$$

Оценки $\underline{u_{ij}}$, $\overline{u_{ij}}$ могут быть получены различными способами, например, при помощи техники интервального анализа.

Для множеств X_i, содержащих только внутренние точки допустимого множества X_i, можно получить более точную миноранту, если учесть условие оптимальности первого порядка. Для случая оптимизации на множествах простой

структуры, когда X и X_i — параллелепипеды, принадлежность X_i внутренности X проверяется тривиально.

Пусть $c_i \in X_i, x_* \in X_* \cap X_i$ и X_i содержит только внутренние точки допустимого множества X. Тогда $f_x(x_*) = 0$ и мажоранта (1.20), построенная в точке x_*, примет вид

$$\mathcal{M}(x) = f(x_*) + \frac{K_i}{2}\|x - x_*\|^2.$$

Для всех $x \in X_i$ выполняется $f(x) \leq \mathcal{M}(x)$. Поэтому $f(c_i) \leq \mathcal{M}(c_i)$. Отсюда получим, что $f(x_*) \geq f(c_i) - \frac{K_i}{2}\|x_* - c_i\|^2$. Таким образом построена новая миноранта

$$\hat{\mu}_i(x) = f(c_i) - \frac{K_i}{2}\|x_* - c_i\|^2. \tag{1.24}$$

для функции $f(x)$ на множестве $X_* \cap X_i$. Лебеговское множество определяется следующим образом:

$$\mathcal{L}(\hat{\mu}_i(\cdot), X_i, f(x_i) - \varepsilon) = \{x \in X_i : f(c_i) - \frac{K_i}{2}\|x_* - c_i\|^2 \geq f(x_i) - \varepsilon\}.$$

Заметим, что если $K_i \leq 0$, то $\mathcal{L}(\hat{\mu}_i(\cdot), X_i, f(x_i) - \varepsilon) = X_i$, поскольку $f(x_i) \leq f(c_i)$. Если $K_i > 0$, то Лебеговское множество есть пересечение X_i с шаром, который имеет центр в точке c_i и радиус $\rho_i = \sqrt{\frac{2}{K_i}\left(f(c_i) - f(x_i) + \varepsilon\right)} \geq \sqrt{2\varepsilon/K_i}$.

Также условия оптимальности первого порядка могут быть учтены следующим образом. Для установления выполнимости условия покрытия достаточно показать, что $X_* \cap \cup_{i=1}^{k} \mathcal{L}_i \neq \emptyset$. Поэтому подмножества, заведомо не содержащие точек из X_*, могут быть исключены из рассмотрения. Если X_i содержит только внутренние точки множества X_i и градиент целевой функции удовлетворяет условию Липшица с константой L_i, то шар с центром в точке c_i и радиусом $\|f_x(c_i)\|/L_i$ может быть исключен из рассмотрения, т.к. в точках этого шара градиент отличен от нуля.

1.5 Обработка подмножеств в алгоритмах COVER и COVERNLP

В процессе работы алгоритмов COVER, COVERNLP, обрабатываются подмножества X_i множества X. Определяется подмножество $\mathcal{L}_i$ Лебеговского множества и его дополнение $X_i' = X_i \backslash \mathcal{L}_i$. Для корректной работы алгоритмов COVER, COVERNLP необходимо, чтобы множества X_i' были n-мерными параллелепипедами. Положим $X_i = \{x \in \mathbb{R}^n : a_i \leq x \leq b_i\}$, $c_i = (a_i + b_i)/2$.

Для рассмотренных выше минорант возможны три вида Лебеговских множеств:

1. пересечение полупространства и параллелепипеда X_i;

2. дополнение шара до параллелепипеда X_i;

3. пересечение шара и параллелепипеда X_i.

Рассмотрим способы построения покрывающих множеств для каждого из этих случаев.

1. Рассмотрим Лебеговское множество, которое является пересечением полупространства $l(x) = \langle p, x\rangle + q \geq 0$, где $p, x \in \mathbb{R}^n, q \in \mathbb{R}$ и параллелепипеда X_i. Минимум функции $l(x)$ на X_i достигается в точке z_i, определяемой соотношениями:

$$z_i^{(j)} = \begin{cases} a_i^{(j)}, \text{ если } p^{(j)} \geq 0, \\ b_i^{(j)}, \text{ если } p^{(j)} < 0, \end{cases} \quad j = 1, .., n.$$

Положим $v_i = \min_{x \in X_i} l(x) = \langle p, z_i\rangle + q$. Если $v_i \geq 0$ то положим $\mathcal{L}_i = X_i$.

Если $v_i < 0$, то определим параллелепипед $I_i = I_i^1 \times \cdots \times I_i^n$, где

$$I_i^j = \begin{cases} [b_i^{(j)} - v_i/p^{(j)}, b_i^{(j)}] \text{ при } p^{(j)} < 0, \\ [a_i^{(j)}, a_i^{(j)} - v_i/p^{(j)}] \text{ при } p^{(j)} > 0, \\ [a_i^{(j)}, b_i^{(j)}] \text{ при } p^{(j)} = 0. \end{cases}$$

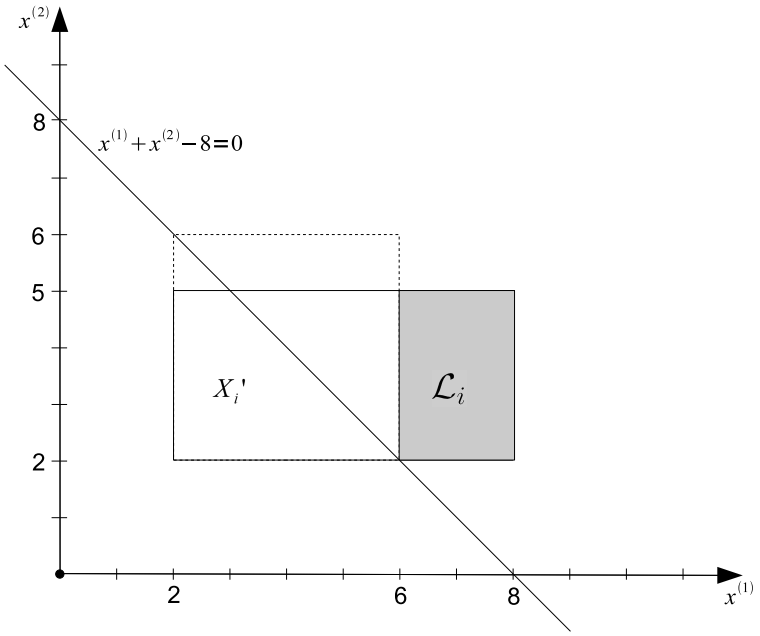

Рис. 1.1: Исключение части параллелепипеда с учетом линейных неравенств

Несложно показать, что $l(x) > 0$ для любых $x \in X_i \setminus I_i$. Поэтому можно положить $X_i' = X_i \cap I_i$, $\mathcal{L}_i = X_i \setminus X_i'$. При этом множество X_i' является параллелепипедом.

Рис. 1.1 иллюстрирует этот случай для параллелепипеда $X_i = [(2,2),(8,5)]$ и $l(x) = x^{(1)} + x^{(2)} - 8 \geq 0$.

2. Пусть Лебеговское множество является дополнением шара B_i с центром z_i и радиусом ρ_i до параллелепипеда X_i. Рассмотрим куб $C_i = \{x \in \mathbb{R}^n : z_i^{(j)} - \rho_i \leq x_i^{(j)} \leq z_i^{(j)} + \rho_i, j = 1,..,n\}$, описанный около шара B_i. Так как $X_i \setminus C_i \subseteq X_i \setminus B_i$, то положим $X_i' = C_i \cap X_i$, $\mathcal{L}_i = X_i \setminus X_i'$. Множество X_i' — параллелепипед, границы которого легко находятся.

3. Пусть Лебеговское множество представляет собой пересечение параллелепипеда X_i и шара B_i с центром в точке z_i и радиусом ρ_i. Выберем базисный вектор $e_s = \{\underbrace{0,\ldots,0}_{s-1},1,0,\ldots,0\} \in \mathbb{R}^n$, перпендикулярно которому будет производиться дробление параллелепипеда, где $s = arg\max_{j=1,\ldots,n}(b_i^{(j)} - a_i^{(j)})$.

Максимальное из расстояний от точки z_i до прямых, проходящих вдоль ребер

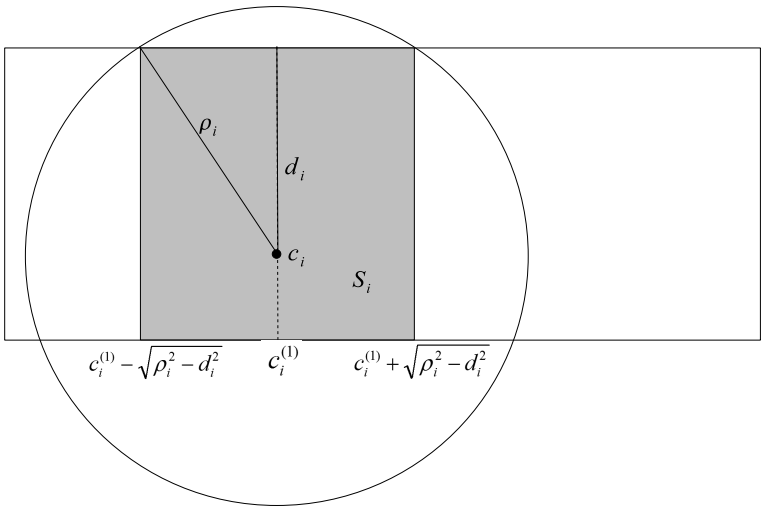

Рис. 1.2: Исключение части параллелепипеда, лежащей внутри шара

параллелепипеда X_i параллельно вектору e_s, определяется по формуле:

$$d_i = \sqrt{\sum_{j=1}^{n} \left(d_i^{(j)}\right)^2}, \text{ где } d_i^{(j)} = \begin{cases} \max(|a_i^{(j)} - z_i^{(j)}|, |b_i^{(j)} - z_i^{(j)}|), \text{ при } j \neq s, \\ 0, \text{ при } j = s. \end{cases} \tag{1.25}$$

При $d_i < \rho_i$, в качестве покрывающего множества возьмем параллелепипед $\mathcal{L}_i = I_i^1 \times \cdots \times I_i^n$, где

$$I_i^j = \begin{cases} \left[a_i^{(j)}, b_i^{(j)}\right], \text{ при } j \neq s, \\ \left[z_i^{(s)} - \sqrt{\rho_i^2 - d_i^2}, z_i^{(s)} + \sqrt{\rho_i^2 - d_i^2}\right] \text{ при } j = s. \end{cases}$$

Несложно видеть, что $\mathcal{L}_i \subseteq B_i \cap X_i$. Этот случай проиллюстрирован на Рис. 1.2 при $n = 2$.

Дополнение $X_i' = X_i \setminus \mathcal{L}_i$ может состоять из одного или более параллелепипедов. Если X_i' — параллелепипед, то он разбивается на два равных параллелепипеда вдоль наибольшего ребра.

В задачах частично-целочисленного программирования на допустимое множество X помимо функциональных ограничений (1.9) накладывается требование целочисленности вектора x либо некоторых его компонент: $x^{(j)} \in \mathbb{Z}, \ \ j \in J$, где J — подмножество индексов целочисленных компонент вектора x. Без огра-

ничения общности можно считать, что $a^{(j)} \in \mathbb{Z}, b^{(j)} \in \mathbb{Z}$ при $j \in J$.

Обозначим через $\lfloor x \rfloor$ наибольшее целое число, не превосходящее x, а через $\lceil x \rceil$ обозначено наименьшее целое число, большее или равное x. На шаге 5 алгоритма COVERNLP целочисленность позволяет уменьшить размер множеств, добавляемых в список следующим образом: $a_i^{(j)} := \lceil a_i^{(j)} \rceil$, $b_i^{(j)} := \lfloor b_i^{(j)} \rfloor$, для всех индексов j из множества J.

Целочисленность также учитывается при определении рекордной точки x_i на шаге 3. Перед ним точка c_i подвергается следующему преобразованию:

$$c_i^{(j)} := \begin{cases} \lfloor c_i^{(j)} \rfloor, \text{ если } j \in J, \\ c_i^{(j)}, \text{ если } j \notin J. \end{cases}$$

Остальные шаги алгоритма остаются без изменений. Таким образом учет условия целочисленности позволяет уменьшить параллелепипеды, получаемые в результате разбиения и, соответственно, сократить число итераций алгоритма. Ниже это свойство демонстрируется на примере.

1.6 Результаты вычислительных экспериментов

Алгоритмы COVER и COVERNLP программно реализованы на базе пакета BNB-Solver [112]. При решении конкретной задачи требуется реализовать модули, вычисляющие значения функции $f(x)$, ее градиент, функции $g^{(i)}(x)$, их градиенты, оценки констант Липшица и границ спектра для целевой функции и ограничений. Для полиномов от нескольких переменных перечисленные объекты могут быть автоматически вычислены. Поэтому были добавлены модули, поддерживающие полиномиальные целевые функции и ограничения.

Поиск изолированного минимума

Рассмотрим следующий пример из [97]:

$$f(x) = x^{(1)} \to \min, \tag{1.26}$$

$$g^{(1)}(x) = (x^{(1)} - 5)^2 + 2(x^{(2)} - 5)^2 + (x^{(3)} - 5)^2 - 18 \leq 0, \tag{1.27}$$

$$g^{(2)}(x) = 100 - (x^{(1)} + 7 - 2x^{(2)})^2 - 4(2x^{(1)} + x^{(2)} - 11)^2 - 5(x^{(3)} - 5)^2 \leq 0. \tag{1.28}$$

Особенностью данной задачи является тот факт, что глобальный минимум достигается в изолированной допустимой точке $x_* = (1, 4, 5)$, где $f(x_*) = 1$, $g^{(1)}(x_*) = g^{(2)}(x_*) = 0$, (в [97] ошибочно указана точка $(1, 3, 5)$). Использованный в [97] метод поиска глобального минимума определил допустимую точку $\overline{x} = (3.747692, 7.171420, 2.362317)$, где $f(\overline{x}) = 3.747692$. Вычисления заняли 33.703 секунды на компьютере Pentium IV 2.53 GHz

Для применения метода неравномерных покрытий в качестве множества P был выбран параллелепипед $-10 \leq x^{(i)} \leq 10, i = 1, 2, 3$, заведомо содержащий допустимое множество рассматриваемой задачи. Использовалась миноранта (1.19).

На Рис. 1.3 показана проекция допустимой области задачи на плоскость $x^{(3)} = 5$. Оно представляет собою пересечение внутренней части эллипса, определяемой ограничением (1.27), и внешней части другого эллипса, соответствующей ограничению (1.28). Это пересечение состоит из заштрихованной области и точки x_*, где эллипсы касаются. На Рис. 1.3(II) заштрихована δ-допустимая область X^δ при $\delta > 0$. Эта область содержит точку минимума x_* вместе с некоторой окрестностью. При $\delta < 0$ множество X^δ представляет собою односвязное множество и изолированная точка x_* ему не принадлежит.

Эксперименты проводились на персональном компьютере с процессором Intel Core 2 Quad, 2.33 GHz. Использовалось одно ядро процессора, полагалось $\varepsilon = |\delta|$.Результаты экспериментов приведены в таблице 1.1, значение $f(x_r)$ совпа-

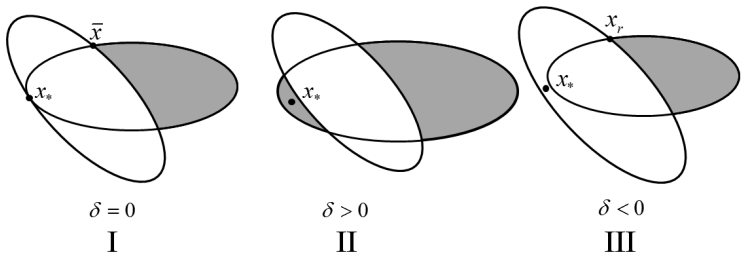

Рис. 1.3: Множество X^δ (заштриховано) при различных δ

дает с первой компонентой вектора x_r, приведенного во втором столбце.

При $\delta_1 = 0, \delta_2 = \delta > 0$ получены приближенные решения, близкие к оптимальному решению x_*. Найденные при $\delta_1 = \delta < 0, \delta_2 = 0$ точки допустимы и близки к точке $\overline{x}$, при этом полученное значение целевой функции меньше, чем найденное в [97] в третьей значащей цифре.

Таблица 1.1: Результаты расчетов при различных значениях δ

δ	рекорд x_r	$\phi(x_r)$	время расчетов (с)	число итераций
-0.01	$(3.722, 7.276, 7.452)$	-0.00204	0.61	10602
-0.0001	$(3.721, 7.15, 2.331)$	-0.00001	24.57	506351
0.0001	$(0.997, 4.007, 4.999)$	0.00007	7.94	165547
0.01	$(0.965, 4.071, 4.994)$	0.00993	0.16	2671

Так как минимум в данной задаче достигается в точке с целыми координатами, то рассматриваемый пример был также использован для тестирования возможностей нахождения целочисленного решения. Для этого на все переменные было наложено условие целочисленности. Расчеты проводились в трех различных вариантах при $\varepsilon = \delta = 0$. В первом использовалась базовая схема алгоритма COVERNLP, применялись стандартные липшицевы миноранты (1.17). Во втором варианте использовались миноранты (1.19). В третьем варианте дополнительно применялась техника определения покрывающих множеств, рассмотренная в разделе 1.5. Глобальный минимум был найден за 585, 121, 55 итераций в первом, втором и третьем вариантах соответственно. Таким обра-

зом, в данной задаче введение условия целочисленности существенно ускоряет расчеты. Поэтому при решении сложных задач представляется целесообразным вводить это условие искусственно для нахождения приближенного решения исходной непрерывной задачи с целью ускорения расчетов и получения близких к оптимуму рекордных значений.

Тестирование полиномах со случайными коэффициентами

Эксперименты проводились для задач безусловной оптимизации, в которых в качестве целевой функции были выбраны полиномы нескольких переменных, коэффициенты которых генерировались случайным образом. Рассматривались полиномы вида:

$$f(x) = \sum_{i=1}^{n} a(x^{(i)})^m + \sum_{d \in D} a_d (x^{(i_1)})^{d_1} \dots (x^{(i_n)})^{d_n}, \tag{1.29}$$

где n размерность вектора x, m — четная степень полинома, $a > 0$ — заданное фиксированное число, $D = \{(d_1, \dots, d_n) : d_i \in \mathbb{Z}_+, \sum_{i=1}^{n} d_i \leq m - 1\}$ — множество кортежей степеней мономов, числа a_d равномерно распределены в интервале $[0, a]$.

Несложно показать, что при четном m полиномы указанного вида имеют глобальный оптимум, который лежит в пределах параллелепипеда $[-M, M]^n$, где $M = |D|$ — мощность множества кортежей D. Указанный параллелепипед выбирался в качестве допустимого множества X.

Эксперименты проводились для 5 серий, каждая из которых содержала 10 тестовых задач, везде $a = 10$:

- Серия 1: $n = 3$, $m = 4$;
- Серия 2: $n = 3$, $m = 6$;
- Серия 3: $n = 3$, $m = 8$;
- Серия 4: $n = 4$, $m = 4$;

- Серия 5: $n = 4$, $m = 6$.

В таблице 1.2 приводятся минимальное (MIN), максимальное (MAX) и среднее (AVR) время расчетов в секундах для каждой из серий при решении методами, использующими

- O0 — стандартную Липшицева миноранту (1.17);
- O1 — миноранту (1.18);
- O2 — миноранту (1.19);
- O3 — миноранту (1.24);
- O3+ — миноранту (1.24) в сочетании с техникой сокращения области поиска (пункт 1.5);

В вариантах O0–O3 покрывающее множество определялось по правилу (1.7). В колонках BR и LG приводятся значения времени решения тестовых примеров, полученные с помощью программ поиска глобального экстремума BARON [114] и LINDOGLOBAL [115] из пакета GAMS [116].

В таблице используются следующие обозначения:

T — вычисления продолжались слишком долго и были остановлены;

A — программа завершила счет аварийно;

E — результат расчетов неверен.

Анализ результатов, приведенных в таблице, показывает, что миноранты, введенные в разделе 1.4 и методы построения покрывающих множеств, рассмотренные в разделе 1.5, дают заметное ускорение расчетов по сравнению со стандартной техникой. Наилучшие результаты были получены при использовании миноранты (1.24) в сочетании с сокращением области поиска методами, приведенными в разделе 1.5. При этом решение задачи осуществляется гораздо быстрее по сравнению с программами BARON и LINDOGLOBAL, которые в

Таблица 1.2: Время расчетов (с) для различных серий полиномов

Серия		O0	O1	O2	O3	O3+	BR	LG
1	AVR	5.399	0.73	0.49	0.08	0.05	1.07	5.42
	MAX	15.12	1.77	1.15	0.12	0.07	1.36	6.02
	MIN	0.66	0.33	0.11	0.07	0.04	0.65	3.43
2	AVR	T	11.24	8.52	0.54	0.29	4.86	E
	MAX	T	28.92	39.17	0.67	0.38	6.56	E
	MIN	T	4.03	2.09	0.46	0.24	3.68	E
3	AVR	T	T	T	2.2	1.38	A	E
	MAX	T	T	T	2.49	1.75	A	E
	MIN	T	T	T	1.9	1.15	A	E
4	AVR	T	107.31	44.15	0.94	0.54	3.51	23.63
	MAX	T	350.07	120.49	1.46	0.77	3.89	27.71
	MIN	T	22.03	10.49	0.76	0.41	3.02	20.54
5	AVR	T	T	T	11.72	5.85	A	E
	MAX	T	T	T	14.11	7.14	A	E
	MIN	T	T	T	9.93	4.7	A	E

ряде случаев не находили решение из-за аварийного завершения или выдавали заведомо не оптимальное решение.

1.7 Основные результаты главы

Данная глава диссертации посвящена развитию метода неравномерных покрытий для задач скалярной оптимизации. Получены следующие результаты:

1. предложены новые миноранты, основанные на оценке спектра матрицы Гессе, разработан метод вычисления таких оценок;
2. для задач оптимизации с параллелепипедными ограничениями предложены новые миноранты, учитывающие необходимые условия оптимальности;
3. предложены способы сокращения области поиска, основанные на построении дополнения покрывающего множества до параллелепипеда, позволяющие существенно ускорить метод неравномерных покрытий;
4. метод неравномерных покрытий перенесен на задачи

частично-целочисленного программирования, доказана корректность алгоритма;

5. приведены результаты экспериментов, демонстрирующие эффективность предложенного подхода.

Результаты данной главы опубликованы в работах [120–123].

Глава 2

Детерминированные методы решения задач многокритериальной оптимизации

2.1 Постановка задачи и обзор существующих результатов

В настоящее время опубликовано большое число работ по многокритериальной оптимизации (МКО). Общую информацию по методам решения таких задач можно найти в монографиях [65–67, 124, 125]. Подробный обзор современного состояния исследований по этой теме содержится в [126].

Методы решения задач многокритериальной оптимизации можно условно разделить на несколько групп. Первую группу образуют методы, основанные на сведении задачи многокритериальной оптимизации к решению серии оптимизационных задач с одним критерием. Простейший вариант такого сведения состоит в использовании линейной свертки критериев [127]. Обзоры по методам этого класса можно найти в работах [75, 128], а некоторые примеры использования в работах [74, 129–132].

Эволюционные алгоритмы [69, 133–135] являются еще одним перспективным подходом к решению задач многокритериальной оптимизации. Термин “эволюционные алгоритмы” обозначает класс стохастических методов оптимизации, которые имитируют процесс естественной эволюции. Такие алгоритмы используют множество точек в пространстве параметров (“популяцию”), при котором

более одного решения участвует в итерации и на каждой итерации развивается новая популяция решений. Конечная совокупность точек, полученная в результате, служит приближением множества эффективных решений. Среди эволюционных подходов следует выделить генетические алгоритмы [136], получившие широкое распространение в последнее время. Также существуют гибридные методы [61], основанные на комбинации различных подходов.

Во многих методах используются локальные процедуры улучшения решений, основанные на принципах оптимальности. Такие методы и критерии оптимальности рассмотрены в работах [137–139].

Хотя методы, описанные выше, могут эффективно справляться с трудными проблемами МКО они являются недетерминированными, т.е. не гарантируют точность получаемых приближений. Несколько детерминированных методов были предложены для линейных и выпуклых проблем МКО. В работе [72] предложен алгоритм последовательной внешней аппроксимации, способный генерировать приближения гарантированной точности для линейных задач МКО. Этот алгоритм был усовершенствован в [70,140]. Расширение алгоритма Бенсона, позволяющее строить приближения гарантированной точности для произвольных выпуклых задач, приведены в работе [141]. . Различные методы решения многокритериальных задач дискретной оптимизации, позволяющие контролировать точность получаемых приближений, предложены в работах [78, 79, 142, 143].

Ю.Г. Евтушенко и М.А. Потаповым был предложен алгоритм (вариант метода неравномерных покрытий), позволяющий строить приближенное решение задачи многокритериальной оптимизации с ограничениями параллелепипедного типа и гарантировать точность (ε-оптимальность) построенного приближения [76]. При этом критерии могут быть произвольными функциями, удовлетворяющими условию Липшица. Для построения оценок используются Липшицевы

миноранты первого порядка. В диссертации результаты этой работы получили развитие по следующим направлениям:

1. доказаны ряд новых свойств множества ε-Парето;

2. оригинальный метод расширен на случай произвольных минорант;

3. метод перенесен на задачи с ограничениями.

Далее будем использовать следующие обозначения. Компоненты n-мерного вектора x обозначаются верхним индексом, заключенным в круглые скобки: $x = (x^{(1)}, .., x^{(n)})$. Через $\|x\|$ обозначается евклидова норма в пространстве $\mathbb{R}^n$: $\|x\| = \sqrt{\sum_{i=1}^{n}(x^{(i)})^2}$. Запись $|x|$ обозначает вектор $(|x^{(1)}|, .., |x^{(n)}|)$, составленный из абсолютных величин компонент x.

Векторные неравенства и бинарные операции над векторами выполняются покомпонентно. Например для векторов $a, b, c \in \mathbb{R}^m$ запись $a \leq b$ означает, что $a^{(j)} \leq b^{(j)}$ для всех $j = 1, .., m$, а соотношение $c = a + b$ означает, что $c^{(j)} = a^{(j)} + b^{(j)}$ для всех $j = 1, .., m$. Шаром радиуса ρ с центром в точке c в пространстве $\mathbb{R}^n$ будем называть следующее множество: $B(c, \rho) = \{x \in \mathbb{R}^n : \|x - c\| \leq \rho\}$. Градиент функции $f(x)$ обозначается через $f_x(x)$, а Гессиан — $f_{xx}(x)$.

В данной главе рассматривается задача многокритериальной оптимизации, формулируемая следующим образом:

$$\min_{x \in X} F(x), \tag{2.1}$$

где X — допустимое множество, а вектор-функция $F(\cdot) : \mathbb{R}^n \to \mathbb{R}^m$ определяет векторный критерий, компоненты которого $f^{(1)}(\cdot), \ldots, f^{(m)}(\cdot)$ составляют набор из m скалярных критериев. Образ $Y = F(X)$ допустимого множества X при отображении F назовем *множеством достижимых критериев*. Всюду далее вектор-функция $F(\cdot)$ предполагается непрерывной, а множество X — непустым

компактом. При сделанных предположениях множество Y также будет компактом.

Для произвольной точки $z \in \mathbb{R}^m$ определим *юго-западное* $\mathrm{SW}(z)$ и *северо-восточное* $\mathrm{NE}(z)$ множества следующим образом:

$$\mathrm{SW}(z) = \{y \in \mathbb{R}^m : y \leq z\}, \;\; \mathrm{NE}(z) = \{y \in \mathbb{R}^m : y \geq z\}.$$

Будем говорить, что точка z *минорирует* все точки из множества $\mathrm{NE}(z)$.

Для произвольного множества $Z \subseteq \mathbb{R}^m$ определим его *Парето-оптимальное подмножество* (или проще *Паретовское подмножество*) $\mathcal{P}(Z)$ следующим образом:

$$\mathcal{P}(Z) = \{z \in Z : Z \cap \mathrm{SW}(z) = z\}.$$

Парето-оптимальное подмножество $\mathcal{P}(Z)$ является объединением таких точек z множества Z, для которых пересечение каждого множеств Z и $\mathrm{SW}(z)$ состоит только из одной точки z.

Отметим важное свойство оператора $\mathcal{P}$, справедливое для произвольного множества $Z \subseteq \mathbb{R}^m$:

$$\mathcal{P}\left(\mathcal{P}(Z)\right) = \mathcal{P}(Z). \tag{2.2}$$

Расширим определения $\mathrm{SW}(z), \mathrm{NE}(z)$ на случай произвольного множества $Z \subseteq \mathbb{R}^m$:

$$\mathrm{SW}(Z) = \cup_{z \in Z} \mathrm{SW}(z), \;\; \mathrm{NE}(Z) = \cup_{z \in Z} \mathrm{NE}(z).$$

Множество $\mathrm{NE}(Z)$ называют *оболочкой Эджворта-Парето* множества Z.

Лемма 2.1 *Для любого непустого компактного множества Z в пространстве $\mathbb{R}^m$ справедливо*

$$Z \subseteq NE(\mathcal{P}(Z)). \tag{2.3}$$

Доказательство. Пусть точка $z \in Z$. Рассмотрим множество $Y = SW(z) \cap Z$. Очевидно, Y компактно. Рассмотрим функцию $\phi(z) = \sum_{i=1}^{m} z^{(i)}$. В силу компактности Y эта функция принимает на нем минимальное значение ϕ_*. Пусть

z_* — точка в которой этот минимум достигается: $\phi(z_*) = \phi_*$. Неравенство $z_* \leq z$ выполняется по определению множества Y. Покажем, что точка z_* принадлежит $\mathcal{P}(Z)$. Предположим обратное. Тогда найдется точка $v \in Z$, такая что $v \leq z_*$ и $v \neq z_*$. Поэтому $\phi(v) < \phi(z_*)$. Так как $v \leq z_* \leq z$, то $v \in Y$, а то это противоречит тому, что в точке z_* функция $\phi(\cdot)$ достигает минимума. Таким образом построена точка z_* из $P(Z)$, такая что $z_* \leq z$. В силу произвольности выбора точки z утверждение леммы доказано. $\square$

Множество, полученное в результате применения введенного оператора $\mathcal{P}$ к множеству достижимых критериев Y назовем *множеством Парето* для задачи (2.1) и обозначим через $Y_* = \mathcal{P}(Y)$. Таким образом Y_* есть такое подмножество Y, что для всякой точки y_* ее юго-западное множество содержит только одну эту точку из всего множества Y. Пусть $A_* \subseteq X$ — прообраз множества Y_* при отображении F, т.е. $Y_* = F(A_*)$. Решение задачи (2.1) состоит в нахождении множества A_*. Приведенные определения не накладывают ограничений на мощность множества X. Оно может быть континуальным, счетным или конечным. При сделанных предположениях о компактности множества X и непрерывности $F(\cdot)$ множество Y_* не пусто.

2.2 Случай ограничений параллелепипедного типа

Определение и свойства множества ε-Парето

В данном разделе рассмотрим частный случай задачи (2.1), когда множество X является n-мерным параллелепипедом. Следуя [76], определим понятие приближенного решения задачи (2.1). Для $\varepsilon \geq 0$ дискретный набор точек $Y_\varepsilon \subseteq Y$ будем называть *множеством ε-Парето*, если:

1. для любой точки $y_* \in Y_*$ существует такая точка $y_\varepsilon \in Y_\epsilon$, что

$$y_\varepsilon - \varepsilon \cdot e_m \leq y_*, \tag{2.4}$$

2.

$$\mathcal{P}(Y_\varepsilon) = Y_\varepsilon. \tag{2.5}$$

Здесь вектор e_m обозначает вектор из пространства $\mathbb{R}^m$, все компоненты которого равны 1.

Пункты 1,2 будем называть *первым и вторым условиями оптимальности* соответственно. Второе условие позволяет отбросить лишние точки из набора точек Y_ε. Множество $A_\varepsilon \subseteq X$, такое что $f(A_\varepsilon) = Y_\varepsilon$ называется ε-*оптимальным решением* задачи (2.1).

Считаем, что задача (2.1) решена с заданной точностью ε, если найдено множество ε-Парето Y_ε и его прообраз A_ε.

Лемма 2.2 *Для любой точки y из оболочки Эджворта-Парето $NE(Y)$ найдется такая точка $y_\varepsilon \in Y_\varepsilon$, что*

$$y_\varepsilon - \varepsilon \cdot e_m \leq y. \tag{2.6}$$

Доказательство. Пусть $y \in \mathrm{NE}(Y)$. Тогда найдется такая точка $y' \in Y$, что $y \in \mathrm{NE}(y')$, т.е. $y' \leq y$. Согласно лемме 2.1 найдется точка $y_* \in Y_*$ такая, что $y_* \leq y'$. Согласно (2.4) для точки y_* найдется такая точка y_ε что $y_\varepsilon - \varepsilon \cdot e_m \leq y_*$. Таким образом, $y_\varepsilon - \varepsilon \cdot e_m \leq y_* \leq y' \leq y$ и, следовательно, неравенство (2.6) имеет место. □

Из (2.4) следует, что множество ε-Парето является одновременно множеством $\overline{\varepsilon}$-Парето для любого $\overline{\varepsilon}$, $\overline{\varepsilon} \geq \varepsilon$. В частности, Y_* является множеством ε-Парето для любого $\varepsilon \geq 0$. Следовательно при любом $\varepsilon \geq 0$ всегда существует хотя бы одно множество ε-Парето. Множество ε-Парето определяется однозначно только при $\varepsilon = 0$. В этом случае оно совпадает с Y_*. При $\varepsilon > 0$ вообще говоря может быть много множеств, удовлетворяющих введенному определению.

Обозначим через $\partial(\mathrm{NE}(Y))$ границу множества $\mathrm{NE}(Y)$. Рассмотрим множество

$$S_\varepsilon(Y) = Y \cap \cup_{y \in \partial(\mathrm{NE}(Y))} \mathrm{SW}(y + \varepsilon \cdot e_m),$$

которое будем называть ε-полосой множества Y.

Утверждение 2.1 *Для любого ε-Парето множества Y_ε справедливо включение*

$$Y_\varepsilon \subseteq S_\varepsilon(Y). \tag{2.7}$$

Доказательство. Пусть включение (2.7) не выполняется. Это означает, что существует ε-Парето множество Y_ε, содержащее точку y_ε, не принадлежащую ε-полосе. В этом случае точка $y' = y_\varepsilon - \varepsilon \cdot e_m$ является внутренней точкой множества Y. Пусть y_* — такая точка из Y_*, что $y_* \leq y'$. Так как y_* — граничная точка Y, а y' — внутренняя, то $y_* \neq y'$. Найдется такая точка $v \in Y_\varepsilon$, что

$$v - \varepsilon \cdot e_m \leq y_* \leq y' = y_\varepsilon - \varepsilon \cdot e_m.$$

Отсюда следует, что $v \leq y_\varepsilon$. При этом $v \neq y_\varepsilon$. Получили противоречие со вторым условием оптимальности $\mathcal{P}(Y_\varepsilon) = Y_\varepsilon$. $\square$

Для любой точки $y \in \mathbb{R}^m$ и множества $Z \subseteq \mathbb{R}^m$ определим расстояние от точки y до множества Z следующим образом: $\rho(y, Z) = \inf_{z \in Z} \|y - z\|$.

Расстояние от непустого множества $Y \subseteq \mathbb{R}^m$ до непустого множества $Z \subseteq \mathbb{R}^m$ определим как:

$$d(Y, Z) = \sup_{y \in Y} \rho(y, Z).$$

Расстояние $d(Y, Z)$ не является симметричным, так как величины $d(Y, Z)$ и $d(Z, Y)$, вообще говоря, отличаются. Введем симметричное *расстояние Хаусдорфа* между двумя непустыми подмножествами Y и Z пространства $\mathbb{R}^m$:

$$d_H(Y, Z) = \max\left(d(Y, Z), d(Z, Y)\right).$$

Легко доказать следующее утверждение, проиллюстрированное на Рис. 2.1.

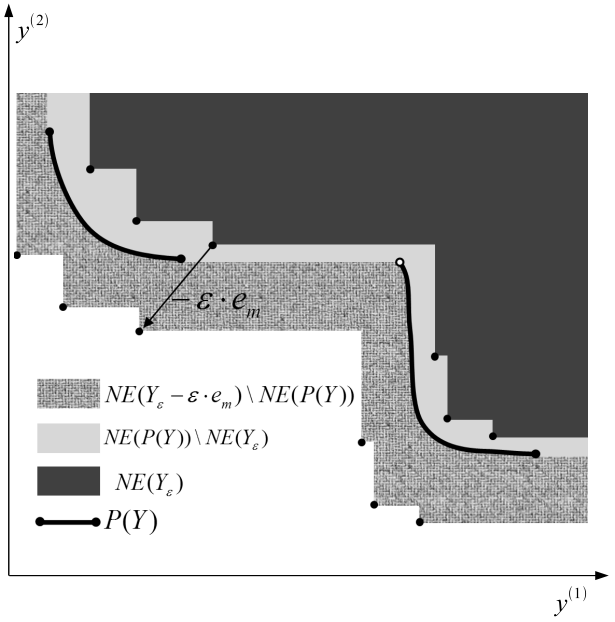

Рис. 2.1: Иллюстрация для теоремы 2.1

Теорема 2.1 *Справедливы следующие свойства, связывающие оболочки Эджворта-Парето множества ε-Парето и множества достижимых критериев:*

$$NE(Y_\varepsilon) \subseteq NE(Y_*) = NE(Y) \subseteq NE(Y_\varepsilon - \varepsilon \cdot e_m), \qquad (2.8)$$

$$d_H(NE(Y_\varepsilon), NE(Y)) \leq d_H(NE(Y_\varepsilon), NE(Y_\varepsilon - \varepsilon \cdot e_m)) \leq \varepsilon\sqrt{m}, \qquad (2.9)$$

$$d_H(NE(Y), NE(Y_\varepsilon - \varepsilon \cdot e_m)) \leq d_H(NE(Y_\varepsilon), NE(Y_\varepsilon - \varepsilon \cdot e_m)) \leq \varepsilon\sqrt{m}. \qquad (2.10)$$

Содержательно утверждение 2.1 означает, что при уменьшении ε, множества $\mathrm{NE}(Y_\varepsilon)$ и $\mathrm{NE}(Y_\varepsilon - \varepsilon \cdot e_m)$ стремятся к оболочке Эджворта-Парето $\mathrm{NE}(Y)$ множества достижимых критериев изнутри и снаружи множества Ω (Рис. 2.1). При этом множество Y_* заключено между множествами $\mathrm{NE}(Y_\varepsilon - \varepsilon \cdot e_m)$ и $\mathrm{NE}(Y_\varepsilon)$, т.е. $Y_* \in (\mathrm{NE}(Y_\varepsilon - \varepsilon \cdot e_m) \setminus \mathrm{NE}(Y_\varepsilon)) \cup Y_\varepsilon$.

Установим теперь связь между ε-Парето множеством и множеством Парето для задачи (2.1).

Лемма 2.3 *Пусть $y_* \in Y_*$. Тогда для любого $\delta > 0$ найдется такое $\varepsilon > 0$, что*

для ε-Парето множества Y_ε выполняется неравенство

$$\rho(y_*, Y_\varepsilon) \leq \delta.$$

Доказательство. Рассмотрим стремящуюся к нулю монотонно убывающую последовательность $\{\varepsilon_k\}$, $\varepsilon_k > 0$, $\lim_{k\to\infty} \varepsilon_k = 0$. Предположим, что утверждение теоремы неверно. Тогда для любого k существует ε_k-Парето множество Y_{ε_k}, такое что $\rho(y_*, Y_{\varepsilon_k}) > \delta$. По определению ε-Парето множества найдется точка $y_k \in Y_{\varepsilon_k}$ такая, что $y_k - \varepsilon_k \cdot e_m \leq y_*$. В силу компактности множества Y без ограничения общности можно считать, что последовательность $\{y_k\}$ сходится к точке $y' \in Y$. Так как $\lim_{k\to\infty} \varepsilon_k = 0$ и $y_k - \varepsilon_k \cdot e_m \leq y_*$, то $y' \leq y_*$. Так как $\rho(y_*, Y_{\varepsilon_k}) > \delta$ для каждого k, то $\rho(y_*, y') \geq \delta$ и, следовательно, $y_* \neq y'$. Но это противоречит тому, что $y \in Y_*$. $\square$

Приведем определение из [144]. Пусть $\varepsilon > 0$ — заданное положительное число. Множество $Z \subseteq \mathbb{R}^m$ называется *ε-сетью* для множества $Y \subseteq \mathbb{R}^m$, если для каждой точки $y \in Y$ найдется такая точка z из Z, что $\|y - z\| \leq \varepsilon$.

Теорема 2.2 *Для любого $\delta > 0$ найдется такое $\varepsilon > 0$, что любое ε-Парето множества Y_ε образует δ-сеть для множества Y_*.*

Доказательство. Очевидно, что множество Y_ε образует ε-сеть множества Y_* тогда и только тогда, когда

$$d(Y_*, Y_\varepsilon) \leq \delta.$$

Рассмотрим стремящуюся к нулю монотонную последовательность $\{\varepsilon_k\}$, $\varepsilon_k > 0$, $\lim_{k\to\infty} \varepsilon_k = 0$. Предположим, что утверждение теоремы неверно. Тогда для любого k существует ε_k-Парето множество Y_{ε_k}, такое что $d(Y_*, Y_{\varepsilon_k}) > \delta$. Это означает существование точки $y_k \in Y_*$, такой что $\rho(y_k, Y_{\varepsilon_k}) > \delta$. В силу компактности множества Y без ограничения общности можно считать, что последовательность $\{y_k\}$ сходится к некоторой точке $y \in Y$. Пусть K — такое натуральное число, что $\|y_i - y\| \leq \delta/3$ при $i \geq K$.

Согласно Лемме 2.3 найдется такое $\varepsilon > 0$, что для любого ε-Парето множеств Y_ε справедливо $\rho(y_K, Y_\varepsilon) \leq \delta/3$. Пусть N — такое целое число, не меньшее K, что $\varepsilon_N \leq \varepsilon$. Согласно предположению, $\rho(y_N, Y_{\varepsilon_N}) > \delta$. С другой стороны, поскольку Y_{ε_N} является ε-Парето множеством, то $\rho(y_K, Y_{\varepsilon_N}) \leq \delta/3$. Следовательно найдется точка $u \in Y_{\varepsilon_N}$, такая что $\|u - y_K\| \leq \delta/3$. Согласно свойствам нормы имеем: $\|u - y_N\| \leq \|u - y_K\| + \|y_K - y\| + \|y - y_N\| \leq \delta$. Пришли к противоречию с неравенством $\rho(y_N, Y_{\varepsilon_N}) > \delta$ и тем самым теорема доказана. □

Теорема 2.2 не устанавливает связь между ϵ и δ. Такую связь можно установить для точек, оптимальных по Джоффриону. Введем обозначение для множества индексов критериев $M = \{1, ..., m\}$. Согласно [65], точка $y_* \in Y$ называется *оптимальной по Джоффриону*, если существует такое положительное число $\theta(y_*)$, что для всех точек $y \in Y$ справедливо следующее: если $y^{(i)} < y_*^{(i)}$, то найдется $j \in M$, такое что $y^{(j)} > y_*^{(j)}$, причем

$$\frac{y_*^{(i)} - y^{(i)}}{y^{(j)} - y_*^{(j)}} \leq \theta(y_*). \tag{2.11}$$

Утверждение 2.2 *Если точка y_* оптимальна по Джоффриону, то для любого ε-Парето множества Y_ε при $\varepsilon > 0$ справедливо неравенство*

$$\rho(y_*, Y_\varepsilon) \leq \varepsilon\sqrt{m}\max(1, \theta(y_*)). \tag{2.12}$$

Доказательство. Пусть $\varepsilon > 0$ и Y_ε — ε-Парето множество. По определению ε-Парето множества найдется такая точка $y_\varepsilon \in Y_\varepsilon$, что $y_\varepsilon - \varepsilon \cdot e_m \leq y_*$. Определим два подмножества M_+ и M_- множества M:

$$M_+ = \{i \in M : y_*^{(i)} > y_\varepsilon^{(i)}\},$$
$$M_- = \{i \in M : y_*^{(i)} \leq y_\varepsilon^{(i)}\}.$$

Очевидно $M = M_+ \cup M_-$. Так как $y_\varepsilon - \varepsilon \cdot e_m \leq y_*$, то для любого $i \in M_-$ справедливо $0 \leq y_\varepsilon^{(i)} - y_*^{(i)} \leq \varepsilon$. Так как y оптимальна по Джоффриону, то для любого $i \in M_+$ найдется такое $j \in M_-$, что

$$\frac{y_*^{(i)} - y_\varepsilon^{(i)}}{y_\varepsilon^{(j)} - y_*^{(j)}} \leq \theta(y_*).$$

Следовательно при $i \in M_+$ выполняется

$$0 \le y_*^{(i)} - y_\varepsilon^{(i)} \le \theta(y_*)(y_\varepsilon^{(j)} - y_*^{(j)}) \le \theta(y_*)\varepsilon.$$

Таким образом для всех $i \in M$ справедливо $|y_\varepsilon^{(i)} - \omega_*^{(i)}| \le \delta$, где $\delta = \max(\varepsilon, \theta(y_*)\varepsilon)$. По определению эвклидовой метрики получаем утверждение теоремы.$\square$

Утверждение 2.2 позволяет оценить сверху точность аппроксимации, которую дает ε-Парето множества, используя величину $\theta(y)$. На рис 2.2 рассмотрена задача

$$\begin{aligned} f^{(1)}(x) &= x^{(1)} \\ f^{(2)}(x) &= (x^{(1)} - 1)^2 + x^{(2)}, \\ 0 &\le x^{(1)} \le 1, \\ 0 &\le x^{(2)} \le 1, \end{aligned} \tag{2.13}$$

Пунктирной линией изображено множество точек, оптимальных по Парето. Это множество в пространстве критериев задается участком кривой $f^{(2)} = (f^{(1)} - 1)^2$. Все точки этого множества являются оптимальными по Джоффриону, кроме точки $y_0 = (1, 0)$. При этом

$$\theta(y_*) = \max\left(2(1 - f^{(1)}), \frac{1}{2(1 - f^{(1)})}\right),$$

где $y_* = (f^{(1)}, (f^{(1)} - 1)^2)$.

Отдельные точки, обозначенные на рисунке, составляют ε-Парето множество, полученное с помощью метода неравномерных покрытий, при $\varepsilon = 0.05$. Видно, что точность аппроксимации в окрестности точки y_0, при приближении к которой $\theta(y_*)$ неограниченно растет, значительно ниже, чем в других точках. Это наблюдение соответствует оценке (2.12). Максимальная плотность точек ε-Парето множества наблюдается в центральной части графика, где значение $\theta(y_*)$ минимально.

Также можно получить количественную оценку точности дискретной аппроксимации на заданном отрезке, содержащем только джеффрионовские точки.

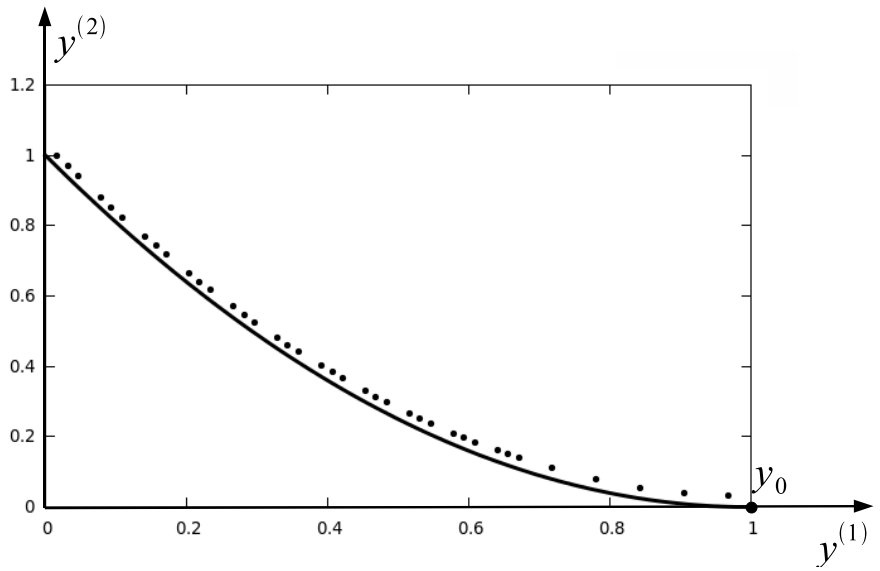

Рис. 2.2: Демонстрационный пример для утверждения 2.2

Например, на отрезке $f^{(1)} \in [0, 1/2]$ параметр $\theta(y_*)$ не превосходит 2. Следовательно расстояние от любой точки множества Парето на этом участке до ближайшей точки ε-Парето множества не превосходит $2\sqrt{2}\varepsilon$.

Нахождение множества ε-Парето

Напомним основную идею метода неравномерных покрытий для задач с одним критерием. Для непрерывной функции $f : \mathbb{R}^n \to \mathbb{R}$ ставится задача отыскания глобального минимума на компактном *допустимом множестве* $X \subseteq \mathbb{R}^n$:

$$f_* = glob \min_{x \in X} f(x) = f(x_*), \tag{2.14}$$

где x_* — любая точка, в которой достигается глобальный минимум, равный f_*. Для этой задачи определим множество *глобальных решений* X_* и множество *ε-оптимальных решений* X_ε:

$$X_* = \{x \in X : f(x) = f_*\}, X_\varepsilon = \{x_\varepsilon \in X : f(x_\varepsilon) \leq f_* + \varepsilon\}, \varepsilon > 0. \tag{2.15}$$

Предполагаем, что множество X_* не пусто. Требуется найти хотя бы одну точку из множества X_ε.

Для множества $Z \subseteq \mathbb{R}^n$, функции $f(x) : \mathbb{R}^n \to \mathbb{R}$ и числа $\lambda \in \mathbb{R}$ определим *Лебеговское множество*

$$\mathcal{L}(f(\cdot), Z, \lambda) = \{x \in Z : f(x) \geq \lambda\}. \tag{2.16}$$

Используя понятие лебеговского множества, можно определить необходимые и достаточные условия глобальной оптимальности для любой точки $x_* \in X$ следующим образом: $x_* \in X_* \Leftrightarrow \mathcal{L}(f(\cdot), X, f(x_*)) = X$.

Аналогично записывается критерий глобальной ε-оптимальности. Пусть $x_\varepsilon \in X$, тогда

$$x_\varepsilon \in X_\varepsilon \Leftrightarrow \mathcal{L}(f(\cdot), X, f(x_\varepsilon) - \varepsilon) = X.$$

Перейдем теперь к рассмотрению случая m-критериев при $m > 1$. Для произвольных множеств $\Lambda \subseteq \Omega$, $Z \subseteq \mathbb{R}^n$ и вектор-функции $F(\cdot) : \mathbb{R}^n \to \mathbb{R}^m$ определим *Лебеговское множество*

$$\mathcal{L}(F(\cdot), Z, \Lambda) = \{x \in Z : F(x) \in \mathrm{NE}(\Lambda)\}. \tag{2.17}$$

В случае $m = 1$ это определение совпадает с (2.16).

Введенное понятие можно использовать для альтернативного определения Парето множества. Пусть $\Theta \subseteq \Omega$ и $\mathcal{P}(\Theta) = \Theta$ тогда

$$\Theta = \Omega_* \Leftrightarrow L(F(\cdot), X, \Theta) = X. \tag{2.18}$$

Критерий глобальной ε-оптимальности имеет следующий вид. Пусть $\Theta \subseteq \Omega$ и $\mathcal{P}(\Theta) = \Theta$ тогда

$$\Theta \text{ есть } \varepsilon - \text{Парето множество} \Leftrightarrow L(F(\cdot), X, \Theta - \varepsilon \cdot e_m) = X. \tag{2.19}$$

Рассмотрим совокупность подмножеств допустимого множества $X_1, \ldots, X_k$, $X_i \subseteq X$ и совокупность конечных подмножеств множества достижимых критериев $\Lambda_1, \ldots, \Lambda_k$, $\Lambda_i \subseteq \Omega$. Пусть $\mu_i(\cdot)$ — миноранта для вектор-функции $F(\cdot)$ на

множестве X_i т.е. $\mu_i(x) \leq f(x)$ для каждого $x \in X_i$. Пусть задана совокупность подмножеств $\mathcal{L}_1, \ldots, \mathcal{L}_k$ множества X, удовлетворяющих соотношениям:

$$\mathcal{L}_i \subseteq \mathcal{L}(\mu_i(\cdot), X_i, \Lambda_i - \varepsilon \cdot e_m), i = 1, \ldots, k, \tag{2.20}$$

где $\Lambda_i - \varepsilon \cdot e_m = \{x : x = \lambda - \varepsilon \cdot e_m \text{ for some } \lambda \in \Lambda_i\}$.

Будем говорить, что совокупность множеств $\{\mathcal{L}_i\}$ *покрывает множество* X, если

$$X = \cup_{i=1}^{k} \mathcal{L}_i. \tag{2.21}$$

Множества $\mathcal{L}_i$ будем называть *покрывающими*.

Теорема 2.3 *Если выполнено условие покрытия (2.21), то множество* $Y_k = \mathcal{P}\left(\cup_{i=1}^{k} \Lambda_i\right)$ *является* ε*-Парето множеством для задачи (2.1).*

Доказательство. Рассмотрим произвольную точку $y_* \in Y_*$, $y_* = f(x_*)$, $x_* \in X$. Если выполнено условие покрытия, то $x_* \in \mathcal{L}(\mu_i(\cdot), X_i, \Lambda_i - \varepsilon \cdot e_m)$ для некоторого $i, 1 \leq i \leq k$. Следовательно найдется $\lambda_i \in \Lambda_i$, такое что $\lambda_i - \varepsilon \cdot e_m \leq \mu_i(x_*) \leq F(x_*)$. Так как $Y_k = \mathcal{P}\left(\cup_{i=1}^{k} \Lambda_i\right)$, то найдется $y_\varepsilon \in Y_k$, такое что $y_\varepsilon \leq \lambda_i$. Отсюда следует, что $y_\varepsilon - \varepsilon \cdot e_m \leq \lambda_i - \varepsilon \cdot e_m \leq F(x_*) = y_*$. В силу произвольности выбора $y_* \in Y_*$ установлена справедливость первого условия оптимальности. Второе условие оптимальности следует из свойства (2.2). □

В процессе своей работы метод неравномерных покрытий строит множество Y_k и совокупность покрывающих множеств $\{\mathcal{L}_i\}$, удовлетворяющих условиям теоремы 2.3. Рассмотрим подробно способы построения этих совокупностей.

Для построения множества Y_k в процессе работы алгоритма поддерживается список A точек допустимого множества X. Образ $F(A)$ множества A при отображении F содержит конечный набор попарно несравнимых точек, который после завершения работы алгоритма будет множеством Y_k. Значения критериев хранятся вместе со значениями параметров. Это позволяет избежать повторного вычисления значений критериев каждый раз при проведении сравнения.

Список A строится последовательно с использованием процедуры **Update**, реализующей добавление очередной точки ко множеству A.

Процедура Update (A, x)

Параметры:

A — текущий список точек;

x — новая точка;

1. Для каждой точки a из A выполнить следующие действия:

   если $F(a) \leq F(x)$, то завершить процедуру;

   если $F(x) \leq F(a)$, то удалить a из A;

2. Добавить x в A.

Очевидно, образ $F(A)$ множества A, построенного при помощи данной процедуры, удовлетворяет условию (2.5). Последовательность добавляемых точек может генерироваться различными способами, более подробно рассматриваемыми при описании всего алгоритма.

Улучшить точность аппроксимации позволяют методы локальной оптимизации, т.е. методы, позволяющие по точке $x \in X$ получить точку $x' \in X$, такую что $F(x') \leq F(x)$. Эта процедура применяется к точке x до добавления к списку A либо после него.

Нахождение множества $\mathcal{L}_i$ непосредственно из определения (2.20) является достаточно сложной задачей, которую можно упростить, используя следующее очевидное соотношение:

$$\mathcal{L}(\mu_i(\cdot), X_i, \Lambda_i - \varepsilon \cdot e_m) = \cup_{\lambda \in \Lambda_i} \mathcal{L}(\mu_i(\cdot), X_i, \lambda - \varepsilon \cdot e_m), \tag{2.22}$$

где $\mathcal{L}(\mu_i(\cdot), X_i, \lambda - \varepsilon \cdot e_m) = \{x \in X_i : \mu_i(x) \geq \lambda - \varepsilon \cdot e_m\}$. Из соотношения (2.22) следует, что множество $\mathcal{L}_i$ можно искать в виде объединения:

$$\mathcal{L}_i = \cup_{\lambda \in \Lambda_i} \mathcal{L}_i^{\lambda}, \tag{2.23}$$

где $\mathcal{L}_i^\lambda \subseteq \mathcal{L}(\mu_i(\cdot), X_i, \lambda - \varepsilon \cdot e_m)$. Множества $\mathcal{L}_i^\lambda$ находятся проще чем $\mathcal{L}_i$. В дальнейшем будут использоваться следующие две миноранты.

Если функция $f^{(j)}(x)$ удовлетворяет условию Липшица с константой l_i^j на множестве X_i, то согласно [46] следующая функция будет минорантой для функции $f(x)$:

$$\mu_i^{(j)}(x) = f^{(j)}(c_i) - l_i^j \|x - c_i\|, \tag{2.24}$$

где $c_i \in X_i$.

Если градиент $f_x^{(j)}(\cdot)$ функции $f^{(j)}(\cdot)$ удовлетворяет условию Липшица с константой l_i^j, то применяется следующая миноранта [48]:

$$\mu_i^{(j)}(x) = f^{(j)}(c_i) + \langle f_x^{(j)}(c_i), x - c_i \rangle - \frac{l_i^j}{2}\|x - c_i\|^2. \tag{2.25}$$

Значения минорант (2.24, 2.25) в точке c_i совпадают со значением минорируемого критерия $f^{(j)}(\cdot)$ в этой точке. Поэтому эти миноранты называют *опорными*, а точку c_i — *опорной точкой*.

Укажем способы нахождения множеств $\mathcal{L}_i^\lambda$ для разных случаев.

1. Пусть $\mu_i(\cdot)$ — произвольная миноранта для вектор-функции $F(\cdot)$ на множестве X_i. Пусть также для каждого $j \in M$ известен способ определения минимума $\alpha_i^{(j)}$ функций $\mu_i^{(j)}(\cdot)$, на множестве X_i, $\alpha_i = (\alpha_i^{(1)}, \ldots, \alpha_i^{(m)})$. Положим

$$\mathcal{L}_i^\lambda = \begin{cases} X_i, \text{ если } \alpha_i \geq \lambda - \varepsilon \cdot e_m, \\ \emptyset \text{ в противном случае.} \end{cases}$$

Тогда согласно (2.22)

$$\mathcal{L}_i = \begin{cases} X_i, \text{ если найдется } \lambda \in \Lambda_i \text{ такое что } \alpha_i \geq \lambda - \varepsilon \cdot e_m, \\ \emptyset \text{ в противном случае .} \end{cases}$$

Этот способ нахождения $\mathcal{L}_i$ удобно применять для минорант (2.24) и (2.25), т.к. известны аналитические формулы для точек, в которых эти минимумы

достигаются на n-мерном параллелепипеде [48].

2. Пусть $\mu_i(\cdot)$ — произвольная миноранта для вектор-функции $F(\cdot)$ на множестве X_i. Пусть также для каждого индекса j, $1 \le j \le m$ известен способ определения множества $K_i^j \subseteq \mathcal{L}(\mu_i^{(j)}(\cdot), X_i, \lambda^{(j)} - \varepsilon)$. Тогда положим

$$\mathcal{L}_i^\lambda = \cap_{j=1}^m K_i^j. \tag{2.26}$$

Для миноранты (2.24) множество $\mathcal{L}(\mu_i^{(j)}(\cdot), X_i, \lambda^{(j)} - \varepsilon)$ является пересечением шара радиуса $\rho_i^j(\lambda) = (f^{(j)}(c_i) - \lambda^{(j)} + \varepsilon)/l_i^j$ с центром в точке c_i и множества X_i. В работе [46] рассмотрен случай, когда миноранты вида (2.24) имеют одну и ту же опорную точку c_i для всех критериев. При этом множество K_i^j полагается равным Лебеговскому множеству, т.е. определяется как пересечение шара $B(c_i, \rho_i^j(\lambda))$ и множества X_i. Тогда:

$$\mathcal{L}_i^\lambda = B(c_i, \rho_i(\lambda)) \cap X_i,$$

где $\rho_i(\lambda) = \min_{1 \le j \le M} \rho_i^j(\lambda)$, а

$$\mathcal{L}_i = B(c_i, \rho_i) \cap X_i, \tag{2.27}$$

где $\rho_i = \max_{\lambda \in \Lambda_i} \rho_i(\lambda)$.

Таким образом

$$\rho_i = \max_{\lambda \in \Lambda_i} \min_{1 \le j \le M} (f^{(j)}(c_i) - \lambda^{(j)} + \varepsilon)/l_i^j.$$

Данная формула с точностью до обозначений совпадает с формулой (3) из работы [120].

Для миноранты (2.25) множество $\mathcal{L}(\mu_i^{(j)}(\cdot), X_i, \lambda^{(j)} - \varepsilon)$ является пересечением шара радиуса

$$\rho_i^j(\lambda) = \sqrt{\frac{2}{l_i^j}\left(\frac{1}{2l_i^j}\|f_x^{(j)}(c_i)\|^2 + f(c_i) - \lambda^{(j)} + \varepsilon\right)}$$

с центром в точке $z_i = c_i + f_x^{(j)}/l_i^j$ и множества X_i.

В этом случае подход, примененный для миноранты (2.24), не работает, т.к. шары имеют различные центры и их пересечение является сложной фигурой, с трудом поддающейся алгоритмической обработке. Рассмотрим два возможных варианта решения этой задачи.

Первый вариант основан на том, что множество $\mathcal{L}(\mu_i^{(j)}(\cdot), X_i, \lambda^{(j)} - \varepsilon)$ содержит шар радиуса

$$\rho_i^j(\lambda) = \left(\sqrt{\|f_x^{(j)}(c_i)\|^2 + 2l_i^j(f^{(j)}(c_i) - \lambda^{(j)} - \varepsilon)} - \|f_x^{(j)}(c_i)\| \right) / l_i^j \qquad (2.28)$$

с центром в точке c_i. Полагая множество K_i^j равным пересечению этого шара со множеством X_i и проводя рассуждения, аналогичные проведенным для миноранты (2.24), получаем что множество $\mathcal{L}_i$ может быть вычислено по формуле (2.27), где $\rho_i^j(\lambda)$ вычисляются по формуле (2.28).

Во втором варианте в качестве множеств K_i^j берутся n-мерные параллелепипеды, принадлежащие множеству $\mathcal{L}(\mu_i^{(j)}(\cdot), X_i, \lambda^{(j)} - \varepsilon)$. Способ нахождения такого параллелепипеда указан в работе [121]. Множество $\mathcal{L}_i^\lambda$ будет пересечением параллелепипедов, а следовательно, тоже параллелепипедом. Множество $\mathcal{L}_i$ представляет собой объединение параллелепипедов, которое удобно использовать в алгоритмах.

Рассмотрим одну из возможных реализаций метода неравномерных покрытий.

Алгоритм Cover

1. Инициализировать список подмножеств $S = \{X\}$ и список точек $A = \emptyset$.
2. Взять из списка S некоторое множество, которое обозначим через X_i.
3. Выбрать точку $c_i \in X_i$ и обновить список точек A: Update(A, c_i).
4. Определить множество $\mathcal{L}_i$ и его дополнение $X_i' : X_i' = X_i \setminus \mathcal{L}_i$.

5. Если $X_i' \neq \emptyset$, то разбить X_i' на p подмножеств $\mathcal{Y}_i = \{Y_1^i, \ldots, Y_p^i\}$ и добавить их в список S.

6. Удалить X_i из списка S.

7. Если список S пуст, то завершить работу алгоритма, в противном случае перейти к шагу 2.

Применив теорему 2.3, легко показать, что после завершения работы алгоритма, множество A является ε-оптимальным решением, а множество $f(A)$ — ε-Парето множеством для задачи (2.1).

Для проведения экспериментов был реализован вариант приведенного алгоритма, характеризующий следующими особенностями:

- предполагается, что допустимое множество — параллелепипед и все множества X_i — также параллелепипеды;
- параллелепипед X_i' всегда разбивается на два равных параллелепипеда вдоль ребра максимальной длины;
- в качестве точки c_i всегда берется центр параллелепипеда X_i.

При сравнении алгоритмов поиска минимума скалярной функции лучшим считается алгоритм, нашедший допустимое решение с наименьшим значением целевой функции. Сравнение различных алгоритмов многокритериальной оптимизации является более сложной задачей, так как здесь нет возможности выделить единый параметр, по которому проводится сравнение. Подробный обзор различных методик сравнения приближенных решений многокритериальных задач, можно найти в работе [71]. В дальнейшем используем два количественных показателя из числа предложенных в ней.

Первый показатель носит название Hyper Volume (HV) и измеряет общий объем области, образованной объединением перекрывающихся параллелепипе-

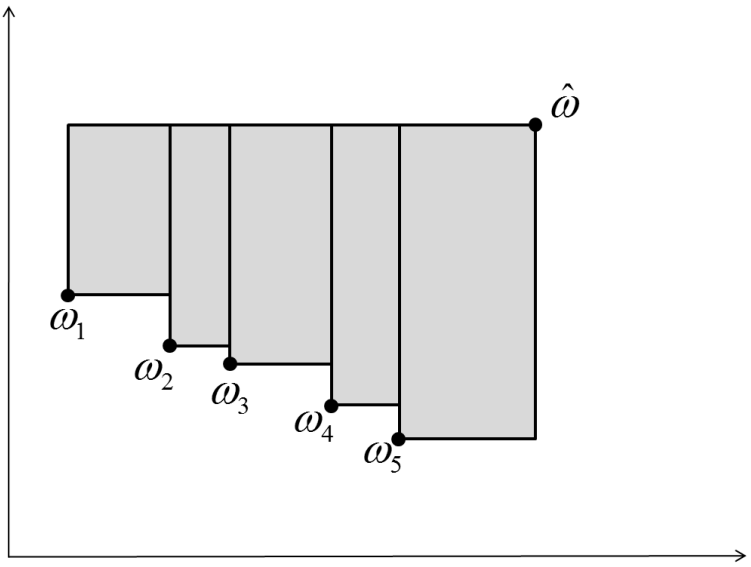

Рис. 2.3: Показатель Hyper Volume (HV)

дов, расположенных между некоторой заданной (*референсной*) точкой и точками дискретной аппроксимации (Рис. 2.3). В качестве референсной обычно берется точка r такая, что $r^{(i)} \geq \max_{x \in X} f^{(i)}(x)$, $i = 1, \ldots, m$. Заметим, что нахождение референсной точки не всегда является тривиальной задачей. Если A и A' — две аппроксимации множества Парето, причем $A' \leq A$, то для одной и той же референсной точки значение показателя HV для A' будет больше, чем у A. Другими словами, большие значения показателя HV обычно соответствуют более точным аппроксимациям.

Помимо близости к истинному множеству Парето качество полученной аппроксимации также характеризуется равномерностью распределения точек в ней. При одинаковом числе точек предпочтительнее аппроксимация, в которой точки распределены более равномерно. Следующий показатель (2.29) позволяет грубо оценить равномерность этого распределения. Для аппроксимации A, содержащей k точек, определим

$$\mathrm{UD}(A) = \sqrt{\sum_{i=1}^{k} (d_i - d)^2}, \;\; d = \left(\sum_{i=1}^{k} d_i \right) / k, \tag{2.29}$$

где d_i — минимальное расстояние от точки с номером i до остальных точек

из A. Меньшие значения данного показателя как правило соответствуют более равномерным распределениям точек. В частности, если все минимальные расстояния одинаковы, то показатель принимает нулевое значение. Заметим, что показатель (2.29) не содержателен для задач, в которых множество Парето несвязно.

В экспериментах проводилось сравнение трех алгоритмов:

1. NUC: метод неравномерных покрытий;
2. MC: метод "Монте-Карло" — стохастический алгоритм , при котором случайные точки равномерно распределены в допустимой области, а дискретная аппроксимация строится с помощью процедуры **Update**;
3. SEMO: генетический алгоритм SEMO из библиотеки PISA [145].

При проведении сравнения измерялись значения следующих параметров:

1. nit — число вызовов функций, вычисляющих критерии;
2. ε — точность, данный параметр имеет смысл только для метода неравномерных покрытий и определяет значение ε для получаемого множества ε-Парето;
3. ngen — число поколений, данный параметр имеет смысл только для генетического алгоритма;
4. np — число точек в дискретной аппроксимации;
5. hv — значение показателя Hyper Volume;
6. ud — значение показателя равномерности распределения точек в аппроксимации.

Для проведения сравнения рассмотрим два примера.

Таблица 2.1: Сравнительные результаты работы различных алгоритмов для примера 1

Метод	nit	ε	ngen	np	hv
SEMO	500	-	500	221	3.27
MC	500	-	-	22	3.38
NUC	490	0.07	-	36	3.42

Пример 1. Рассмотрим задачу (2.1) где критерии задаются формулами

$$f^{(1)}(x^{(1)}, x^{(2)}) = x^{(1)},\ f^{(2)}(x^{(1)}, x^{(2)}) = \min(|x^{(1)} - 1|, 1.5 - x^{(1)}) + x^{(2)} + 1,$$

а допустимая область определяется неравенствами $0 \leq x^{(1)} \leq 2, 0 \leq x^{(2)} \leq 2$.

Граница Парето в данном примере состоит из двух отрезков. Первый соединяет точки $(0, 2)$ и $(1, 1)$, а второй — точки $(1.5, 1)$ и $(2, 0.5)$.

Параметры трех алгоритмов выбирались таким образом, чтобы обеспечить примерно одинаковое число вычислений критериев, близкое к 500. В таблице 2.1 приведены результаты экспериментов для различных алгоритмов. Прочерк в таблице означает, что параметр не имеет смысла для данного метода. Показатель равномерности не вычислялся, поскольку множество Парето несвязно.

На Рис. 2.4 показаны дискретные аппроксимации множества Парето, полученные различными методами SEMO, MC, NUC для примера 1. Визуально видно, что качество аппроксимации метода неравномерных покрытий существенно выше чем у двух других методов.

Пример 2. В данном примере решается задача (2.1), в которой критерии задаются формулами:

$$f^{(1)}(x) = (x^{(1)} - 1) \cdot (x^{(2)})^2 + 1, f^{(2)}(x) = x^{(2)}.$$

Допустимая область определяется неравенствами $0 \leq x^{(1)} \leq 1, 0 \leq x^{(2)} \leq 1$.

Несложно убедиться, что решением данной задачи в пространстве критериев будет участок параболы $y^{(1)} = 1 - (y^{(2)})^2$ при $y^{(1)}, y^{(2)} \in [0, 1]$. Задача не выпукла. Также как и в предыдущем примере параметры трех алгоритмов выбирались таким образом, чтобы обеспечить примерно одинаковое число вычислений

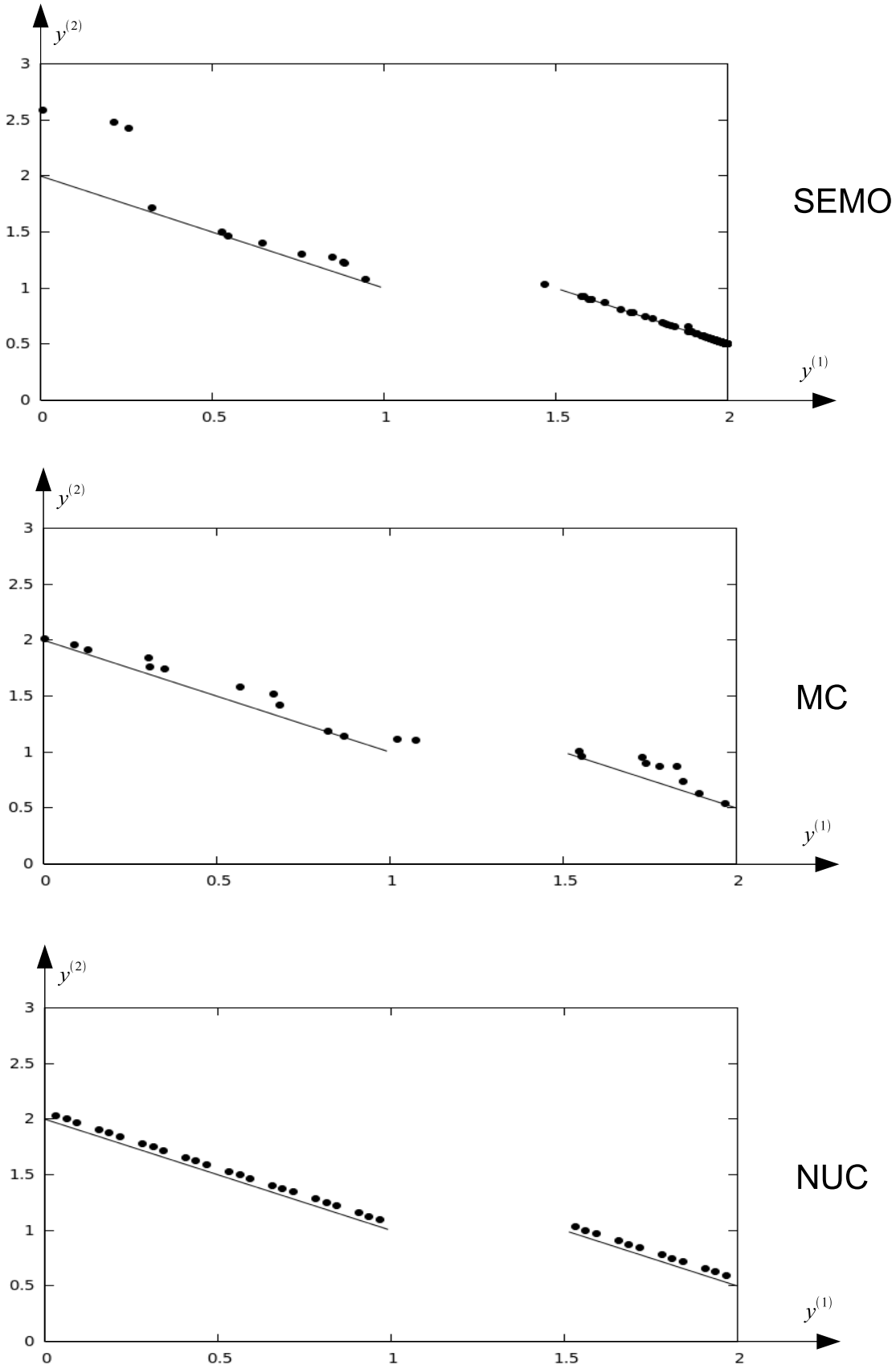

Рис. 2.4: Аппроксимации множества Парето, полученные различными методами для примера 1

Таблица 2.2: Сравнительные результаты работы различных алгоритмов для примера 2

Метод	nit	ε	ngen	np	hv	ud
SEMO	500	-	500	104	0.312	1.116
MC	500	-	-	67	0.300	1.277
NUC	515	0.0675	-	29	0.306	0.210

критериев, близкое к 500. В таблице 2.2 приведены результаты экспериментов для различных алгоритмов. На Рис. 2.5 показаны дискретные аппроксимации множества Парето, полученные различными методами SEMO, MC, NUC для примера 2.

Анализ данных таблицы 2.2 и графиков на Рис. 2.5 показывает, что по показателю Hyper Volume метод неравномерных покрытий немного превзошел стохастический алгоритм и несколько уступил генетическому алгоритму. При этом, с помощью генетического алгоритма построена аппроксимация, содержащая в три с лишним раза больше точек по сравнению с методом неравномерных покрытий. Показатель ud, характеризующий равномерность распределения точек и визуальный анализ графиков показывают что точки, построенные методом неравномерных покрытий дают существенно более равномерное распределение.

Следует отметить, что в отличие от генетического и стохастического алгоритмов, метод неравномерных покрытий гарантирует ε-оптимальность для заданного ε.

2.3 Решение задач многокритериальной оптимизации при наличии функциональных ограничений

В данном разделе рассмотрим задачу многокритериальной оптимизации (2.1) в которой допустимое множество задано с помощью функциональных ограничений:

$$X = \{x \in \mathbb{R}^n : G(x) \leq 0_k\}, \tag{2.30}$$

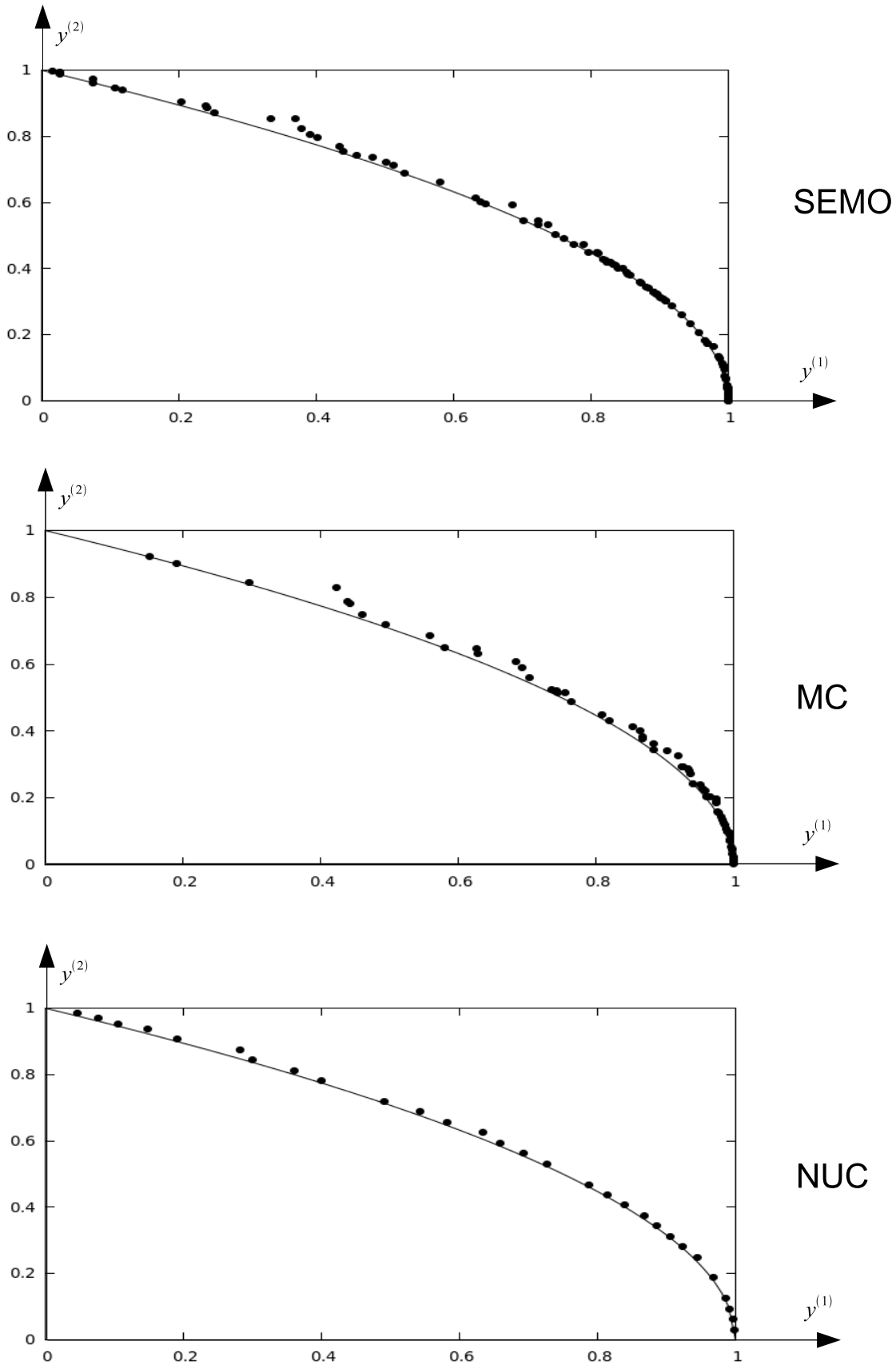

Рис. 2.5: Аппроксимации множества Парето, полученные различными методами для примера 2

где $G(\cdot) = (g^{(1)}(\cdot), \ldots, g^{(k)}(\cdot))$ — набор функций ограничений, а 0_k — k-мерный вектор с нулевыми компонентами. Условия (2.30) можно переписать в эквивалентном виде

$$X = \{x \in \mathbb{R}^n : \phi(x) \leq 0\}, \tag{2.31}$$

где $\phi(x) = \max_{i=1,\ldots,k} g^{(i)}(x)$.

Всюду далее вектор-функции $F(\cdot), G(\cdot)$ предполагаются непрерывными, а множество X предполагается непустым и ограниченным: $X \subseteq B$, где $B = [a, b] = \{x \in \mathbb{R}^n : a \leq x \leq b\}$ — n-мерный параллелепипед. При сделанных предположениях множества X и Y будут непустыми компактами.

Решение задачи (2.1) состоит в нахождении множества Парето и его прообраза при отображении F. Нахождение точного решения возможно только в простейших случаях, поэтому обычно ищется приближенное решение. В предыдущем разделе был рассмотрен случай простых ограничений параллелепипедного типа. В случае функциональных ограничений предложенный метод не сработает, так как в отличие от параллелепипеда, произвольное множество сложно декомпозировать. Определим понятие приближенного решения в случае задачи с ограничениями.

Для $\delta \in \mathbb{R}$ определим $X^\delta = \{x \in \mathbb{R}^k : \phi(x) \leq \delta\}$, положим $Y^\delta = F(X^\delta)$. Заметим, что множества X^δ и Y^δ не сужаются с увеличением δ. Пусть $\underline{\delta} = \inf \delta \in \mathbb{R} : X^\delta \neq \emptyset$, $\overline{\delta} = \sup \delta \in \mathbb{R} : X^\delta \subseteq B$.

Пусть даны действительные числа ε, δ, такие что $\varepsilon \geq 0$, $0 \leq \delta \leq \overline{\delta}$. Множество $Q \subseteq Y^\delta$ будем называть множеством ε, δ-Парето , если выполнены условия:

1. для любой точки $y_* \in \mathcal{P}(Y)$ существует такая точка $q \in Q$, что

$$q - \varepsilon \cdot 1_m \leq y_*, \tag{2.32}$$

2.

$$\mathcal{P}(Q) = Q. \tag{2.33}$$

Здесь через 1_m обозначен m-мерный вектор, все компоненты которого единицы.

Рассмотрим совокупность $X_1, \ldots, X_k$, $X_i \subseteq B, i = 1, \ldots, k$ параллелепипеда B и совокупность $\Lambda_1, \ldots, \Lambda_k$ конечных подмножеств множества Y_δ , $\Lambda_i \subseteq Y^\delta, i = 1, \ldots, k$. Пусть $\mu_i(\cdot)$ — миноранта для вектор-функции $F(\cdot)$ на множестве X_i т.е. $\mu_i(x) \leq F(x)$ для каждого $x \in X_i$. Пусть $\nu_i(\cdot)$ — миноранта для функции $\phi(\cdot)$ на множестве X_i. Пусть задана совокупность подмножеств $\mathcal{L}_1, \ldots, \mathcal{L}_k$ множества B, удовлетворяющих соотношениям:

$$\mathcal{L}_i \subseteq \mathcal{L}(\mu_i(\cdot), X_i, \Lambda_i - \varepsilon \cdot e_m) \cup \mathcal{L}'(\nu_i(\cdot), X_i, 0), i = 1, \ldots, k,$$

где $\Lambda_i - \varepsilon \cdot 1_m = \{x : x = \lambda - \varepsilon \cdot 1_m \text{ for some } \lambda \in \Lambda_i\}$.

Будем говорить, что совокупность множеств $\{\mathcal{L}_i\}$ *покрывает множество* B, если

$$B = \cup_{i=1}^k \mathcal{L}_i. \tag{2.34}$$

Множества $\mathcal{L}_i$ будем называть *покрывающими*.

Теорема 2.4 *Если выполнено условие покрытия (2.34), то множество* $Y_k = \mathcal{P}\left(\cup_{i=1}^k \Lambda_i\right)$ *является множеством* ε, δ*-Парето для задачи (2.1).*

Доказательство. Рассмотрим произвольную точку $y_* \in \mathcal{P}(Y)$, $y_* = F(x_*)$, $x_* \in X$. Если выполнено условие покрытия, то $x_* \in \mathcal{L}_i$ для некоторого $i, 1 \leq i \leq k$. Следовательно, $x_* \in \mathcal{L}(\mu_i(\cdot), X_i, \Lambda_i - \varepsilon \cdot e_m)$ или $x_* \in \mathcal{L}'(\nu_i(\cdot), X_i, 0)$.

Заметим, что если $x_* \in \mathcal{L}'(\nu_i(\cdot), X_i, 0)$, то $\phi(x_*) \geq \nu_i(x_*) > 0$, что противоречит тому, что $x_* \in X$.

Пусть $x_* \in \mathcal{L}(\mu_i(\cdot), X_i, \Lambda_i - \varepsilon \cdot 1_m)$. Тогда найдется $\lambda_i \in \Lambda_i$, такое что $\lambda_i - \varepsilon \cdot 1_m \leq \mu_i(x_*) \leq F(x_*)$. Так как $Y_k = \mathcal{P}\left(\cup_{i=1}^k \Lambda_i\right)$, то найдется $y \in Y_k$, такое что $y \leq \lambda_i$. Отсюда следует, что $y - \varepsilon \cdot 1_m \leq \lambda_i - \varepsilon \cdot 1_m \leq F(x_*) = y_*$.

В силу произвольности выбора $y_* \in \mathcal{P}(Y)$ установлена справедливость условия (2.32). Условие (2.33) следует из свойства (2.2). $\square$

Рассмотрим теперь алгоритм для построения множества ε, δ-Парето. Алгоритм основан на разбиении параллелепипеда B на параллелепипеды B_i. На каждом параллелепипеде определяются *миноранты* $\mu_i^{(s)}(\cdot), \nu_i^{(t)}(\cdot)$ для функций $f^{(s)}(\cdot)$ и $g^{(t)}(\cdot)$:

$$f^{(s)}(x) \geq \mu_i^{(s)}(x),\ x \in B_i,\ s = 1, \ldots, m,$$
$$g^{(t)}(x) \geq \nu_i^{(t)}(x),\ x \in B_i,\ t = 1, \ldots, k.$$

Основное требование к используемым минорантам состоит в том, чтобы их минимум на параллелепипеде находился аналитически. К этому классу относятся, например, Липшицевы [46] и интервальные миноранты [95].

В процессе работы алгоритма поддерживается список точек из множества X^δ такой, что на каждом шаге его образ при отображении F содержит конечный набор попарно несравнимых точек, который после завершения работы алгоритма будет искомым множеством ε, δ-Парето. Значения критериев хранятся вместе со значениями параметров. Это позволяет избежать повторного вычисления значений критериев каждый раз при проведении сравнений. Список строится последовательно с использованием процедуры **Update**, реализующей добавление очередной точки к списку.

Процедура Update (A, x, A')

Параметры:

A — текущий список точек;

x — новая точка;

A' — новый список;

1. ЦИКЛ по точкам $a \in A$:

   если $F(a) \leq F(x)$, то положить $A' = A$, перейти к шагу 3;

   если $F(x) \leq F(a)$, то удалить a из A;

КОНЕЦ ЦИКЛА

2. Положить $A' = A \cup x$;

3. КОНЕЦ ПРОЦЕДУРЫ

Перед началом работы алгоритма создается список S параллелепипедов, содержащий на первом шаге только один элемент — исходный параллелепипед B. Список точек A_0 пуст.

Алгоритм Cover

1. Если $S = \emptyset$, то завершить алгоритм.

2. Выбрать и удалить из списка параллелепипед $B_i = [a_i, b_i]$.

3. Найти точку $c_i = (a_i + b_i)/2$. Если $c_i \in X^\delta$, то выполнить процедуру Update(A_{i-1},c_i, A_i).

4. Для $s = 1, \ldots, m$ определить $\alpha_i^{(t)} = \min_{x \in B_i} \mu_i^{(s)}(x)$. Если для некоторой точки $x \in A_i$ выполнено $\alpha_i \geq x - \varepsilon \cdot 1_m$, то перейти к шагу 1.

5. Для $t = 1, \ldots, k$ определить $\beta_i^{(t)} = \min_{x \in B_i} \nu_i^{(t)}(x)$.
   Если $\max_{t=1,\ldots,k} \beta_i^{(t)} > 0$, то перейти к шагу 1.

6. Разбить параллелепипед B_i на два равных параллелепипеда B_i' и B_i'' вдоль ребра максимальной длины и добавить их в список S.

Теорема 2.5 *Если алгоритм* **Cover** *завершился за K шагов, то множество $F(A_K)$ будет ε, δ-Парето множеством.*

Доказательство. Прежде всего заметим, что множества A_i формируются только из точек множества X^δ и таким образом, что условие (2.5) всегда выполнено. Покажем справедливость условия (2.4). Поскольку алгоритм **Cover** завершился за K шагов, то $B = \cup_{i=1}^K B_i$. Рассмотрим произвольную точку $y_* \in \mathcal{P}(Y)$ и ее прообраз при отображении F — точку $x_* \in X$. Пусть $x_* \in B_i$ для неко-

торого $i, 1 \leq i \leq K$. Так как алгоритм завершился, то параллелепипед B_i был исключен из дальнейшего рассмотрения либо на шаге 4, либо на шаге 5.

Покажем, что параллелепипед B_i не мог быть исключен из дальнейшего рассмотрения на шаге 5. Действительно, в этом случае

$$\phi(x_*) = \max_{t=1,\ldots,k} g^{(t)}(x_*) \geq \max_{t=1,\ldots,k} \min_{x \in B_i} g^{(t)}(x) \geq$$
$$\max_{t=1,\ldots,k} \min_{x \in B_i} \nu_i^{(t)}(x) = \max_{t=1,\ldots,k} \beta_i^{(t)} > 0,$$

что противоречит тому, что $x_* \in X$.

Значит, параллелепипед B_i был исключен из дальнейшего рассмотрения на шаге 4. Это означает, что найдется точка x из списка A_i, такая что $\alpha_i \geq F(x) - \varepsilon \cdot 1_m$. Несложно заметить, что способ добавление точек к списку A_i обеспечивает выполнение следующего свойства: $NE(F(A_1)) \subseteq NE(F(A_2)) \subseteq \cdots \subseteq NE(F(A_K))$. Следовательно, найдется такая точка $x' \in A_K$, что $\alpha_i \geq F(x') - \varepsilon \cdot 1_m$. Следовательно

$$y_* = F(x_*) \geq \min_{x \in B_i} F(x) \geq$$
$$\min_{x \in B_i} \mu_i(x) = \alpha_i \geq F(x') - \varepsilon \cdot 1_m.$$

Тем самым показано выполнение условия (2.4) и теорема доказана. □

Эксперименты проводились на компьютере с процессором Intel (R) Core(TM) 2 Quad, 2.83 MHz, имеющем 4Gb оперативной памяти. Для получения нижних оценок использовались интервальные миноранты [95]. Для экспериментального исследования предложенного алгоритма был выбран пример из работы [146].

Пример 2.1 *Задача с двумя критериями, двумя ограничениями и двумя параметрами*

$$f^{(1)}(x) = x^{(1)},$$
$$f^{(2)}(x) = x^{(2)},$$
$$-(x^{(1)})^2 - (x^{(2)})^2 + 1 + 0.1\cos(16 \arctan \tfrac{x^{(1)}}{x^{(2)}}) \leq 0,$$
$$(x^{(1)} - 0.5)^2 + (x^{(2)} - 0.5)^2 - 0.5 \leq 0.$$

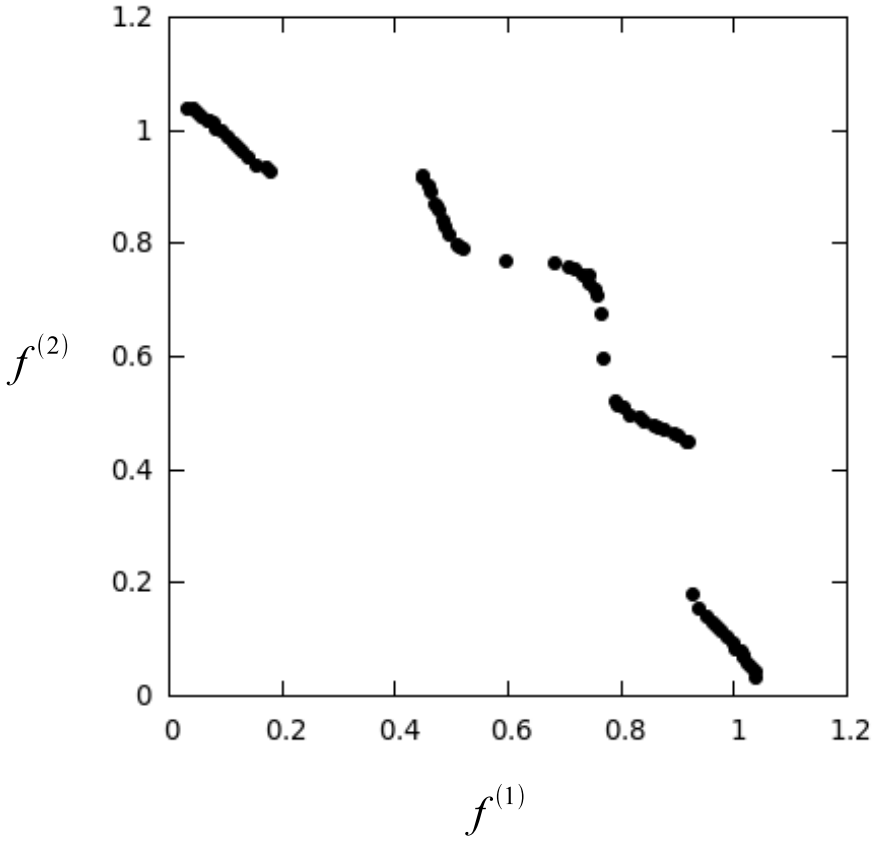

Рис. 2.6: Множество $0.01, 0.01$-Парето для примера 2.1.

Множество Парето для данной задачи многосвязно. При помощи метода неравномерных покрытий было построено множество ε, δ-Парето с $\varepsilon = 0.01$, $\delta = 0.01$. Для построения множества ε, δ-Парето потребовалось 1040 итераций алгоритма **Cover**. Вычисления заняли 0.03 секунды. Построенное множество, содержащее 70 точек, представлено на Рис. 2.6.

Пример 2.2 *Задача с двумя критериями, шестью ограничениями и шестью*

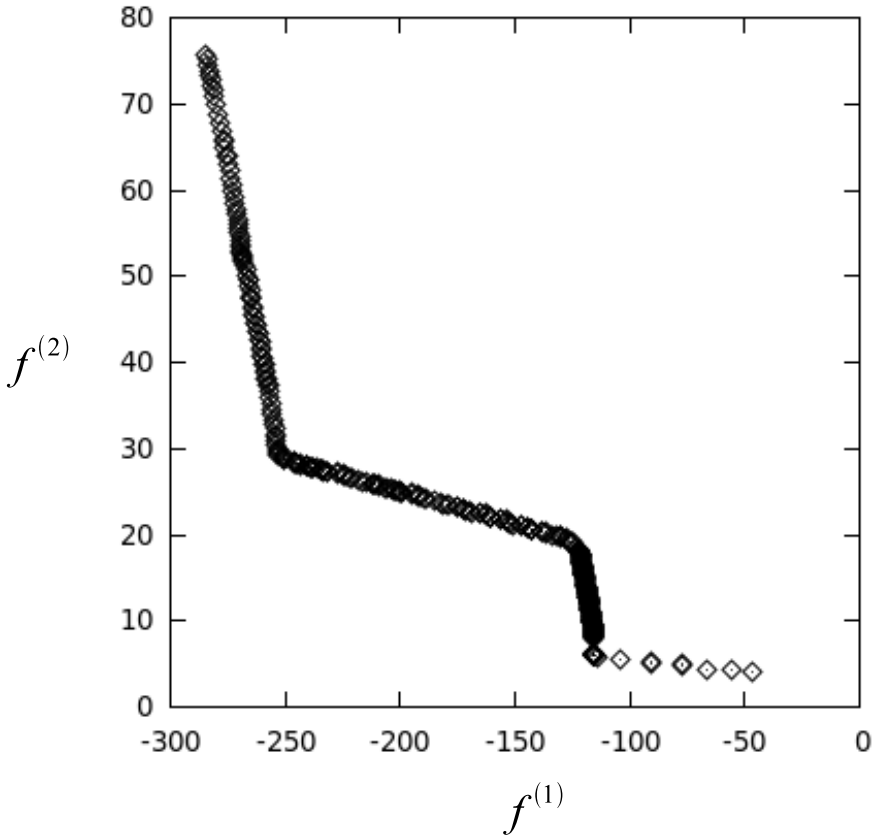

Рис. 2.7: Множество $1, 0.1$-Парето для примера 2.2.

параметрами

$$f^{(1)}(x) = -\left(25(x^{(1)}-1)^2 + (x^{(2)}-2)^2 + (x^{(3)}-1)^2 + (x^{(4)}-4)^2 + (x^{(5)}-1)^2\right),$$

$$f^{(2)}(x) = (x^{(1)})^2 + (x^{(2)})^2 + (x^{(3)})^2 + (x^{(4)})^2 + (x^{(5)})^2 + (x^{(6)})^2,$$

$$-x^{(1)} - x^{(2)} + 2 \leq 0,$$

$$x^{(1)} + x^{(2)} - 6 \leq 0,$$

$$-x^{(1)} + x^{(2)} - 2 \leq 0,$$

$$x^{(1)} - 3x^{(2)} - 2 \leq 0,$$

$$(x^{(3)}-3)^2 + x^{(4)} - 4 \leq 0,$$

$$-(x^{(5)}-3)^2 - x^{(6)} + 4 \leq 0.$$

Результаты работы метода неравномерных покрытий с параметрами $\varepsilon = 1$, $\delta = 0.1$ представлены на Рис. 2.7. Построение множества ε, δ-Парето было осуществлено за 17793690 итераций алгоритма **Cover**. Вычисления заняли 421 секунду. Построенное множество ε, δ-Парето содержит 382 точки.

Проведенные эксперименты показывают, что предложенный метод позволяет получать дискретные аппроксимации с высокой заданной точностью.

2.4 Основные результаты главы

Данная глава была посвящена разработке, развитию и реализации детерминированных методов многокритериальной оптимизации. Были получены следующие результаты:

1. Для задачи МКО с параллелепипедными ограничениями доказаны новые свойства ε-Парето множества, устанавливающие связь между этим множеством, границей Парето и оболочкой Эджворта-Парето.

2. Метод неравномерных покрытий для задач МКО обобщен на случай произвольных минорант, доказана сходимость метода.

3. Для задач МКО с функциональными ограничениями введено понятие приближенного решения — множества ε, δ-Парето и предложен алгоритм для его построения, доказана сходимость метода.

Глава 3

Эффективная оболочка невыпуклых множеств и метод ее аппроксимации

В данной главе определяется понятие *эффективной оболочки множества* и предлагается метод ее аппроксимации для случая, когда множество является образом компакта при непрерывном отображении. Показывается, что эффективная оболочка компактного множества содержится в его выпуклой оболочке. Предложенный метод аппроксимации эффективной оболочки, обобщающий подход, разработанный в [46, 76], применен для описания рабочей области многосекционного робота-манипулятора.

3.1 Эффективная граница и эффективная оболочка

В дальнейшем компоненты вектора из пространства $\mathbb{R}^n$ обозначаются верхним индексом в круглых скобках: $x = \left(x^{(1)}, \ldots, x^{(n)}\right)$.

Будем использовать следующую сокращенную запись:

$$a \leq b \Leftrightarrow a^{(i)} \leq b^{(i)}, i = 1, \ldots, n.$$

Для множества $V \subseteq \mathbb{R}^m$ и вектора $v \in \mathbb{R}^m$ определим сумму вектора и множества $v + V = \{x \in \mathbb{R}^m : x - v \in V\}$. Для двух множеств $V \subseteq \mathbb{R}^m$, $W \subseteq \mathbb{R}^m$ определим их сумму $V + W = \cup_{v \in V}(v + W)$.

Обозначим через Λ_m множество всех векторов размерности m, компонентами

которых являются 1 или -1. Для вектора $\lambda \in \Lambda_m$ запись $y_2 \succcurlyeq_\lambda y_1$ означает одновременное выполнение m условий

$$\left(y_2^{(i)} - y_1^{(i)}\right) \lambda^{(i)} \geq 0 \text{ для } i = 1, \ldots, m. \tag{3.1}$$

Если не выполнены соотношения $y_1 \succcurlyeq_\lambda y_2$ и $y_2 \succcurlyeq_\lambda y_1$, то векторы y_1 и y_2 называются *несравнимыми для данного* $\lambda \in \Lambda_m$.

Для точки $y \in \mathbb{R}^m$ и заданного $\lambda \in \Lambda_m$ определим множества $D_\lambda(y) = \{z \in \mathbb{R}^m : z \succcurlyeq_\lambda y\}$ и $D^*_\lambda(y) = \{z \in \mathbb{R}^m : y \succcurlyeq_\lambda z\}$. Введенные обозначения обобщаются на случай множества $Z \subseteq \mathbb{R}^m$: $D_\lambda(Z) = \cup_{z \in Z} D_\lambda(z)$, $D^*_\lambda(Z) = \cup_{z \in Z} D^*_\lambda(z)$.

Для произвольного компактного множества $Y \subseteq \mathbb{R}^m$ точку $y \in Y$ назовем λ-*точкой*, если в множестве Y нет таких точек z, что $z \succcurlyeq_\lambda y$ и $z \neq y$. Объединение всех λ-точек назовем λ-*границей множества* Y и обозначим через $\mathcal{P}_\lambda(Y)$, т.е.

$$\mathcal{P}_\lambda(Y) = \{y \in Y : Y \cap D_\lambda(y) = y\}. \tag{3.2}$$

Объединение λ-границ $\mathcal{P}_{eff}(Y) = \cup_{\lambda \in \Lambda_m} \mathcal{P}_\lambda(Y)$ назовем *эффективной границей множества* Y. На Рис. 3.1 даны иллюстрации понятий λ-границы и эффективной границы.

Напомним, что *строго выпуклым множеством* называется выпуклое множество, все граничные точки которого — крайние, т.е. не являются внутренними ни для какого отрезка, концы которого принадлежат этому множеству. Справедливо следующее утверждение.

Утверждение 3.1 *Эффективная граница компактного строго выпуклого множества совпадает с его границей.*

Несложно убедиться, что данное утверждение перестает быть верным если вместо строго выпуклого рассмотреть произвольное выпуклое множество. Например, эффективная граница n-мерного параллелепипеда совпадает с множеством его вершин, а граница дополнительно содержит грани и ребра.

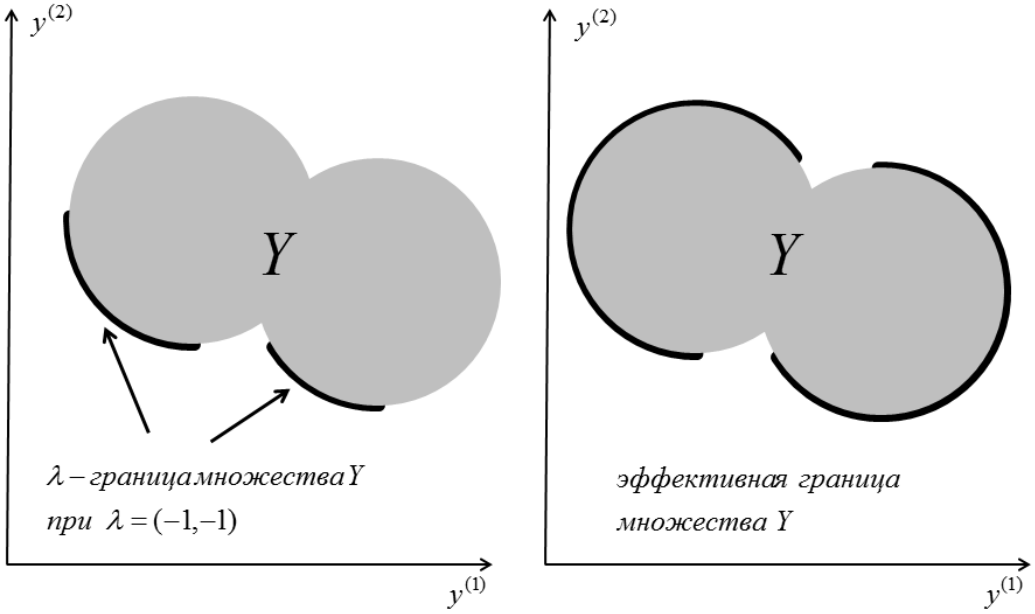

Рис. 3.1: λ-граница и эффективная граница множества Y

Для доказательства утверждения 3.1 потребуется несколько вспомогательных понятий и фактов.

Пусть $Y \subseteq \mathbb{R}^m$ — выпуклое множество, y — его граничная точка, $y \in \partial Y$. Гиперплоскость $H = \{z \in \mathbb{R}^m : h^T z = b\}$, $h \in \mathbb{R}^m$ называется *опорной в точке* y, если выполнены следующие два условия:

1. точка y принадлежит гиперплоскости H: $h^T y = b$;
2. множество Y целиком лежит в отрицательном полупространстве гиперплоскости H: $Y \subseteq H^-$, где $H^- = \{z \in \mathbb{R}^m : h^T z \leq b\}$.

Известно, что для каждой граничной точки выпуклого множества существует опорная гиперплоскость.

Лемма 3.1 *Если Y — строго выпуклое множество, а H — опорная гиперплоскость в точке $y \in \partial Y$, то $H^+ \cap Y = \{y\}$, где $H^+ = \{z \in \mathbb{R}^m : h^T z \geq b\}$.*

Доказательство. Предположим, что найдется точка $y' \in Y \cap H^+$, $y' \neq y$. В силу выпуклости множеств Y и H^+, отрезок $[y, y']$ также принадлежит пересечению $Y \cap H^+$. Следовательно, отрезок $[y, y'] \subseteq \partial Y$. Это противоречит

определению строгой выпуклости. Следовательно, утверждение леммы верно. □.

Используя данную лемму, докажем утверждение 3.1. Рассмотрим строго выпуклое множество $Y \subseteq \mathbb{R}^m$ с границей ∂Y. Включение $\mathcal{P}_{eff}(Y) \subseteq \partial Y$ очевидно. Покажем теперь, что любая граничная точка $y \in \partial Y$ принадлежит эффективной границе. Рассмотрим опорную гиперплоскость H в точке y. Покажем, что y является λ-точкой для вектора $\lambda \in \Lambda_m$, компоненты которого задаются следующей формулой:

$$\begin{cases} \lambda^{(i)} = 1, \text{ если } h^{(i)} \geq 0, \\ \lambda^{(i)} = -1, \text{ если } h^{(i)} < 0, \end{cases}$$

для $i = 1, \ldots, m$. Действительно, для любой точки $z \in D_\lambda(y)$ верно $(z^{(i)} - y^{(i)})\lambda^{(i)} \geq 0$, $i = 1, \ldots, m$. Следовательно $h^T z = \sum_{i=1}^m h^{(i)} z^{(i)} \geq \sum_{i=1}^m h^{(i)} y^{(i)} = b$. Таким образом $z \in H^+$. В силу произвольности выбора z имеем включение $D_\lambda(y) \subseteq H^+$. Тогда из леммы 3.1 следует $D_\lambda(y) \cap Y = y$, т.е. y — λ-точка и, значит, $y \in \mathcal{P}_{eff}(Y)$. Утверждение 3.1 доказано.

Множество $H(Y) = \cap_{\lambda \in \Lambda_m} D^*_\lambda \left(\mathcal{P}_\lambda(Y)\right)$ будем называть *эффективной оболочкой множества* Y. Так как для компактного множества $Y \subseteq \mathbb{R}^m$ справедливо включение $Y \subseteq D^*_\lambda \left(P_\lambda(Y)\right)$ при любом $\lambda \in \Lambda_m$, то $Y \subseteq H(Y)$. Следующая теорема связывает выпуклую оболочку $Conv(Y)$ множества Y (наименьшее выпуклое множество, содержащее Y) и его эффективную оболочку.

Теорема 3.1 *Для произвольного компактного множества Y справедливо включение*

$$Y \subseteq H(Y) \subseteq Conv(Y), \tag{3.3}$$

и равенство

$$Conv(Y) = Conv(\mathcal{P}_{eff}(Y)). \tag{3.4}$$

Доказательство. Теорема будет доказана, если будет установлено, что любая точка из $H(Y)$ представима в виде выпуклой комбинации точек множества $\mathcal{P}_{eff}(Y)$. Если $y \in H(Y)$, то по определению эффективной оболочки для каждого $\lambda \in \Lambda_m$ найдется точка $y_\lambda \in \mathcal{P}_\lambda(Y)$, такая что $y \in D^*_\lambda(y_\lambda)$. Покажем, что точка y представима в виде выпуклой комбинации точек $\{y_\lambda\}$, $\lambda \in \Lambda_m$. Докажем это утверждение методом математической индукции по размерности m пространства $\mathbb{R}^m$:

Рассмотрим случай $m = 1$. Пусть $y \in \mathcal{P}_h(Y)$. Тогда $y_{\{-1\}} \le y \le y_{\{1\}}$. В это случае очевидно $y = \kappa y_{\{-1\}} + (1-\kappa)y_{\{1\}}$, для некоторого κ, $0 \le \kappa \le 1$. Таким образом утверждение для $m = 1$ доказано.

Пусть утверждение доказано для $m \ge 1$. Докажем его справедливость для $m+1$. Рассмотрим произвольную точку y, принадлежащую эффективной оболочке множества $Y \subseteq \mathbb{R}^{m+1}$ и набор точек $y_\lambda \in \mathcal{P}_\lambda(Y)$, $\lambda \in \Lambda_{m+1}$, таких что $y \in D^*_\lambda(y_\lambda)$. Будем обозначать через $\hat{v}$ вектор, полученный из $v = (v^{(1)}, \ldots, v^{(m+1)})$ удалением последней компоненты: $\hat{v} = (v^1, \ldots, v^m)$. Заметим, что

$$\hat{y} \in D^*_{\hat{\lambda}}(\hat{y_\lambda}), \tag{3.5}$$

для $\lambda \in \Lambda_{m+1}$. Представим множество Λ_{m+1} в виде объединения множеств $\Lambda^+_{m+1} = \{\lambda \in \Lambda_{m+1} : \lambda^{(m+1)} = 1\}$ и $\Lambda^-_{m+1} = \{\lambda \in \Lambda_{m+1} : \lambda^{(m+1)} = -1\}$. По предположению индукции и согласно соотношению (3.5) найдутся наборы $\{\alpha_\lambda\}$ и $\{\beta_\lambda\}$, такие что:

$$\begin{aligned}
&\textstyle\sum_{\lambda\in\Lambda^+_{m+1}} \alpha_\lambda = 1,\ \alpha_\lambda \ge 0, \lambda \in \Lambda^+_{m+1},\\
&\textstyle\sum_{\lambda\in\Lambda^-_{m+1}} \beta_\lambda = 1,\ \beta_\lambda \ge 0, \lambda \in \Lambda^-_{m+1},\\
&\hat{y} = \textstyle\sum_{\lambda\in\Lambda^+_{m+1}} \alpha_\lambda \hat{y}_\lambda = \sum_{\lambda\in\Lambda^-_{m+1}} \beta_\lambda \hat{y}_\lambda,\\
&\text{где } \hat{v} = (v^1, \ldots, v^m).
\end{aligned} \tag{3.6}$$

Очевидно, справедливы следующие неравенства:

$$y_\lambda^{(m+1)} \le y^{(m+1)}, \text{ для } \lambda \in \Lambda_{m+1}^-, \tag{3.7}$$

$$y_\lambda^{(m+1)} \ge y^{(m+1)}, \text{ для } \lambda \in \Lambda_{m+1}^+, \tag{3.8}$$

Рассмотрим точки $y_+ = \sum_{\lambda\in\Lambda_{m+1}^+} \alpha_\lambda y_\lambda$ и $y_- = \sum_{\lambda\in\Lambda_{m+1}^-} \beta_\lambda y_\lambda$. Из (3.6) следует, что $y_-^{(i)} = y_+^{(i)} = y^{(i)}$ для $i = 1, \ldots, m$. Из неравенств (3.7), (3.8) следует, что $y_-^{(m+1)} \le y^{(m+1)} \le y_+^{(m+1)}$. Тогда найдется такое $0 \le \kappa \le 1$, что $y^{(1)} = \kappa y_-^{(1)} + (1-\kappa)y_+^{(1)}$. Таким образом, $y = \kappa y_- + (1-\kappa)y_+$.

Рассмотрим набор $\{\theta_\lambda\}$, $\lambda \in \Lambda_{m+1}$ такой, что

$$\theta_\lambda = \begin{cases} (1-\kappa)\alpha_\lambda, \text{ при } \lambda \in \Lambda_{m+1}^+, \\ \kappa\beta_\lambda \text{ при } \lambda \in \Lambda_{m+1}^-. \end{cases}$$

Очевидно, что $\theta_\lambda \ge 0, \lambda \in \Lambda_{m+1}$, $\sum_{\lambda\in\Lambda_{m+1}} \theta_\lambda = 1$ и $\sum_{\lambda\in\Lambda_{m+1}} \theta_\lambda y_\lambda = y$. Тем самым показано, что y — выпуклая комбинация точек $\{y_\lambda\}$, $\lambda \in \Lambda_{m+1}$. □

Эффективная оболочка, вообще говоря, ближе к множеству Y, чем его выпуклая оболочка и, в этом смысле, дает более точное описание этого множества (Рис. 3.2). Из теоремы 3.1 следует, что для компактного выпуклого множества само множество, его эффективная оболочка и выпуклая оболочка совпадают.

3.2 Аппроксимация эффективной оболочки

Для $\varepsilon \ge 0$ и некоторого $\lambda \in \Lambda_m$ множество точек Y_ε^λ из $\mathbb{R}^m$ назовем *ε-аппроксимацией λ-границы*, если для каждой точки $y_* \in \mathcal{P}_\lambda(Y)$ найдется такая точка y_ε в Y_ε^λ, что $y_\varepsilon + \varepsilon\lambda \succcurlyeq_\lambda y_*$ и, кроме того, множество Y_ε^λ состоит только из несравнимых точек.

Формально перечисленные условия записываются следующим образом:

$$\mathcal{P}_\lambda(Y) \subseteq D_\lambda^*(Y_\varepsilon^\lambda + \varepsilon\lambda), \tag{3.9}$$

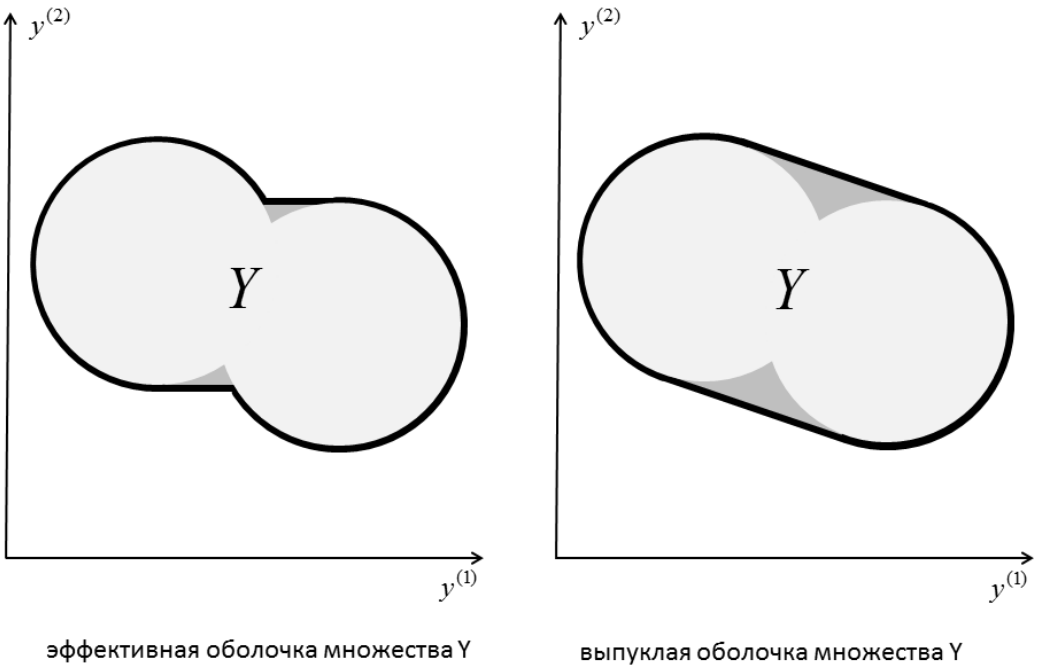

Рис. 3.2: Эффективная и выпуклая оболочки множества Y

$$\mathcal{P}_\lambda(Y_\varepsilon^\lambda) = Y_\varepsilon^\lambda. \tag{3.10}$$

Пусть имеется набор $\{Y_\lambda^\varepsilon\}, \lambda \in \Lambda_m$. Тогда $\cup_{\lambda\in\Lambda_m} Y_\lambda^\varepsilon$ назовем *ε-эффективной границей* множества Y. *Внешней ε-эффективной границей* будем называть множество $\cup_{\lambda\in\Lambda_m} (Y_\lambda^\varepsilon + \varepsilon\lambda)$.

Так как Y — компакт, то $Y \subseteq D_\lambda^* (P_\lambda(Y))$ и, следовательно, $Y \subseteq D_\lambda^* (Y_\lambda^\varepsilon + \varepsilon\lambda)$ для всех $\lambda \in \Lambda_m$. Таким образом, $Y \subseteq \cap_{\lambda\in\Lambda_m} D_\lambda^* (Y_\lambda^\varepsilon + \lambda\varepsilon)$.

Множество $\cap_{\lambda\in\Lambda_m} D_\lambda^* (Y_\lambda^\varepsilon + \lambda\varepsilon)$ будем называть *ε-эффективной оболочкой множества* Y. Очевидно $H(Y) \subseteq \cap_{\lambda\in\Lambda_m} D_\lambda^* (Y_\lambda^\varepsilon + \lambda\varepsilon)$ для любого $\varepsilon > 0$.

На Рис. 3.3 изображена ε-эффективная граница множества Y, состоящая из 25 светлых кружков. В качестве λ брались векторы из множества $\Lambda_2 = \{(1,1), (1,-1), (-1,1), (-1,-1)\}$. Темные кружки обозначают точки внешней ε-эффективной границы. Темно-серым цветом закрашена ε-эффективная оболочка множества Y.

Справедлива следующая лемма.

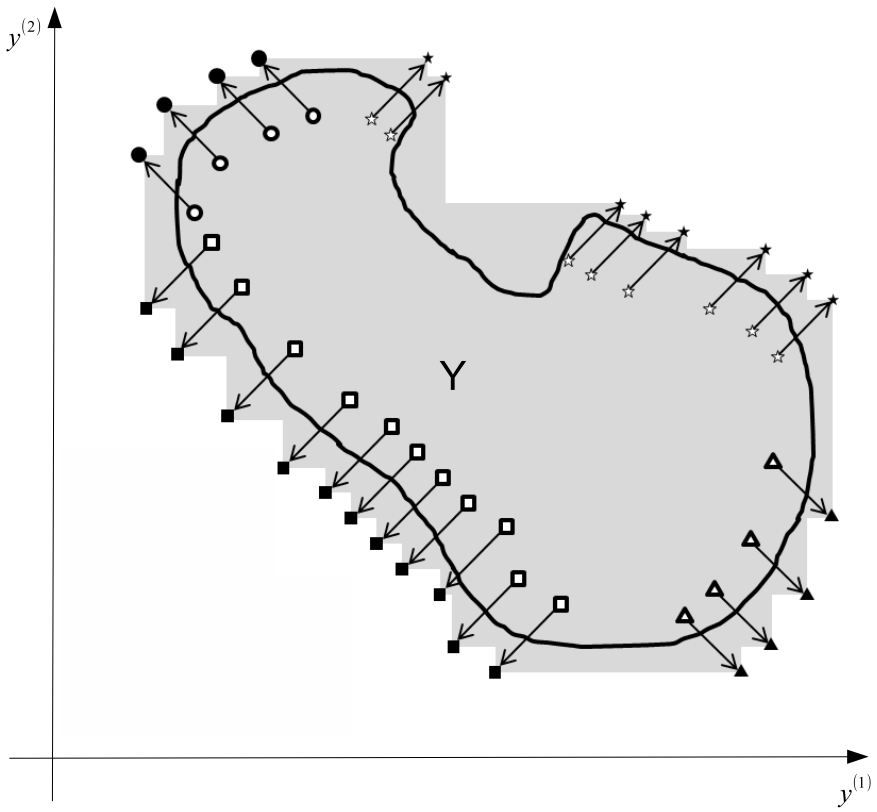

Рис. 3.3: ε-эффективная граница множества Y

Лемма 3.2 *Пусть* $H_\varepsilon \cap_{\lambda\in\Lambda_m} D^*_\lambda\left(Y^\varepsilon_\lambda + \lambda\varepsilon\right)$ *— ε-эффективная оболочка множества* Y*, а* $\overline{B}_\varepsilon = \cup_{\lambda\in\Lambda_m}\left(Y^\varepsilon_\lambda + \lambda\varepsilon\right)$ *— соответствующая внешняя ε-эффективная граница. Тогда выполняется*

$$\mathcal{P}_{eff}(H_\varepsilon) = \overline{B}_\varepsilon. \tag{3.11}$$

Доказательство. По определению $H_\varepsilon = \cap_{\lambda\in\Lambda_m} D^*_\lambda\left(Y^\varepsilon_\lambda + \lambda\varepsilon\right)$. Рассмотрим произвольный вектор $\lambda \in \Lambda_m$. Покажем, что $\mathcal{P}_\lambda(H_\varepsilon) = Y^\varepsilon_\lambda + \lambda\varepsilon$. Пусть $y \in Y^\varepsilon_\lambda + \lambda\varepsilon$. Предположим, что $D_\lambda(y) \cap H_\varepsilon$ содержит точку $z \neq y$. По определению $D_\lambda(y)$ выполняется

$$z \succcurlyeq_\lambda y. \tag{3.12}$$

Так как $z \in \cap_{\lambda\in\Lambda_m} D^*_\lambda\left(Y^\varepsilon_\lambda + \lambda\varepsilon\right)$, то $z \in D^*_\lambda\left(Y^\varepsilon_\lambda + \lambda\varepsilon\right)$ и, значит

$$t \succcurlyeq_\lambda z. \tag{3.13}$$

для некоторого $t \in Y^\varepsilon_\lambda + \lambda\varepsilon$. Из неравенств (3.12) и (3.13) следует, что $t \succcurlyeq_\lambda y$, причем $y \in Y^\varepsilon_\lambda + \lambda\varepsilon$ и $t \in Y^\varepsilon_\lambda + \lambda\varepsilon$. Пусть $t' = t - \varepsilon\lambda$, $y' = y - \varepsilon\lambda$, где

$y' \in Y_\lambda^\varepsilon$, $t' \in Y_\lambda^\varepsilon$. Тогда выполняется $t' \succcurlyeq_\lambda y'$, что противоречит свойству (3.10). Значит, наше предположение неверно и $D_\lambda(y) \cap H_\varepsilon = y$. Тем самым доказано, что $Y_\lambda^\varepsilon + \lambda\varepsilon \subseteq \mathcal{P}_\lambda(H_\varepsilon)$. Включение $\mathcal{P}_\lambda(H_\varepsilon) \subseteq Y_\lambda^\varepsilon + \lambda\varepsilon$ очевидно. Таким образом для всех $\lambda \in \Lambda_m$ выполняется $\mathcal{P}_\lambda(H_\varepsilon) = Y_\lambda^\varepsilon + \lambda\varepsilon$. Отсюда непосредственно следует соотношение (3.11). Лемма доказана.□

Из доказанной леммы и соотношения (3.4) следует утверждение.

Утверждение 3.2 *Пусть* $H_\varepsilon \cap_{\lambda\in\Lambda_m} D_\lambda^* \left(Y_\lambda^\varepsilon + \lambda\varepsilon\right)$ *— ε-эффективная оболочка множества Y, а* $\overline{B}_\varepsilon = \cup_{\lambda\in\Lambda_m} \left(Y_\lambda^\varepsilon + \lambda\varepsilon\right)$ *— соответствующая внешняя ε-эффективная граница. Тогда выполняется включение*

$$Y \subseteq H_\varepsilon \subseteq Conv(\overline{B}_\varepsilon). \tag{3.14}$$

Включение (3.14) дает конструктивную процедуру построения выпуклого множества, содержащего множество Y. Для этого сначала строится ε-эффективная граница множества Y — конечное множество B_ε, а затем строится выпуклая оболочка множества $\overline{B}_\varepsilon$ известными алгоритмами.

Алгоритм построения ε-эффективная границы

Приведем алгоритм построения ε-эффективной границы множества Y, заданного как образ n-мерного параллелепипеда X при непрерывном отображении $F : \mathbb{R}^n \to \mathbb{R}^m$, $F(\cdot) = \left(f^{(1)}(\cdot), \ldots, f^{(m)}(\cdot)\right)$. Для каждого $\lambda \in \Lambda_m$ выполняется нахождение ε-аппроксимации λ-границы множества $Y = F(X)$. Объединение построенных таким образом множеств образует ε-эффективная границу множества Y.

В процессе работы алгоритма поддерживается список A точек из множества X такой, что его образ при отображении F содержит конечный набор попарно несравнимых точек, который после завершения работы алгоритма будет ε-аппроксимацией λ-границы множества Y. Список строится последовательно с

использованием процедуры **Update**, реализующей добавление очередной точки к списку таким образом, чтобы указанное свойство несравнимости выполнялось.

Процедура Update (λ, A, x)

Параметры:

λ — вектор, компоненты которого входят в условие (3.1)

A — список точек из X

$x \in X$ — новая точка

1. НАЧАЛО ПРОЦЕДУРЫ
2. ЦИКЛ по всем точкам $a \in A$:

   если $F(a) \succcurlyeq_\lambda F(x)$, то перейти к пункту 4

   если $F(x) \succcurlyeq_\lambda F(a)$, то точку a удалить из списка A

   КОНЕЦ ЦИКЛА
3. Добавить точку x к списку A: $A = A \cup x$;
4. КОНЕЦ ПРОЦЕДУРЫ

Приведем алгоритм построения ε-аппроксимации λ-границы для рассматриваемого случая. В процессе работы алгоритма, исходный параллелепипед X разбивается на меньшие параллелепипеды. На каждом параллелепипеде B, получаемом в результате разбиения, строятся *оценочные функции* $\mu_B(\cdot) : \mathbb{R}^n \to \mathbb{R}^m$ для функции $F(\cdot)$, удовлетворяющие соотношениям:

$$\mu_B(x) \succcurlyeq_\lambda F(x),\ x \in B,\ k = 1, \ldots, m.$$

Основное требование к используемым оценочным функциям состоит в том, чтобы их максимум на параллелепипеде находился просто. К этому классу относятся, например, Липшицевы [46, 147] и интервальные миноранты [148, 149].

Алгоритм **Bisect** построения ε-аппроксимации λ-границы множества Y формирует в процессе своей работы список S, состоящий из n-мерных параллелепипедов и список точек A. Перед началом работы алгоритма список S состоит из единственного элемента — параллелепипеда X. Список точек A пуст. Алгоритм останавливается, когда список параллелепипедов становится пустым, при этом набор точек $F(A)$ будет искомой ε-аппроксимацией λ-границы множества Y.

Процедура Bisect (λ, ε, X, F, A)

Параметры:

ε — точность аппроксимации

λ — параметр, определяющий направление построения аппроксимации

X — исходное множество

F — функциональное отображение

A — список точек из X

1. НАЧАЛО ПРОЦЕДУРЫ
2. $S = \{X\}$, $A = \emptyset$
3. Если $S = \emptyset$, то завершить алгоритм
4. Выбрать и удалить из списка S произвольный параллелепипед B
5. Определить точку c — центр параллелепипеда и выполнить процедуру Update(λ, A,c)
6. Для $i = 1, \ldots, m$ определить $\alpha^{(i)} = \begin{cases} \max_{x \in B} \mu_B^{(i)}(x), \text{ если } \lambda_i = 1, \\ \min_{x \in B} \mu_B^{(i)}(x), \text{ если } \lambda_i = -1. \end{cases}$. Если $A \neq \emptyset$ и $\alpha \in D_\lambda^*(F(A) + \varepsilon\lambda)$, то перейти к пункту 2
7. Разбить параллелепипед B на два одинаковых параллелепипеда B' и B'' вдоль ребра максимальной длины и добавить их к списку S, перейти к пункту 3

8. КОНЕЦ ПРОЦЕДУРЫ

Алгоритм построения ε-эффективной границы множества Y состоит в применении процедуры **Bisect** для всех $\lambda \in \Lambda_m$ и объединении результатов. В результате будет построено множество A такое, что $F(A)$ — ε-эффективная граница Y.

Процедура Epsbound (ε, X, F, A)

Параметры:

ε — точность аппроксимации

λ — параметр, определяющий направление построения аппроксимации

X — исходное множество

F — функциональное отображение

A — список точек

1. НАЧАЛО ПРОЦЕДУРЫ
2. $A = \emptyset$.
3. ЦИКЛ по всем $\lambda \in \Lambda_m$:

   Выполнить процедуру Bisect(λ, ε, X, F, A')

   $A = A \cup A'$

   КОНЕЦ ЦИКЛА
4. КОНЕЦ ПРОЦЕДУРЫ

3.3 Применение эффективной оболочки для описания рабочей области робота-манипулятора

Идея применить эвристические методы многокритериальной оптимизации для задач аппроксимации границы рабочей области робота-манипулятора была предложена в [150]. Метод, использованный в [150], не гарантировал того, что рабочая область целиком лежит внутри построенной аппроксимации. Для ряда

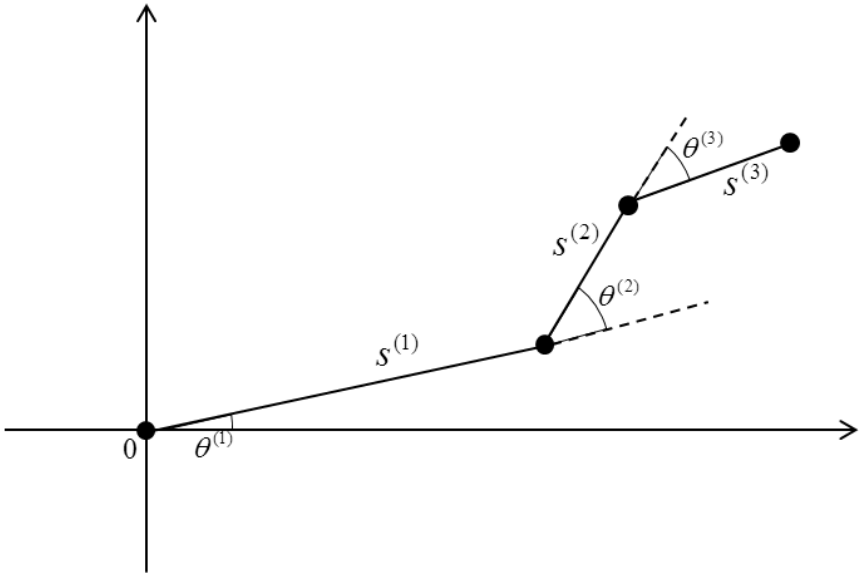

Рис. 3.4: Углы и звенья, определяющие конфигурацию робота-манипулятора

приложений (космическая отрасль, медицина) подобная гарантия необходима. Предлагаемый далее подход позволяет строить множество, гарантированно содержащее рабочую область.

Рассматривается планарный робот с k звеньями переменной длины (Рис. 3.4). Один из концов манипулятора закреплен в начале координат. Рабочая область определяется как множество возможных позиций свободного (второго) конца робота, также называемого *схватом*. Позиция схвата полностью определяется вектором длин звеньев $s = \left(s^{(1)}, \ldots, s^{(k)}\right)$ и вектором величин углов $\theta = (\theta^{(1)}, \ldots, \theta^{(k)})$ между соответствующими звеньями. Допустимое множество углов и длин звеньев является параллелепипедом

$$X = [\underline{s^{(1)}}, \overline{s^{(1)}}] \times \cdots \times [\underline{s^{(k)}}, \overline{s^{(k)}}] \times [\underline{\theta^{(1)}}, \overline{\theta_1}] \times \cdots \times [\underline{\theta^{(k)}}, \overline{\theta^{(k)}}],$$

где $[\underline{s^{(i)}}, \overline{s^{(i)}}]$ и $[\underline{\theta^{(i)}}, \overline{\theta^{(i)}}]$ — заданные диапазоны возможных значений длины звена s_i и угла θ_i соответственно.

Рабочая область робота-манипулятора — это образ $Y = F(X)$ допустимого

множества X, где $F(s,\theta) = \left(f^{(1)}(s,\theta), f^{(2)}(s,\theta)\right)$ задается формулами (3.15).

$$f^{(1)}(s,\theta) = \sum_{i=1}^{k} s^{(i)} \cdot \cos\left(\sum_{j=1}^{i} \theta^{(j)}\right), f^{(2)}(s,\theta) = \sum_{i=1}^{k} s^{(i)} \cdot \sin\left(\sum_{j=1}^{i} \theta^{(j)}\right). \quad (3.15)$$

Для рабочей области Y строится ε-эффективная граница, с помощью метода, изложенного в предыдущем пункте. Построенное множество точек полностью определяет ε-эффективную оболочку, включающую в себя рабочую область робота-манипулятора. Уменьшая ε, можно получать более точные аппроксимации, при этом возрастает и время расчетов.

Рассмотрим пример трехзвенного робота-манипулятора с фиксированными длинами звеньев $s_1 = 3, s_2 = 2, s_3 = 0.25$ и заданными диапазонами изменения углов $\theta_1 \in [\pi/6, \pi/3]$, $\theta_2 \in [\pi/6, \pi/3]$, $\theta_3 \in [-\pi, \pi]$. На рис. 3.5 приведена внешняя ε-эффективная граница при различных $\varepsilon = 0.5, 0.25, 0.01$. Время расчетов на персональном компьютере Intel (TM) Core i5 CPU, 3.1 GHz, 4Gb Ram составило 0.2 секунды, 0.4 секунды и 47.9 секунда соответственно. Данный пример наглядно демонстрирует повышение точности аппроксимации, сопровождаемое увеличением времени расчетов.

Рис. 3.6 содержит аппроксимации внешней границы рабочей области робота манипулятора с тремя звеньями для трех вариантов ограничений. Использовались следующие ограничения на углы между звеньями:

$$0 \le \theta_1 \le \pi/6,\ \pi/6 \le \theta_2 \le \pi/3,\ -\pi \le \theta_3 \le \pi.$$

Расчеты проводились для трех различных диапазонов значений звеньев:

$$1\text{: } s_1 = 1,\ s_2 = 1,\ 0.1 \le s_3 \le 0.25,$$
$$2\text{: } s_1 = 1,\ s_2 = 1,\ 0.1 \le s_3 \le 0.5,$$
$$3\text{: } s_1 = 1,\ 0.5 \le s_2 \le 1.5,\ 0.1 \le s_3 \le 0.5.$$

на персональном компьютере с конфигурацией Intel (c) Core i-5, 3.1 GHz, 4 Gb оперативной памяти с точностью, равной $\varepsilon = 0.01$. Время вычислений для

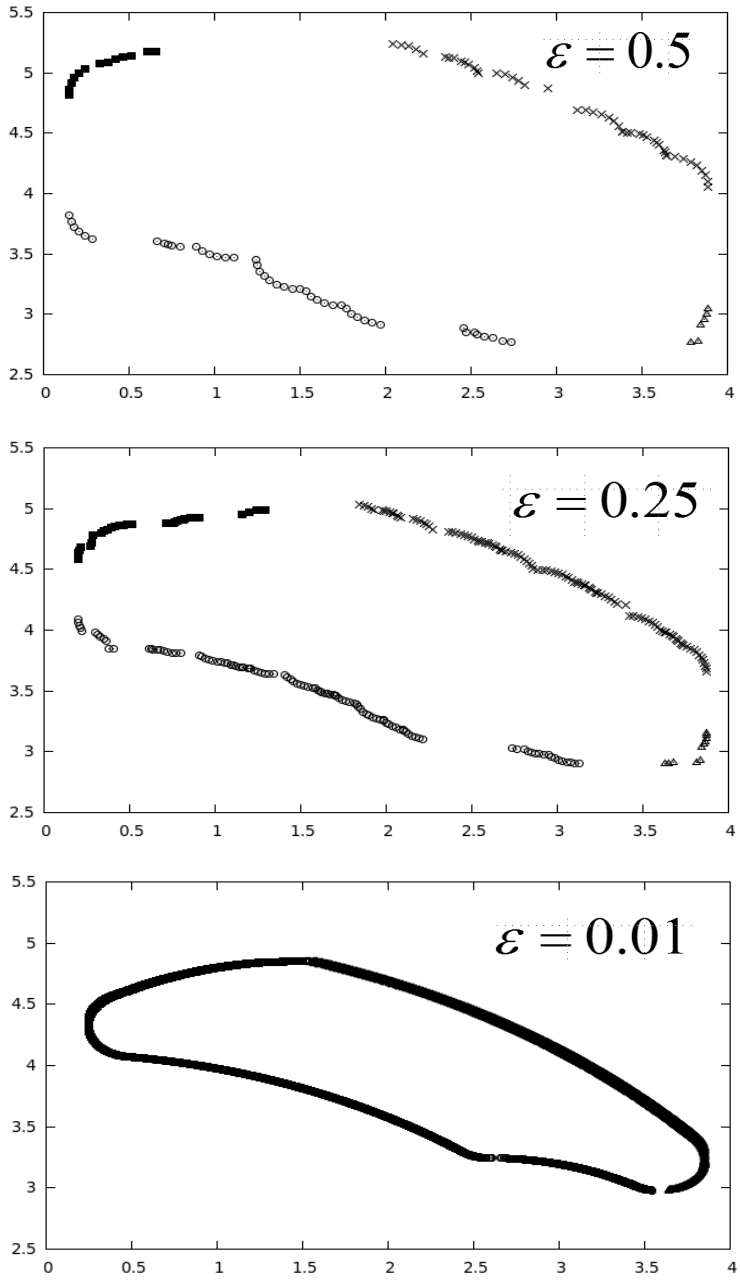

Рис. 3.5: Внешняя ε-эффективная граница при разных значениях ε

первого, второго и третьего варианта заняли 15 секунд, 18 секунд и 1 минуту 23 секунды соответственно.

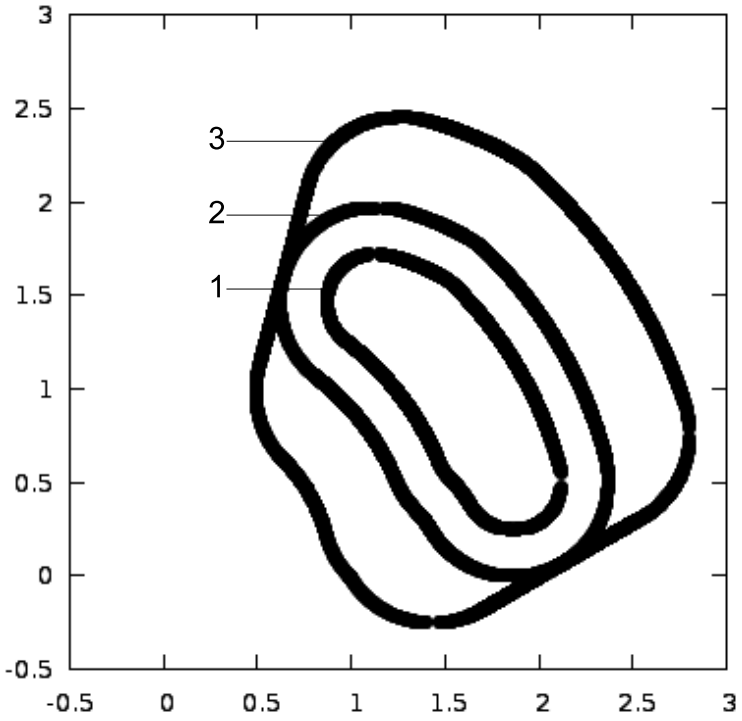

Рис. 3.6: Пример аппроксимации границы рабочей области робота-манипулятора со звеньями переменной длины

3.4 Основные результаты главы

В данной главе получены следующие результаты:

1. Определено понятие эффективной границы и эффективной оболочки множества, доказаны их свойства, отражающие связь со свойствами выпуклости.

2. Введены понятия ε-эффективной границы и ε-эффективной оболочки множества. Разработан метод численного построения этих объектов.

3. Введенные понятия применены для построения рабочей области многосекционного робота-манипулятора с заданной точностью.

Глава 4

Оценки сложности метода ветвей и границ

4.1 Постановка задачи и обзор существующих результатов

Различные аспекты сложности алгоритмов решения задач оптимизации изучались достаточно давно. Сложность детерминированных методов решения различных классов оптимизационных задач рассматривалась в работах Заозерской Л.А., Колоколова А.А. [86, 87]. Сложность локальных методов в задачах дискретной оптимизации исследовалась в работах Алексеевой Е.В., Кочетова Ю.А., Плясунова А.В. [84, 151]. Сложностные свойства приближенных алгоритмов исследовались в работах Хачая М.Ю., Гимади Э.Х., Кельманова А.А. и др. [82, 83, 152, 153].

В работах И.Х. Сигала [154, 155] рассматриваются вопросы вычислительной сложности для задач дискретной оптимизации. Также следует отметить работу Ю.Ю. Финкельштейна [24], в которой даются оценки сложности метода ветвей и границ для конкретных вариантов метода ветвей и границ в задачах ранцевого типа и работу В.П. Гришухина [156] по данной теме.

Метод ветвей и границ является является одним из наиболее распространенных подходов к решению задач глобальной оптимизации. Рассмотрим общую

схему метода для задачи:

$$f(x) \to \min, x \in X, \tag{4.1}$$

где $f(\cdot)$ — некоторая (не обязательно скалярная) функция, а X — допустимое множество.

Метод ветвей и границ состоит в последовательной декомпозиции исходной задачи на подзадачи, с исключением подзадач, которые заведомо не содержат оптимального решения. Подзадачи как правило задаются их допустимым подмножеством, а декомпозиция подзадач сводится к разбиению допустимых подмножеств. Общая схема метода состоит в следующем:

1. Поместить в список подмножеств S исходное допустимое множество X [1].
2. Выбрать и удалить из списка S некоторое подмножество G.
3. Произвести действия, направленные на вычисление верхних и нижних оценок целевой функции, сокращение и разбиение подмножества G. Полученные новые подмножества (если таковые получены) добавить в список S.
4. Если список S — пуст, то закончить алгоритм, в противном случае перейти к шагу 2.

Содержание шагов 1-4 приведенной общей схемы определяется конкретной реализацией алгоритма и особенностями решаемой задачи. Результатом работы алгоритма является допустимое решение с наименьшим найденным значением целевой функции либо множество недоминируемых решений в случае нескольких критериев, найденные в процессе работы алгоритма.

Процесс работы метода ветвей и границ можно схематично представить в виде *дерева ветвлений* — ациклического ориентированного графа, вершинами которого являются обрабатываемые подмножества, а дуги соединяют одно

[1] В некоторых случаях (например в случае задачи математического программирования) добавляется множество простой структуры, содержащее X

подмножество с другими, полученными из него в результате декомпозиции, выполненной на шаге 3 алгоритма.

Сложностью метода ветвей и границ при решении данной задачи будем называть число итераций цикла 2-4. Очевидно, что это число совпадает с количеством вершин в дереве ветвлений на момент завершения работы алгоритма. Известно, что число вершин дерева S связано с числом его концевых вершин соотношением $S = 2 * V - 1$. Количество концевых вершин дерева ветвлений будем называть *приведенной сложностью*.

4.2 Сложность метода бисекций

Основным способом алгоритмической реализации метода неравномерных покрытий является метод бисекций. Варианты этого алгоритма были изложены в главах 1, 2.

Базовый вариант метода бисекций характеризуется следующими свойствами:

1. исходное допустимое множество X и подмножества, хранящиеся в списке S, являются n-мерными параллелепипедами;
2. декомпозиция на шаге 3 общей схемы МВГ представляет собой деление параллелепипеда на два одинаковых параллелепипеда гиперплоскостью, проходящий через середину самого длинного ребра перпендикулярно ему.

Докажем следующую теорему.

Теорема 4.1 *Пусть существует $d_* > 0$ такое, что любой параллелепипед с диаметром, не превосходящим d_*, всегда полностью исключается из рассмотрения на шаге 3 метода ветвей и границ, d — диаметр исходного параллелепипеда. Тогда сложность метода бисекций не превосходит величины*

$$4\left(\frac{d_*}{d}\right)^{\theta_n} - 1, \tag{4.2}$$

где $\theta_n = 2/\log_2\left(1 - \frac{3}{4n}\right)$.

Доказательство. Рассмотрим произвольный n-мерный параллелепипед $B = [a^{(1)}, b^{(1)}] \times \cdots \times [a^{(n)}, b^{(n)}]$ с диаметром $d(B) = \frac{1}{2}\sqrt{(b^{(1)} - a^{(1)})^2 + \cdots + (b^{(n)} - a^{(n)})^2}$. Пусть j — номер самого длинного ребра параллелепипеда B. Диаметр полученного в результате декомпозиции параллелепипеда B' вычисляется по следующей формуле:

$$d(B') = \frac{1}{2}\sqrt{(b^{(1)} - a^{(1)})^2 + \cdots + \frac{(b^{(j)} - a^{(j)})^2}{4} + \cdots + (b^{(n)} - a^{(n)})^2}.$$

Так как $|b^{(j)} - a^{(j)}| \geq |b^{(i)} - a^{(i)}|$ для любого $1 \leq i \leq n$, то

$$d(B') = \tfrac{1}{2}\sqrt{4d(B)^2 - \tfrac{3}{4}(b^{(j)} - a^{(j)})^2} \leq$$
$$\leq \tfrac{1}{2}\sqrt{4d(B)^2 - \tfrac{3}{4n}4d(B)^2} = d(B)\sqrt{1 - \tfrac{3}{4n}}.$$

Следовательно, $d(B')/d(B) \leq \sqrt{1 - \frac{3}{4n}}$, минимально возможное уменьшение диагонали при декомпозиции составляет величину $\alpha_n = \sqrt{1 - \frac{3}{4n}}$.

Рассмотрим дерево ветвления, соответствующее работе алгоритма бисекций. Так как по условию теоремы параллелепипеды с диаметром, меньшим d_*, полностью исключаются из рассмотрения, то диаметры параллелепипедов, соответствующих внутренним вершинам дерева ветвления, не меньше чем d_*. Следовательно, справедливо соотношение

$$\alpha_n^{(k-1)} d \geq d_*, \tag{4.3}$$

где k — максимальная длина пути от корня дерева до его концевой вершины. Преобразуя выражение (4.3), получим:

$$k \leq \log_{\alpha_n}\left(\frac{d_*}{d}\right) + 1 = \log_2\left(\frac{d_*}{d}\right) / \log_2 \alpha_n + 1.$$

Так как дерево ветвления — бинарное, то вычислительная сложность S_n, соответствующая общему числу вершин в дереве, не превосходит величины $2^{k+1} - 1$. Таким образом

$$S_n \leq 2^{k+1} - 1 \leq 4\left(\frac{d_*}{d}\right)^{1/\log_2 \alpha_n} - 1 = 4\left(\frac{d_*}{d}\right)^{\theta_n} - 1,$$

где $\theta_n = 2/\log_2\left(1-\frac{3}{4n}\right)$. Что и требовалось доказать. $\square$

Оценим асимптотическое поведение функции сложности при росте n. Заметим, что $\log_2(1-\frac{3}{4n}) = \frac{1}{\ln 2}\ln(1-\frac{3}{4n}) \simeq -\frac{1}{\ln 2}\frac{3}{4n}$. Поэтому, при $n \to \infty$ верхняя оценка асимптотически ведет себя как $4\left(d/d_*\right)^{\frac{8\cdot\ln 2}{3}n}$. Полученное соотношение показывает, что верхняя оценка сложности (4.2) экспоненциально растет с ростом n.

Из полученной оценки следует конечность числа шагов метода бисекций при условии $d_* > 0$. Обеспечение выполнения этого свойства достигается за счет использования асимптотически точных минорант.

Будем говорить, что $\mu(\cdot)$ является *асимптотически точной минорантой функции* $f(\cdot)$, если для произвольного $\varepsilon > 0$ найдется такое $d(\varepsilon) > 0$, что для любого параллелепипеда B с диаметром, меньшим или равным $d(\varepsilon)$ выполняется

$$f(c) - \mu_* \le \varepsilon. \tag{4.4}$$

Здесь через c обозначен центр параллелепипеда B, а $\mu_* = \min_{x\in B}\mu(x)$. Функцию $d(\varepsilon)$ будем называть *модулем точности*.

Рассмотрим наиболее общий случай — задачу многокритериальной оптимизации (2.1) в которой допустимое множество задано с помощью функциональных ограничений (2.30). Пусть каждая из минорант $\mu^{(i)}(\cdot)$, $i = 1,\ldots,m$ является асимптотически точной для своего критерия с модулем точности $d^{(i)}(\cdot)$. Пусть миноранта $\nu^{(i)}(\cdot)$ также является асимптотически точной для i-го ограничения с модулем точности $d^{(m+i)}(\cdot)$, $i = 1,\ldots,k$. Положим

$$d_1 = \min_{i=1,\ldots,m} d^{(i)}(\varepsilon), d_2 = \min_{i=1,\ldots,k} d^{(m+i)}(\delta), d_* = \min(d_1, d_2).$$

Рассмотрим параллелепипед B, с диаметром, не превосходящем d_*. Если центр параллелепипеда c не является δ-допустимой точкой, то найдется хотя бы одно ограничение $g^{(j)}(\cdot)$ такое, что $g^{(j)}(c) > \delta$. Поскольку $d^{(m+j)}(\varepsilon) \ge d_*$, то со-

гласно (4.4) $g^{(j)}(c) - \nu_*^{(j)} \leq \delta$. Следовательно $\nu_*^{(j)} > 0$ и, значит параллелепипед B исключается из дальнейшего рассмотрения.

Пусть теперь c является δ-допустимой точкой. Согласно свойству (4.4) для всех $i = 1, \ldots, m$ имеем $f^{(i)}(c) - \mu_*^{(i)} \leq \varepsilon$, где $\mu_*^{(i)} = \min_{x \in B} \mu^{(i)}(x)$. Следовательно $F(c) - \varepsilon \cdot e_m \leq \mu_*$, то есть точка $F(c)$ ε-доминирует точку μ_*. Так как точка c является δ-допустимой, то μ_* доминируется текущим архивом. По правилу отсева параллелепипед B будет исключен из дальнейшего рассмотрения.

4.3 Общие сведения о сложности метода ветвей и границ для задачи о ранце

Задача о ранце [157, 158] с одним ограничением является одной из классических задач дискретной оптимизации, применяющейся при моделировании различных экономических процессов, решении проблем, возникающих в промышленном производстве, планировании, управлении и других сферах. Различным вопросам, связанным с данной задачей, посвящено большое число исследований, статей и монографий.

Задача о ранце формулируется следующим образом. Даны n предметов. Предмет i характеризуется весом w_i и ценой p_i. Требуется положить в ранец грузоподъемностью C набор предметов максимальной стоимости. Данное неформальное описание может быть математически записано следующим образом:

$$f(\overline{x}) = \sum_{i=1}^{n} p_i x_i \to \max;$$

$$\sum_{i=1}^{n} w_i x_i \leq C; \tag{4.5}$$

$$x_i \in \{0, 1\}, i = 1, \ldots, n.$$

При этом обычно предполагаются справедливыми неравенства $w_i \leq C$, $\sum_{i=1}^{n} w_i > C$. Функция $f(\overline{x})$, где через $\overline{x}$ обозначается набор $(x_1, \ldots, x_n)$, называется *целе-*

вой функцией для данной задачи.

Задачу о ранце можно решать путем полного перебора всех кортежей длиной n. При этом для решения задачи размерности n в наихудшем случае потребуется просмотреть 2^n кортежей. Сделать поиск решения более эффективным позволяет метод ветвей и границ. Этот метод заключается в последовательной декомпозиции исходной задачи на подзадачи, с отсевом подзадач, решение которых заведомо не приведет к нахождению оптимума исходной задачи. Общая схема метода ветвей и границ соответствует приведенной в разделе 4.1.

Дадим общее описание метода ветвей и границ [157–159] для решения задачи о ранце.

Для удобства описания метода ветвей и границ введем в рассмотрение более общую постановку задачи о ранце, в которой предполагается, что часть переменных принимают фиксированные значения:

$$\begin{gathered} \sum_{i=1}^{n} p_i x_i \to max; \\ \sum_{i=1}^{n} w_i x_i \leq C; \\ x_i = \theta_i, i \in I, \theta_i \in \{0,1\} \\ x_i \in \{0,1\}, i \in N \setminus I, \end{gathered} \tag{4.6}$$

где I — множество индексов всех переменных x_i, принимающих фиксированные значения θ_i. Введенная постановка позволяет описывать подзадачи, возникающие в процессе работы метода ветвей и границ. Отметим, что для задача (4.5) является частным случаем задачи (4.6), когда множество I пусто.

Задаче (4.6) соответствует следующая линейная *задача релаксации*:

$$\begin{gathered}\sum_{i=1}^{n} p_i x_i \to max; \\ \sum_{i=1}^{n} w_i x_i \le C; \\ x_i = \theta_i, i \in I, \theta_i \in \{0,1\} \\ 0 \le x_i \le 1, i \in N \setminus I.\end{gathered} \tag{4.7}$$

Оптимум задачи (4.7) не меньше оптимума задачи (4.6). Поэтому, если оптимальное решение задачи (4.7) достигается на целочисленном наборе значений $x_1, \ldots, x_n$, то этот набор является также оптимальным решением задачи (4.6).

Задача релаксации решается за линейное относительно числа переменных количество операций методом Данцига [157, 158]. Известно, что ее оптимальное решение достигается на наборе значений переменных $x_1^r, \ldots, x_n^r$, содержащем не более одного дробного (не целого) значения:

$$\begin{aligned} x_i^r &= 1, 1 \le i \le s-1, i \in N \setminus I, \\ x_s^r &= \frac{C - \sum_{i \in I} \theta_i w_i - \sum_{i \in \{1,\ldots,s-1\} \setminus I} w_i}{w_s}, \\ x_i^r &= 0, s+1 \le i \le N, i \in N \setminus I, \\ x_i^r &= \theta_i, i \in I. \end{aligned} \tag{4.8}$$

где индекс s *дробной* переменной x_s^r определяется по формуле

$$s = \min i : \sum_{j \in \{1,\ldots,i\} \setminus I} w_j > C - \sum_{j \in I} \theta_j w_j, \ i \in N \setminus I.$$

Оценка, получаемая с использованием релаксационной задачи, вычисляется по следующей формуле:

$$U_{LR} = \lfloor \sum_{i \in \{1,\ldots,s\} \setminus I} p_i + \sum_{i \in I} \theta_i p_i + p_s \frac{C - \sum_{i \in \{1,\ldots,s\} \setminus I} w_i - \sum_{i \in I} \theta_i w_i}{w_s} \rfloor. \tag{4.9}$$

Если же $\sum_{i \in N \setminus I} w_i \le C - \sum_{i \in I} \theta_i w_i$ и искомого s не существует, то решением задачи релаксации будет единичный набор, а $U_{LR} = \sum_{i \in N \setminus I} p_i + \sum_{i \in I} \theta_i p_i$.

В процессе работы алгоритма поддерживаются следующие данные: *рекорд*, т.е. наибольшее найденное на данный момент времени значение целевой функции f, *рекордное решение*, на котором достигается рекорд, и текущий список *подзадач*, на которые разбита исходная задача. Рекорд выполняет роль нижней оценки оптимума. Подзадачи из списка являются задачами вида (4.6).

Данные: рекорд f^0, рекордное решение x^0, список подзадач.

Шаг 1. В список подзадач помещается исходная задача. Рекорд полагается равным 0.

Шаг 2. Если список подзадач пуст, то алгоритм завершается. В противном случае выбирается подзадача P из списка подзадач. Подзадача P удаляется из списка.

Шаг 3. Проверяется, выполнены ли для выбранной подзадачи P указанные далее в работе *условия отсева*. Если одно из условий отсева выполнено, то осуществляется переход к шагу 2. При необходимости на этом шаге также обновляются рекорд и соответствующее рекордное решение.

Шаг 4. Выбранная подзадача подвергается декомпозиции. Для этого выбирается переменная x_b, называемая *переменной ветвления*. Подзадача P вида (4.6) разбивается на две подзадачи P_0 и P_1, получаемые присваиванием переменной x_b значений 0 и 1 соответственно:
подзадача P_0:

$$\sum_{i=1}^{n} p_i x_i \to max;$$

$$\sum_{i=1}^{n} w_i x_i \leq C;$$

$$x_i = \theta_i, i \in I, x_b = 0,$$

$$x_i \in \{0, 1\}, i \in N \setminus \{I \cup b\},$$

подзадача P_1:

$$\sum_{i=1}^{n} p_i x_i \to max;$$

$$\sum_{i=1}^{n} w_i x_i \leq C;$$

$$x_i = \theta_i, i \in I, x_b = 1,$$

$$x_i \in \{0, 1\}, i \in N \setminus \{I \cup b\},$$

Построенные подзадачи P_0 и P_1 помещаются в список подзадач и осуществляется переход к шагу 2.

На шаге 3 метода ветвей и границ вычисляются верхние и нижние оценки. Нижней оценкой является наибольшее найденное к данному шагу значение целевой функции (рекорд). Вычисление рекорда f_r проводится следующим образом. Перед началом расчетов полагаем значение рекорда равным нулю $f_r = 0$, которое достигается на нулевом рекордном решении $x_r = 0$. Если $\sum_{i \in I} \theta_i w_i + \sum_{j \in N \setminus I} w_i \leq C$, то вектор $\hat{x}$, где

$$\hat{x}_i = \begin{cases} \theta_i, i \in I, \\ 1, i \in N \setminus I, \end{cases}$$

будет допустимым решением задачи (4.10). Если $f(\hat{x}) \geq f_r$ то производится обновление рекорда $f_r := f(\hat{x}), x_r := \hat{x}$.

Результатом работы алгоритма является окончательное рекордное решение. Заметим, что работа описанного нами алгоритма существенным образом зависит от процедуры выбора очередной подзадачи из списка подзадач и процедуры выбора переменной ветвления для декомпозиции выбранной подзадачи. Алгоритм, для которого данные процедуры строго определены, будем называть *вариантом* метода ветвей и границ.

Исключение подзадач из дальнейшего рассмотрения (отсев) производится на

шаге 3 с помощью специальных *правил отсева*. В стандартном варианте метода ветвей и границ для задачи о ранце решается так называемая *оценочная задача*, которая позволяет получить верхнюю оценку для решения рассматриваемой подзадачи. В качестве оценочной часто выбирают линейную релаксацию задачи (4.7). Вариант МВГ будем называть *стандартным*, если условиями отсева выбранной подзадачи P вида (4.6) являются следующие условия:

1. **C1**: $\sum_{i \in I} \theta_i w_i > C$, т.е. подзадача P не имеет решения;
2. **C2**: Оценка U_{LR}, полученная с помощью задачи (4.7) не превосходит значение текущего рекорда, тем самым оптимальное значение целевой функции f для подзадачи P также заведомо не превосходит значение текущего рекорда.

Вариант МВГ будем называть *ослабленным*, если условиями отсева выбранной подзадачи P вида (4.6) являются условие **C1** и условие
C2′: $\sum_{i \in I} \theta_i w_i + \sum_{i \in N \setminus I} w_i \leq C$, т.е. оптимальным решением подзадачи P является набор значений $x_1, \ldots, x_n$, такой что $x_i = \theta_i$ при $i \in I$ и $x_i = 1$ при $i \in N \setminus I$.

Утверждение 4.1 *Если для некоторой подзадачи задачи (4.5) выполнено условие отсева* **C2′**, *то условие* **C2** *также выполнено для этой подзадачи.*

Доказательство. Пусть для подзадачи P выполнено условие отсева **C2′**. Тогда $\sum_{i \in I} \theta_i w_i + \sum_{i \in N \setminus I} w_i \leq C$. Из этого следует, что решение задачи x' релаксации для подзадачи P является целочисленным и совпадает с решением подзадачи P. Согласно правилу обновления рекорда, $f(x') \leq f_r$, поэтому данная подзадача может быть исключена из дальнейшего рассмотрения по правилу отсева **C2**. □

Декомпозиция, выполняемая на шаге 4 МВГ, состоит в разбиении исходной задачи на две путем присваивания одной из переменных значений 0 и 1 соответственно и называется *ветвлением* задачи по переменной. Наиболее рас-

пространенными способами выбора переменной для ветвления являются выбор первой переменной в соответствии с некоторым порядком или выбор дробной переменной в задаче релаксации. Пример первого подхода можно найти в работе [160], а второго — в работе [161]. Ветвление по дробной переменной считается стандартным и предлагается в качестве основного в некоторых учебных курсах [159].

Для получения оценок сложности метода ветвей и границ для задачи о ранце часто используется ее частный случай — задача о сумме подмножеств [157], формулируемая следующим образом:

$$\begin{cases} f(x) = \sum_{i=1}^{n} w_i x_i \to max, \\ \text{при условиях} \\ \sum_{i=1}^{n} w_i x_i \leq C, \end{cases} \tag{4.10}$$

где $x = (x_1, \ldots, x_n)$ — вектор булевых переменных.

В данной постановке через w_i обозначен вес i-го предмета, помещаемого в ранец грузоподъемностью C. Задача состоит в определении набора предметов максимального суммарного веса, который можно разместить в ранце.

В дальнейшем часто будут рассматриваться только такие варианты задачи о сумме подмножеств, в которых суммарный вес любой комбинации предметов не совпадает с грузоподъемностью C, т.е. следующее уравнение не имеет решений

$$\sum_{i=1}^{n} w_i x_i = C, \tag{4.11}$$

где $x = (x_1, \ldots, x_n)$ — вектор булевых переменных.

Для таких задач несложно заметить, что условия отсева C1-C2 сводятся к двум условиям C1 и C2'.

Утверждение 4.2 *Для любой подзадачи задачи 4.10, удовлетворяющей условию (4.11) условия отсева* $\mathbf{C2'}$ *и* **C2** *эквивалентны.*

Доказательство. Действительно, рассмотрим некоторую подзадачу задачи (4.10)

$$\sum_{i=1}^{n} w_i x_i \to max;$$

$$\sum_{i=1}^{n} w_i x_i \leq C; \tag{4.12}$$

$$x_i = \theta_i, i \in I, \theta_i \in \{0, 1\}$$

$$x_i \in \{0, 1\}, i \in N \setminus I,$$

и соответствующую ей задачу релаксации:

$$\sum_{i=1}^{n} w_i x_i \to max;$$

$$\sum_{i=1}^{n} w_i x_i \leq C; \tag{4.13}$$

$$x_i = \theta_i, i \in I, \theta_i \in \{0, 1\}$$

$$0 \leq x_i \leq 1, i \in N \setminus I.$$

Поскольку имеет место утверждение (4.1), то для установления справедливости утверждения достаточно доказать, что если **C2'** не выполнено, то **C2** также не выполняется. Если **C2'** не выполнено, то $\sum_{i \in I} \theta_i w_i + \sum_{i \in N \setminus I} w_i > C$. В этом случае оптимальное значение целевой функции в задаче (4.13) составляет C. Так как задача удовлетворяет условию (4.11), то значение оптимума, а следовательно и любого рекорда, меньше C. Таким образом, правило $C2$ не выполнено. Тем самым, утверждение доказано. □.

4.4 Асимптотическая оценка сложности наихудшего случая в методе ветвей и границ с ветвлением по дробной переменной для задачи о ранце

В работе [24] приводится пример следующей задачи:

$$f(\overline{x}) = \sum_{i=1}^{n} 2x_i \to max; \sum_{i=1}^{n} 2x_i \leq 2\lfloor \frac{n}{2} \rfloor + 1; x_i \in \{0,1\}, i = 1, \ldots, n, \qquad (4.14)$$

и показывается, что сложность решения этой задачи методом ветвей и границ при любом способе выбора очередной подзадачи и переменной для ветвления составляет

$$\Phi(n) = 2\begin{pmatrix} n+1 \\ \lfloor \frac{n}{2} \rfloor + 1 \end{pmatrix} - 1. \qquad (4.15)$$

Заметим, что выбор ценовых и весовых коэффициентов равными 2 в целевой функции задачи (4.14) на первый взгляд может показаться излишним усложнением. Более естественным представляется положить их равными 1. Однако, в таком случае, верхняя оценка $U_{LR} = \lfloor \lfloor \frac{n}{2} \rfloor + \frac{1}{2} \rfloor = \lfloor \frac{n}{2} \rfloor$ очевидно совпадает с рекордом и задача решается как только этот рекорд будет найден.

Задача (4.14) играет существенную роль в дальнейшем изложении. Будем называть ее *задачей Финкельштейна* в соответствии с фамилией автора монографии [24] и обозначать сложность ее решения через $\Phi(n)$.

Возникает вопрос о точности данной оценки. В работе [156] доказывается, что если для ветвления выбирается каждый раз переменная, соответствующая предмету с максимальным весом, то (4.15) будет точной верхней оценкой для сложности такого варианта МВГ. В данном разделе будет показано, что если для ветвления выбирается дробная переменная, то можно построить примеры, в которых сложность базового варианта метода ветвей и границ превзойдет $\Phi(n)$.

Рассматривается семейство задач о сумме подмножеств $P(T, m, k)$, где k — целое число, m — целое неотрицательное число, а $T = \{t_1, \ldots, t_n\} \in \mathbb{N}^n$ —

упорядоченный набор натуральных чисел длины n. Задача $P(T, m, k)$ имеет следующий вид:

$$\sum_{i=1}^{m+n} w_i x_i \to \max;$$

$$\sum_{i=1}^{m+n} w_i x_i \le ka + 1,$$

$$x_i \in \{0, 1\}, i = 1, \ldots, n,$$

где $a \in \mathbb{N}, a \ge 2, w_i = \begin{cases} a, \text{ если } 1 \le i \le m; \\ t_{n+m+1-i} \cdot a, \text{ если } m < i \le m + n. \end{cases}$

Заметим, что сложность решения задачи $P(T, m, k)$ не зависит от параметра a. Учитывая этот факт, мы обозначаем данную сложность через $S(T, m, k)$. Далее в данном разделе нам будет удобнее оценивать не общее число вершин в дереве ветвлений, а число концевых вершин $V(T, m, k)$. Очевидно,

$$S(T, m, k) = 2V(T, m, k) - 1. \tag{4.16}$$

Число концевых вершин в дереве ветвлений для задачи Финкельштейна обозначим через $V_\Phi(n)$. Очевидно $V_\Phi(n) = \binom{n+1}{\lfloor \frac{n}{2} \rfloor + 1}$.

4.4.1 Нахождение дробной переменной

Задаче о сумме подмножеств (4.10) соответствует следующая линейная задача релаксации:

$$\sum_{i=1}^{n} w_i x_i \to max;$$

$$\sum_{i=1}^{n} w_i x_i \le C; \tag{4.17}$$

$$0 \le x_i \le 1, i = 1, \ldots, n.$$

Оптимум задачи (4.17) не меньше оптимума задачи (4.10). Поэтому, если оптимальное решение задачи (4.17) достигается на целочисленном наборе значений

$x_1, \ldots, x_n$, то оно является также оптимальным решением задачи (4.10).

Задача (4.17) представляет собой одномерную задачу линейного программирования и может быть решена методом Данцига [159] следующим образом. Сначала определяется номер *s дробной переменной* по следующему правилу:

$$s = \min \left\{ j \in \{1, \ldots, n\} : \sum_{i=1}^{j} w_i > C \right\}$$

Если такого s не существует, т.е. $\sum_{i=1}^{n} w_i \leq C$, то решением задачи (4.17) является единичный набор значений $x_1, \ldots, x_n$. В противном случае решение задачи (4.17) задается следующим образом:

$$x_i = 1 \qquad \text{если} \quad i < s,$$

$$x_i = 0 \qquad \text{если} \quad s < i \leq n,$$

$$x_s = \frac{C - \sum_{i=1}^{s-1} w_i}{w_s}.$$

Определение дробной переменной и решение задачи релаксации требует линейного относительно числа n количества операций.

4.4.2 Рекуррентные соотношения для сложности метода ветвей и границ

Получим теперь рекуррентные соотношения для $V(T, m, k)$. Пусть $T = \{t_1, \ldots, t_n\}$. Обозначим через $|T|$ число элементов в наборе T, через $\emptyset$ — набор, не содержащий ни одного элемента, и через $T^{(i)}$ — набор $\{t_1, \ldots, t_{i-1}, t_{i+1}, \ldots, t_n\}$, полученный из T удалением i-го элемента. Выпишем следующие соотношения для $S(T, m, k)$ при различных k.

Базовые рекуррентные соотношения:

1. Пусть $k < 0$. В этом случае задача не имеет решения, поэтому $S(T, m, k) = 1$.

2. Пусть $0 \leq k < m$. Тогда дробная переменная имеет номер $l = k + 1 \leq m$.

При присваивании переменной x_l значения 0 получается подзадача $P(T, m-1, k)$, а при присваивании переменной x_l значения 1 — подзадача $P(T, m-1, k-1)$. Следовательно, $V(T, m, k) = V(T, m-1, k) + V(T, m-1, k-1)$.

3. Пусть[2] $m + \sum_{j=i+1}^{n} t_j \le k < m + \sum_{j=i}^{n} t_j$ для некоторого $i \in 1, \ldots, n$. Тогда при присваивании значений 0 и 1 дробной переменной x_{m+i} исходная задача $P(T, m, k)$ распадается на подзадачи $P(T^{(i)}, m, k)$ и $P(T^{(i)}, m, k-t_i)$ соответственно. Поэтому $V(T, m, k) = V(T^{(i)}, m, k) + V(T^{(i)}, m, k-t_i)$.

4. Пусть $m + \sum_{j=1}^{n} t_i \le k$. В этом случае задача $P(T, m, k)$ имеет целочисленное решение $x_1 = x_2 = \ldots = x_{m+n} = 1$, т. е. $V(T, m, k) = 1$.

Данные соотношения позволяют получить рекуррентную формулу, выражающую $V(T, m, k)$ через $V(T', m, k)$, где $|T'| = |T| - 1$. Рассмотрим сначала случай $|T| = 0$, т. е. $T = \emptyset$.

Утверждение 4.3 *Для всех m, k таких, что $k \ge -1$, $m \ge 0$, $m \ge k$, имеет место равенство*

$$V(\emptyset, m, k) = \binom{m+1}{k+1}. \tag{4.18}$$

Доказательство. Рекуррентные соотношения для $V(\emptyset, m, k)$ имеют следующий вид:

$$V(\emptyset, m, k) = \begin{cases} 1, \text{ если } k < 0; \\ V(\emptyset, m-1, k) + V(\emptyset, m-1, k-1), \text{ если } 0 \le k < m, m \ge 1; \\ 1, \text{ если } k = m. \end{cases} \tag{4.19}$$

Выписанные рекуррентные соотношения можно наглядно представить в виде треугольной таблицы, в ячейках которой стоят значения $V(\emptyset, m, k)$:

[2] Здесь и далее сумму по пустому множеству полагаем равной 0.

m \ k	-1	0	1	2	...
0	1	1			
1	1	2	1		
2	1	3	3	1	
3	1	4	6	4	1
⋮			...		

На границах данной таблицы ($k = -1$ и $m = k$) имеем $V(\emptyset, m, k) = 1$, а в остальных ячейках значения $V(\emptyset, m, k)$ вычисляются согласно правилу $V(\emptyset, m, k) = V(\emptyset, m-1, k) + V(\emptyset, m-1, k-1)$. Несложно заметить, что данной таблица идентична треугольнику Паскаля, поэтому $V(\emptyset, m, k) = \binom{m+1}{k+1}$. Утверждение доказано. □

В дальнейшем нам понадобится следующее обобщение треугольника Паскаля.

Определение. *Аддитивным треугольником* с вершиной (m_0, k_0) называется функция Δ, принимающая целые значения и заданная на множестве пар целых чисел $\{(m,k) | k \geq k_0, m > m_0, m - m_0 \geq k - k_0\}$, такая что $\Delta(m,k) = \Delta(m-1,k) + \Delta(m-1,k-1)$ для $k > k_0$ и $m - m_0 > k - k_0$. Множество $\{(m,k) | k = k_0, m > m_0\}$ называется *левой границей*, а множество $\{(m,k) | m - m_0 = k - k_0, k > k_0\}$ — *правой границей* треугольника Δ. Объединение левой и правой границ образует *границу* треугольника Δ. Множество пар $\{(m,k) | k > k_0, m - m_0 > k - k_0\}$ называется *внутренностью треугольника* Δ.

Аддитивные треугольники удобно изображать в виде треугольных таблиц. В следующей таблице представлен треугольник с вершиной (m_0, k_0), элементы границы выделены серым тоном:

m \ k	...	k_0	k_0+1	...	
$\vdots$	...	$\vdots$	$\vdots$		
m_0	...				
m_0+1	...	*	*		
m_0+2	...	*	*	*	
m_0+3	...	*	*	*	*
$\vdots$			...		

Над аддитивными треугольниками можно определить операции суммы и разности.

Определение. Пусть Δ' и Δ'' — аддитивные треугольники с вершиной (m_0, k_0). Функция Δ, такая что $\Delta(m,k) = \Delta'(m,k) + \Delta''(m,k)$ на множестве $\{(m,k)|k \geq k_0, m > m_0, m - m_0 \geq k - k_0\}$ называется *суммой* треугольников Δ' и Δ''. Будем в этом случае писать $\Delta = \Delta' + \Delta''$.

Аналогично определяется понятие *разности* двух аддитивных треугольников. Несложно показать, что если Δ' и Δ'' — аддитивные треугольники с вершиной (m_0, k_0), то $\Delta' - \Delta''$ и $\Delta' + \Delta''$ также удовлетворяют определению аддитивного треугольника с вершиной (m_0, k_0).

Заметим, что поскольку значения аддитивного треугольника на его внутренности однозначно определяются значениями на границе, то из выполнения соотношения $\Delta(m,k) = \Delta'(m,k) + \Delta''(m,k)$ на границе следует выполнение этого соотношения на всей области определения, т. е. $\Delta = \Delta' + \Delta''$. Это наблюдение позволяет проверять, что один треугольник является суммой или разностью двух других треугольников на основании сравнения значений треугольников на границе. Определим также понятие *подтреугольника*.

Определение. Пусть Δ — аддитивный треугольник с вершиной (m_0, k_0), тогда его *подтреугольником* Δ' с вершиной (m_1, k_1), $m_1 \geq m_0$, $k_1 \geq k_0$, называет-

ся аддитивный треугольник с вершиной (m_1, k_1) такой, что $\Delta'(m,k) = \Delta(m,k)$ при всех m, k таких, что $m > m_1$, $k \geq k_1$, $m - m_1 \geq k - k_1$.

Перейдем к рассмотрению рекуррентных соотношений для $V(T,m,k)$ в случае $|T| > 0$.

Утверждение 4.4 *Справедливы следующие соотношения:*
если $0 \leq k < \min(m, t_n - 1)$, *то*

$$V(T,m,k) = V(T^{(n)},m,k) + \binom{m}{k}; \tag{4.20}$$

если $t_n - 1 \leq k \leq m - 1$, *то*

$$\begin{aligned} V(T,m,k) \ &= V(T^{(n)},m,k) + V(T^{(n)},m,k-t_n) \\ &\quad + \tbinom{m}{k} - \tbinom{m-t_n+1}{k-t_n+1}; \end{aligned} \tag{4.21}$$

если существует такое i, $i \in \{1,\ldots,n\}$, *что* $m + \sum_{j=i+1}^{j=n} t_j \leq k < m + \sum_{j=i}^{j=n} t_j$, *то*

$$V(T,m,k) = V(T^{(i)},m,k) + V(T^{(i)},m,k-t_i). \tag{4.22}$$

Доказательство. Соотношение (4.22) является непосредственным следствием полученных ранее базовых рекуррентных соотношений. Докажем справедливость соотношений (4.20) и (4.21).

Рассмотрим рекуррентные соотношения для $V(T,m,k)$ при $|T| > 0$ и $0 \leq k < m$:

$$V(T,m,k) = \begin{cases} 1, \text{ если } k < 0; \\ V(T,m-1,k) + V(T,m-1,k-1), \text{ если } 0 \leq k < m; \\ V(T^{(n)},m,k) + V(T^{(n)},m,k-t_n), \text{ если } k = m. \end{cases} \tag{4.23}$$

Этим рекуррентным соотношениям соответствует аддитивный треугольник Δ_1 следующего вида:

m \ k	-1	0	1	2	...
0	1	$V(T^{(n)},0,0)+$ $V(T^{(n)},0,-t_n)$			
1	1		$V(T^{(n)},1,1)+$ $V(T^{(n)},1,1-t_n)$		
2	1			$V(T^{(n)},2,2)+$ $V(T^{(n)},2,2-t_n)$	
⋮			...		

Треугольник Δ_1 может быть представлен в виде суммы двух треугольников Δ_2 и Δ_3. Треугольник Δ_2 имеет следующий вид:

m \ k	-1	0	1	2	...
0	1	$V(T^{(n)},0,0)$			
1	1		$V(T^{(n)},1,1)$		
2	1			$V(T^{(n)},2,2)$	
⋮			...		

Этот треугольник представляет собой в точности треугольник для $V(T^{(n)},m,k)$, поэтому $\Delta_2(m,k) = V(T^{(n)},m,k)$. Треугольник Δ_3 имеет вид:

m \ k	-1	0	1	2	...
0	0	$V(T^{(n)},0,-t_n)$			
1	0		$V(T^{(n)},1,1-t_n)$		
2	0			$V(T^{(n)},2,2-t_n)$	
⋮			...		

Так как $V(T^{(n)},m,k) = 1$ при $k < 0$, то первые t_n ячеек правой границы содержат 1. Этот треугольник может быть разложен на сумму двух треугольников Δ_4 и Δ_5. Треугольник Δ_5 будет рассмотрен далее в статье, а треугольник Δ_4 имеет следующий вид:

m \ k	-1	0	1	…	t_n-2	t_n-1	t_n	…
0	0	1						
1	0		1					
⋮	⋮			⋱				
t_n-2	0		…		1			
t_n-1	0			…		0		
t_n	0			…			0	
⋮				…				

Левая граница этого треугольника представляет собой последовательность, состоящую из нулей. Первые $t_n - 1$ ячеек правой границы содержат единицы, а остальные содержат нули. Этот треугольник может быть представлен в виде разности двух треугольников: $\Delta_4 = \Delta_7 - \Delta_8$. Треугольник Δ_7 имеет вид:

m \ k	-1	0	1	…	t_n-2	t_n-1	t_n	…
0	0	1						
1	0		1					
⋮	⋮			⋱				
t_n-2	0		…		1			
t_n-1	0			…		1		
t_n	0			…			1	
⋮				…				

Левая граница этого треугольника представляет собой последовательность, состоящую из нулей, а правая граница состоит из единиц. Несложно заметить, что $\Delta_7(m,k) = \binom{m}{k}$ для $k \geq 0$. Треугольник Δ_8 имеет вид:

m \ k	-1	0	1	…	t_n-2	t_n-1	t_n	…
0	0	0						
1	0		0					
⋮	⋮			⋱				
t_n-2	0		…		0			
t_n-1	0		…			1		
t_n	0		…				1	
⋮				…				

Левая граница этого треугольника представляет собой последовательность, состоящую из нулей. Первые t_n-1 ячеек правой границы содержат нули, а остальные содержат единицы. Этот треугольник также представляет собой треугольник Паскаля, сдвинутый на t_n-1 вправо и вниз. Поэтому

$$\Delta_8(m,k)=\begin{cases}0, \text{ если } -1\le k<t_n-1;\\ \binom{m-t_n+1}{k-t_n+1}, \text{ если } t_n-1\le k.\end{cases}$$

Так как $\Delta_4=\Delta_7-\Delta_8$, то

$$\Delta_4(m,k)=\begin{cases}\binom{m}{k}, \text{ если } 0\le k<t_n-1;\\ \binom{m}{k}-\binom{m-t_n+1}{k-t_n+1}, \text{ если } t_n-1\le k.\end{cases}$$

Треугольник Δ_5 имеет следующий вид:

m \ k	-1	0	1	…	t_n-2	t_n-1	t_n	t_n+1	…
0	0	0							
1	0	0	0						
⋮	⋮		⋱						
t_n-2	0	⋯			0				
t_n-1	0	⋯			0	1			
t_n	0	⋯			0	1	$V(T^{(n)},t_n,0)$		
t_n+1	0	⋯			0	1		$V(T^{(n)},t_n+1,1)$	
⋮						⋯			

Левая граница треугольника Δ_5 содержит только нули. Правая граница является последовательностью вида:

$$\underbrace{0,\ldots,0}_{t_n-1},1,V(T^{(n)},t_n,0),V(T^{(n)},t_n+1,1),\ldots,V(T^{(n)},t_n+i,i),\ldots$$

Очевидно, что $\Delta_5(m,k)=0$ при $k\le t_n-2$, $\Delta_5(m,k)=1$ при $k=t_n-1$. Легко заметить, что подтреугольник с вершиной (t_n-1,t_n-1) треугольника Δ_5 является подтреугольником аддитивного треугольника для $V(T^{(n)},m,k)$ вершиной $(t_n-1,-1)$. Следовательно,

$$\Delta_5(m,k)=\begin{cases}0,\ \text{если}\ -1\le k<t_n-1;\\ V(T^{(n)},m,k-t_n),\ \text{если}\ t_n-1\le k.\end{cases}$$

Суммируя выражения, полученные для треугольников Δ_2,Δ_4 и Δ_5, получаем равенство

$$V(T,m,k)=\begin{cases}V(T^{(n)},m,k)+\binom{m}{k},\ \text{если}\ 0\le k<t_n-1;\\ V(T^{(n)},m,k)+V(T^{(n)},m,k-t_n)+\binom{m}{k}-\binom{m-t_n+1}{k-t_n+1},\\ \quad\text{если}\ t_n-1\le k<m.\end{cases}$$

Тем самым утверждение доказано. □

Утверждение 4.4 можно применять при получении точных и асимптотических формул для $V(T, m, k)$. Получение точных формул продемонстрируем на следующем примере.

Пример. Пусть требуется вычислить сложность решения задачи $P(\{2\}, m, k)$. Рассмотрим различные варианты значения k. Пусть $0 \le k < 2$. Согласно утверждению 4.4 имеем

$$V(\{2\}, m, k) = V(\emptyset, m, k) + \binom{m}{k}.$$

Используя утверждение 4.3, получаем

$$V(\{2\}, m, k) = \binom{m+1}{k+1} + \binom{m}{k}.$$

Пусть теперь $2 \le k < m$. Согласно утверждениям 1 и 2 получим

$$\begin{aligned} V(\{2\}, m, k) &= V(\emptyset, m, k) + V(\emptyset, m, k-2) + \binom{m}{k} - \binom{m-1}{k-1} \\ &= \binom{m+1}{k+1} + \binom{m+1}{k-1} + \binom{m-1}{k}. \end{aligned}$$

Если $m \le k < m+2$, то аналогичным образом получаем

$$V(\{2\}, m, k) = V(\emptyset, m, k) + V(\emptyset, m, k-2) = \binom{m+1}{k+1} + \binom{m+1}{k-1}.$$

Асимптотическая оценка сложности

Рассмотрим асимптотическое поведение функции $\max_k V(T, m, k)$ при $m \to \infty$. Для этого потребуется ряд вспомогательных утверждений.

Утверждение 4.5 *Пусть γ, γ' — вещественные числа такие, что $0 < \gamma' < \gamma \le \frac{1}{2}$. Тогда*

$$\lim_{m\to\infty} \frac{\binom{m}{\lfloor \gamma' m \rfloor}}{\binom{m}{\lfloor \gamma m \rfloor}} = \lim_{m\to\infty} \frac{\binom{m}{\lceil (1-\gamma') m \rceil}}{\binom{m}{\lceil (1-\gamma) m \rceil}} = 0. \tag{4.24}$$

Доказательство. Из соотношений

$$\frac{\binom{m}{\lfloor\gamma' m\rfloor}}{\binom{m}{\lfloor\gamma m\rfloor}} = \frac{\frac{m!}{(\lfloor\gamma' m\rfloor)!(m-\lfloor\gamma' m\rfloor)!}}{\frac{m!}{(\lfloor\gamma m\rfloor)!(m-\lfloor\gamma m\rfloor)!}} = \frac{\prod_{i=\lfloor\gamma' m\rfloor}^{\lfloor\gamma m\rfloor} i}{\prod_{i=m-\lfloor\gamma' m\rfloor}^{m-\lfloor\gamma m\rfloor} i}$$
$$= \prod_{i=0}^{i=\lfloor\gamma m\rfloor-\lfloor\gamma' m\rfloor} \frac{i+\lfloor\gamma' m\rfloor}{i+m-\lfloor\gamma m\rfloor} \le \prod_{i=0}^{i=\lfloor\gamma m\rfloor-\lfloor\gamma' m\rfloor} \frac{\lfloor\gamma m\rfloor}{m-\lfloor\gamma' m\rfloor}$$
$$\le \prod_{i=0}^{i=\lfloor\gamma m\rfloor-\lfloor\gamma' m\rfloor} \frac{\gamma m}{(1-\gamma')m} \le \left(\frac{\gamma}{1-\gamma'}\right)^{\lfloor\gamma m\rfloor-\lfloor\gamma' m\rfloor+1}$$

и $\frac{\gamma}{1-\gamma'} < 1$ вытекает, что

$$\lim_{m\to\infty} \frac{\binom{m}{\lfloor\gamma' m\rfloor}}{\binom{m}{\lfloor\gamma m\rfloor}} \le \lim_{m\to\infty} \left(\frac{\gamma}{1-\gamma'}\right)^{\lfloor\gamma m\rfloor-\lfloor\gamma' m\rfloor+1} = 0.$$

Аналогично показывается, что $\lim_{m\to\infty} \frac{\binom{m}{\lceil(1-\gamma')m\rceil}}{\binom{m}{\lceil(1-\gamma)m\rceil}} = 0$. □

Утверждение 4.6 *Пусть a и b — целые числа, γ — вещественное число такое, что $0 < \gamma < 1$. Тогда*

$$\lim_{m\to\infty} \frac{\binom{m+b}{\lfloor\gamma m\rfloor+a}}{\binom{m}{\lfloor\gamma m\rfloor}} = \frac{(1-\gamma)^{a-b}}{\gamma^a}. \tag{4.25}$$

Доказательство. Обозначим через $C(a,b)$ выражение, стоящее под знаком предела. Преобразуем это выражение:

$$C(a,b) = \frac{\binom{m+b}{\lfloor\gamma m\rfloor+a}}{\binom{m}{\lfloor\gamma m\rfloor}} = \frac{\frac{(m+b)!}{(\lfloor\gamma m\rfloor+a)!(m-\lfloor\gamma m\rfloor+b-a)!}}{\frac{m!}{\lfloor\gamma m\rfloor!(m-\lfloor\gamma m\rfloor)!}}$$
$$= \frac{(m+b)!}{m!} \cdot \frac{\lfloor\gamma m\rfloor!}{(\lfloor\gamma m\rfloor+a)!} \cdot \frac{(m-\lfloor\gamma m\rfloor)!}{(m-\lfloor\gamma m\rfloor+b-a)!)}.$$

Преобразуем первый сомножитель:

$$\frac{(m+b)!}{m!} = \begin{cases} \prod_{i=1}^{b}(m+i), \text{ если } b \ge 0; \\ \frac{1}{\prod_{i=b+1}^{0}(m+i)}, \text{ если } b < 0. \end{cases}$$

Легко заметить, при любом b справедливо соотношение

$$\lim_{m\to\infty} \frac{\frac{(m+b)!}{m!}}{m^b} = 1.$$

Будем писать $f(m) \sim g(m)$, если $\lim_{m\to\infty} \frac{f(m)}{g(m)} = 1$. Тогда $\frac{(m+b)!}{m!} \sim m^b$. Аналогично показывается справедливость следующих соотношений:

$$\frac{\lfloor \gamma m \rfloor !}{(\lfloor \gamma m \rfloor + a)!} \sim \lfloor \gamma m \rfloor^{-a} \sim (\gamma m)^{-a},$$
$$\frac{(m - \lfloor \gamma m \rfloor)!}{(m - \lfloor \gamma m \rfloor + b - a)!} \sim (m - \lfloor \gamma m \rfloor)^{a-b} \sim ((1-\gamma)m)^{a-b}.$$

Следовательно,

$$\begin{aligned} C(a,b) &\sim m^b \cdot (\gamma m)^{-a} \cdot ((1-\gamma)m)^{a-b} \\ &= m^{b-a+a-b} \cdot \frac{(1-\gamma)^{a-b}}{\gamma^a} = \frac{(1-\gamma)^{a-b}}{\gamma^a}. \end{aligned}$$

□

В случае $\gamma = \frac{1}{2}$ формула (4.25) приобретает более простой вид:

$$\lim_{m\to\infty} \frac{\binom{m+b}{\lfloor \frac{m}{2} \rfloor + a}}{\binom{m}{\lfloor \frac{m}{2} \rfloor}} = 2^b. \tag{4.26}$$

Покажем, каким образом соотношение (4.26) может быть применено для асимптотического анализа сложности задачи из примера 2 при $k = \lfloor \frac{m}{2} \rfloor$. Для этого, учитывая соотношение (4.26), сравним ее сложность с числом $\binom{m}{\lfloor \frac{m}{2} \rfloor}$:

$$\begin{aligned} \lim_{m\to\infty} \frac{V(\{2\}, m, \lfloor \frac{m}{2} \rfloor)}{\binom{m}{\lfloor \frac{m}{2} \rfloor}} &= \frac{\binom{m+1}{\lfloor \frac{m}{2} \rfloor + 1} + \binom{m+1}{\lfloor \frac{m}{2} \rfloor - 1} + \binom{m-1}{\lfloor \frac{m}{2} \rfloor}}{\binom{m}{\lfloor \frac{m}{2} \rfloor}} \\ &= 2 + 2 + 1/2 = 9/2. \end{aligned}$$

Задача Финкельштейна с тем же количеством переменных имеет сложность $V_\Phi(m+1) = \binom{m+2}{\lfloor \frac{m+1}{2} \rfloor + 1}$. Используя снова соотношение (4.26), можно показать, что $\lim_{m\to\infty} \frac{\binom{m+2}{\lfloor \frac{m+1}{2} \rfloor + 1}}{\binom{m}{\lfloor \frac{m}{2} \rfloor}} = 4$. Таким образом, $V(\{2\}, m, \lfloor \frac{m}{2} \rfloor) \sim \frac{9}{8} V_\Phi(m+1)$. Другими словами, сложность задачи $P(\{2\}, m, \lfloor \frac{m}{2} \rfloor)$ равна асимптотически $\frac{9}{8}$ сложности задачи Финкельштейна, имеющей одинаковое с $P(\{2\}, m, \lfloor \frac{m}{2} \rfloor)$ число число переменных.

Утверждение 4.7 *Для любого набора $T = \{t_1, \ldots, t_n\}$ натуральных чисел и*

любого целого t справедливо соотношение

$$\lim_{m\to\infty}\frac{V(T,m,\lfloor\frac{m}{2}\rfloor+t)}{\binom{m+n}{\lfloor\frac{m}{2}\rfloor}}=2+\sum_{i=1}^{n}\left(\frac{1}{2^i}-\frac{1}{2^{i+t_i-1}}\right).$$

Доказательство. Проведем доказательство методом математической индукции по длине n набора T. Пусть $n=0$. Согласно утверждению 1 имеем $V(\emptyset,m,\lfloor\frac{m}{2}\rfloor+t)=\binom{m+1}{\lfloor\frac{m}{2}\rfloor+t+1}$. Поэтому согласно формуле (4.26) получаем

$$\lim_{m\to\infty}\frac{V(\emptyset,m,\lfloor\frac{m}{2}\rfloor+t)}{\binom{m}{\lfloor\frac{m}{2}\rfloor}}=\lim_{m\to\infty}\frac{\binom{m+1}{\lfloor\frac{m}{2}\rfloor+t+1}}{\binom{m}{\lfloor\frac{m}{2}\rfloor}}=2.$$

Предположим теперь, что доказываемое утверждение справедливо для всех наборов $|T|$ длины $n-1$. Пусть $|T|=n$. Заметим, что найдется M такое, что при $m>M$ выполняется $t_n-1\le\lfloor\frac{m}{2}\rfloor+t\le m-1$. Поэтому согласно формуле (4.21) имеем

$$\frac{V(T,m,\lfloor\frac{m}{2}\rfloor+t)}{\binom{m+n}{\lfloor\frac{m}{2}\rfloor}}=\frac{1}{\binom{m+n}{\lfloor\frac{m}{2}\rfloor}}\cdot\left(V(T^{(n)},m,\lfloor\frac{m}{2}\rfloor+t)\right.$$
$$\left.+V(T^{(n)},m,\lfloor\frac{m}{2}\rfloor+t-t_n)+\binom{m}{\lfloor\frac{m}{2}\rfloor+t}-\binom{m-t_n+1}{\lfloor\frac{m}{2}\rfloor+t-t_n+1}\right).$$

Согласно предположению индукции и формуле (4.26) получаем

$$\lim_{m\to\infty}\frac{V(T^{(n)},m,\lfloor\frac{m}{2}\rfloor+t)}{\binom{m+n}{\lfloor\frac{m}{2}\rfloor}}=\lim_{m\to\infty}\frac{\binom{m+n-1}{\lfloor\frac{m}{2}\rfloor}}{\binom{m+n}{\lfloor\frac{m}{2}\rfloor}}\cdot\lim_{m\to\infty}\frac{V(T^{(n)},m,\lfloor\frac{m}{2}\rfloor+t)}{\binom{m+n-1}{\lfloor\frac{m}{2}\rfloor}}$$
$$=\frac{1}{2}\cdot\left(2+\sum_{i=1}^{n-1}\left(\frac{1}{2^i}-\frac{1}{2^{i+t_i-1}}\right)\right).$$

Аналогично показывается справедливость соотношения

$$\lim_{m\to\infty}\frac{V(T^{(n)},m,\lfloor\frac{m}{2}\rfloor+t-t_n)}{\binom{m+n}{\lfloor\frac{m}{2}\rfloor}}=\frac{1}{2}\cdot\left(2+\sum_{i=1}^{n-1}\left(\frac{1}{2^i}-\frac{1}{2^{i+t_i-1}}\right)\right).$$

С помощью формулы (4.26) можно также установить справедливость равенства

$$\lim_{m\to\infty}\frac{\binom{m}{\lfloor\frac{m}{2}\rfloor+t}-\binom{m-t_n+1}{\lfloor\frac{m}{2}\rfloor+t-t_n+1}}{\binom{m+n}{\lfloor\frac{m}{2}\rfloor}}=\frac{1}{2^n}-\frac{1}{2^{n+t_n-1}}.$$

Суммируя полученные выражения, получаем

$$\lim_{m\to\infty}\frac{V(T,m,\lfloor\frac{m}{2}\rfloor+t)}{\binom{m+n}{\lfloor\frac{m}{2}\rfloor}}=2+\sum_{i=1}^{n-1}\left(\frac{1}{2^i}-\frac{1}{2^{i+t_i-1}}\right)+\frac{1}{2^n}-\frac{1}{2^{n+t_n-1}}$$
$$=2+\sum_{i=1}^{n}\left(\frac{1}{2^i}-\frac{1}{2^{i+t_i-1}}\right).$$

□

Утверждение 4.8 *Для любого набора $T=\{t_1,\ldots,t_n\}$ натуральных чисел, любого вещественного γ, $0<\gamma<\frac{1}{2}$, и любого целого t справедливо соотношение*

$$\lim_{m\to\infty}\frac{\max_{\gamma m\le k\le(1-\gamma)m}V(T,m,k+t)}{\binom{m+n}{\lfloor\frac{m}{2}\rfloor}}=2+\sum_{i=1}^{n}\left(\frac{1}{2^i}-\frac{1}{2^{i+t_i-1}}\right).$$

Доказательство. Проведем доказательство методом математической индукции по длине n набора T. Пусть $n=0$. Согласно утверждению 4.3 имеем

$$\max_{\gamma m\le k\le(1-\gamma)m}V(\emptyset,m,k+t)=\max_{\gamma m\le k\le(1-\gamma)m}\binom{m+1}{k+t+1}\le\binom{m+1}{\lfloor\frac{m+1}{2}\rfloor}.$$

С другой стороны, найдется M такое, что при $m>M$ выполнено неравенство $\gamma m\le\lfloor\frac{m}{2}\rfloor\le(1-\gamma)m$. Таким образом, для всех $m>M$ выполнено

$$V(\emptyset,m,\lfloor\frac{m}{2}\rfloor+t)\le\max_{\gamma m\le k\le(1-\gamma)m}V(\emptyset,m,k+t).$$

Согласно утверждению 5 имеем $\lim_{m\to\infty}\frac{V(\emptyset,m,\lfloor\frac{m}{2}\rfloor+t)}{\binom{m}{\lfloor\frac{m}{2}\rfloor}}=2$. Согласно формуле (4.26) получаем $\lim_{m\to\infty}\frac{\binom{m+1}{\lfloor\frac{m+1}{2}\rfloor}}{\binom{m}{\lfloor\frac{m}{2}\rfloor}}=2$. Следовательно,

$$\lim_{m\to\infty}\frac{\max_{\gamma m\le k\le(1-\gamma)m}V(\emptyset,m,k+t)}{\binom{m}{\lfloor\frac{m}{2}\rfloor}}=2.$$

Тем самым утверждение доказано для случая $n=0$.

Предположим теперь, что утверждение справедливо для всех наборов T длины $n-1$. Рассмотрим набор T длины n. Обозначим $\frac{\max_{\gamma m\le k\le(1-\gamma)m}V(T,m,k+t)}{\binom{m+n}{\lfloor\frac{m}{2}\rfloor}}$ через $U(m)$. Заметим, что существует M' такое, что при $m>M'$ из $\gamma m\le k\le(1-\gamma)m$ вытекает, что $t_n-1\le k+t\le m-1$. Поэтому при $m>M'$ мы можем оценить

$U(m)$ сверху согласно формуле (4.21):

$$U(m) \le U_1(m) + U_2(m) + U_3(m),$$

где

$$U_1(m) = \frac{\max_{\gamma m \le k \le (1-\gamma)m} V(T^{(n)}, m, k+t)}{\binom{m+n}{\lfloor \frac{m}{2} \rfloor}},$$

$$U_2(m) = \frac{\max_{\gamma m \le k \le (1-\gamma)m} V(T^{(n)}, m, k+t-t_n)}{\binom{m+n}{\lfloor \frac{m}{2} \rfloor}},$$

$$U_3(m) = \frac{\max_{\gamma m \le k \le (1-\gamma)m} \left(\binom{m}{k+t} - \binom{m-t_n+1}{k+t-t_n+1} \right)}{\binom{m+n}{\lfloor \frac{m}{2} \rfloor}}.$$

Покажем существование предела $\lim_{m\to\infty} U_1(m)$. Так как $|T^{(n)}| = n-1$, то согласно предположению индукции и формуле (4.26) получаем

$$\lim_{m\to\infty} U_1(m) = \left(2 + \sum_{i=1}^{n-1} \frac{1}{2^i} - \frac{1}{2^{i+t_i-1}}\right) \cdot \lim_{m\to\infty} \frac{\binom{m+n-1}{\lfloor \frac{m}{2} \rfloor}}{\binom{m+n}{\lfloor \frac{m}{2} \rfloor}}$$

$$= \frac{1}{2}\left(2 + \sum_{i=1}^{n-1} \left(\frac{1}{2^i} - \frac{1}{2^{i+t_i-1}}\right)\right).$$

Аналогично доказывается существование предела $\lim_{m\to\infty} U_2(m)$:

$$\lim_{m\to\infty} U_2(m) = \frac{1}{2}\left(2 + \sum_{i=1}^{n-1} \left(\frac{1}{2^i} - \frac{1}{2^{i+t_i-1}}\right)\right).$$

Из очевидного равенства $\binom{m}{k+t} - \binom{m-t_n+1}{k+t-t_n+1} = \sum_{i=1}^{t_n-1} \binom{m-i}{k+t-i+1}$ следует, что

$$U_3(m) = \frac{\max_{\gamma m \le k \le (1-\gamma)m} \left(\binom{m}{k+t} - \binom{m-t_n+1}{k+t-t_n+1} \right)}{\binom{m+n}{\lfloor \frac{m}{2} \rfloor}}$$

$$\le \sum_{i=1}^{t_n-1} \frac{\max_{\max_{\gamma m \le k \le (1-\gamma)m} \binom{m-i}{k+t-i+1}}}{\binom{m+n}{\lfloor \frac{m}{2} \rfloor}}.$$

Обозначим сумму $\sum_{i=1}^{t_n-1} \frac{\max_{\max_{\gamma m \le k \le (1-\gamma)m} \binom{m-i}{k+t-i+1}}}{\binom{m+n}{\lfloor \frac{m}{2} \rfloor}}$ через $U_3'(m)$, а сумму $U_1(m) + U_2(m) + U_3'(m)$ через $U'(m)$. Очевидно, что $U(m) \le U'(m)$. В силу формулы (4.26) справедливо соотношение

$$\lim_{m\to\infty} \frac{\max_{\gamma m \le k \le (1-\gamma)m} \binom{m-i}{k+t-i+1}}{\binom{m+n}{\lfloor \frac{m}{2} \rfloor}} = \lim_{m\to\infty} \frac{\binom{m-i}{\lfloor \frac{m-i+1}{2} \rfloor}}{\binom{m+n}{\lfloor \frac{m}{2} \rfloor}} = \frac{1}{2^{n+i}}.$$

Следовательно,

$$\lim_{m\to\infty} U_3'(m) = \sum_{i=1}^{t_n-1} \frac{1}{2^{n+i}} = \frac{1}{2^n} - \frac{1}{2^{n+t_n-1}}.$$

Суммируя выражения для $\lim_{m\to\infty} U_1(m)$, $\lim_{m\to\infty} U_2(m)$ и $\lim_{m\to\infty} U_3'(m)$, получим

$$\begin{aligned}\lim_{m\to\infty} U'(m) &= 2 + \sum_{i=1}^{n-1}\left(\frac{1}{2^i} - \frac{1}{2^{i+t_i-1}}\right) + \frac{1}{2^n} - \frac{1}{2^{n+t_n-1}} \\ &= 2 + \sum_{i=1}^{n}\left(\frac{1}{2^i} - \frac{1}{2^{i+t_i-1}}\right).\end{aligned}$$

Заметим, что существует M'' такое, что при $m > M''$ выполняется $\gamma m < \lfloor \frac{m}{2} \rfloor < (1-\gamma)m$. Поэтому

$$U(m) = \frac{\max_{\gamma m \le k \le (1-\gamma)m} V(T,m,k)}{\binom{m+n}{\lfloor \frac{m}{2} \rfloor}} \ge \frac{V(T,m,\lfloor \frac{m}{2} \rfloor + t)}{\binom{m+n}{\lfloor \frac{m}{2} \rfloor}}.$$

Обозначим выражение $\frac{V(T,m,\lfloor \frac{m}{2} \rfloor + t)}{\binom{m+n}{\lfloor \frac{m}{2} \rfloor}}$ через $U''(m)$. Согласно утверждению 5 имеем

$$\lim_{m\to\infty} U''(m) = 2 + \sum_{i=1}^{n}\left(\frac{1}{2^i} - \frac{1}{2^{i+t_i-1}}\right).$$

Таким образом,

$$\lim_{m\to\infty} U'(m) = \lim_{m\to\infty} U''(m) = 2 + \sum_{i=1}^{n}\left(\frac{1}{2^i} - \frac{1}{2^{i+t_i-1}}\right)$$

и для $m > \max(M', M'')$ выполняется $U''(m) \le U(m) \le U'(m)$. Следовательно,

$$\lim_{m\to\infty} U(m) = 2 + \sum_{i=1}^{n}\left(\frac{1}{2^i} - \frac{1}{2^{i+t_i-1}}\right).$$

□

Утверждение 4.9 *Для любого набора $T = \{t_1, \ldots, t_n\}$ натуральных чисел, любого вещественного γ, $0 < \gamma < \frac{1}{2}$, и любого целого t справедливо соотношение*

$$\lim_{m\to\infty} \frac{\max_{k\in[0,\gamma m]\cup[(1-\gamma)m,\infty)} V(T,m,k+t)}{\binom{m+n}{\lfloor \frac{m}{2} \rfloor}} = 0.$$

Доказательство. Проведем доказательство методом математической индукции по длине n набора T. Пусть $n = 0$. Выберем произвольное γ', удовлетворяющее соотношению $0 < \gamma < \gamma' < \frac{1}{2}$. Тогда найдется M такое, что для любого $m > M$ выполнено $\gamma' m \le \lfloor \frac{m}{2} \rfloor \le (1-\gamma')m$ и $k + t \in [0, \gamma' m] \cup [(1-\gamma')m, \infty)$. Поэтому при $m > M$ согласно формуле (4.18) получим

$$\begin{aligned}
&\frac{\max_{k \in [0,\gamma m] \cup [(1-\gamma)m,\infty)} V(\emptyset, m, k+t)}{\binom{m}{\lfloor \frac{m}{2} \rfloor}} \\
&\le \frac{\max_{k \in [0,\gamma' m] \cup [(1-\gamma')m,\infty]} \binom{m+1}{k+1}}{\binom{m}{\lfloor \frac{m}{2} \rfloor}} \\
&\le \frac{\max\left(\binom{m+1}{\lfloor \gamma' m \rfloor + 1}, \binom{m+1}{\lfloor (1-\gamma')m \rfloor - 1} \right)}{\binom{m}{\lfloor \frac{m}{2} \rfloor}} \\
&= \max\left(\frac{\binom{m+1}{\lfloor \gamma' m \rfloor + 1}}{\binom{m}{\lfloor \frac{m}{2} \rfloor}}, \frac{\binom{m+1}{\lfloor (1-\gamma')m \rfloor - 1}}{\binom{m}{\lfloor \frac{m}{2} \rfloor}} \right).
\end{aligned}$$

Применяя формулу (4.24), несложно показать, что

$$\lim_{m\to\infty} \frac{\binom{m+1}{\lfloor \gamma' m \rfloor + 1}}{\binom{m}{\lfloor \frac{m}{2} \rfloor}} = \lim_{m\to\infty} \frac{\binom{m+1}{\lfloor (1-\gamma')m \rfloor - 1}}{\binom{m}{\lfloor \frac{m}{2} \rfloor}} = 0.$$

Следовательно,

$$\lim_{m\to\infty} \frac{\max_{k \in [0,\gamma m] \cup [(1-\gamma)m,\infty)} V(\emptyset, m, k)}{\binom{m}{\lfloor \frac{m}{2} \rfloor}} = 0.$$

Пусть утверждение доказано для любого набора T длины $n-1$. Докажем справедливость утверждения при $|T| = n$. Из соотношений (4.20 – 4.22) следует, что для некоторого i, $1 \le i \le n$, справедливо соотношение

$$V(T, m, k+t) \le V(T^{(i)}, m, k+t) + V(T^{(i)}, m, k+t-t_i) + \binom{m}{k+t}. \qquad (4.27)$$

Так как $|T^{(i)}| = n-1$, то согласно предположению индукции и формуле (4.26) получаем

$$\begin{aligned}
&\lim_{m\to\infty} \frac{\max_{k \in [0,\gamma m] \cup [(1-\gamma)m,\infty)} V(T^{(i)}, m, k+t)}{\binom{m+n}{\lfloor \frac{m}{2} \rfloor}} \\
&= \frac{1}{2} \cdot \lim_{m\to\infty} \frac{\max_{k \in [0,\gamma m] \cup [(1-\gamma)m,\infty)} V(T^{(i)}, m, k+t)}{\binom{m+n-1}{\lfloor \frac{m}{2} \rfloor}} = 0.
\end{aligned} \qquad (4.28)$$

Аналогично показывается, что

$$\lim_{m\to\infty} \frac{\max_{k\in[0,\gamma m]\cup[(1-\gamma)m,\infty)} V(T^{(i)},m,k+t-t_i)}{\binom{m+n}{\lfloor\frac{m}{2}\rfloor}} = 0. \tag{4.29}$$

Аналогично случаю $n = 0$ доказывается справедливость соотношения

$$\lim_{m\to\infty} \frac{\max_{k\in[0,\gamma m]\cup[(1-\gamma)m,\infty)} \binom{m}{k+t}}{\binom{m+n}{\lfloor\frac{m}{2}\rfloor}} = 0. \tag{4.30}$$

Из соотношений (4.27 – 4.30) следует, что

$$\lim_{m\to\infty} \frac{\max_{k\in[0,\gamma m]\cup[(1-\gamma)m,\infty)} V(T,m,k)}{\binom{m+n}{\lfloor\frac{m}{2}\rfloor}} = 0.$$

Box

Следующая теорема является непосредственным следствием утверждений 5, 6 и 7.

Теорема 4.2 *Для любого набора $T = \{t_1, \ldots, t_n\}$ натуральных чисел справедливо соотношение*

$$\lim_{m\to\infty} \frac{\max_k V(T,m,k)}{\binom{m+n}{\lfloor\frac{m}{2}\rfloor}} = 2 + \sum_{i=1}^{n} \left(\frac{1}{2^i} - \frac{1}{2^{i+t_i-1}} \right). \tag{4.31}$$

Формула (4.31) может быть использована непосредственно для вычисления асимптотической сложности $\max_k V(T,m,k)$ при $m \to \infty$ для заданного набора T. В частности, для задачи из примера , используя формулу (4.31), получим

$$\max_k V(\{2\},m,k) \sim \left(2 + \frac{1}{2} - \frac{1}{4}\right) \binom{m+1}{\lfloor\frac{m}{2}\rfloor} = \frac{9}{4} \binom{m+1}{\lfloor\frac{m}{2}\rfloor}.$$

Теорема 4.2 позволяет получить

Следствие 4.1 *Справедливы следующие утверждения:*

1. *для любого набора T натуральных чисел*

$$\lim_{m\to\infty} \frac{\max_k V(T,m,k)}{\binom{m+n}{\lceil\frac{m}{2}\rceil}} \le 3, \text{ где } n = |T|;$$

2. *для любого $\epsilon > 0$ существует набор T натуральных чисел такой, что*

$$\lim_{m\to\infty} \frac{\max_k V(T,m,k)}{\binom{m+n}{\lceil\frac{m}{2}\rceil}} \ge 3 - \epsilon, \text{ где } n = |T|.$$

Доказательство. Справедливость первого утверждения устанавливается следующим образом:

$$\lim_{m\to\infty}\frac{\max_k V(T,m,k)}{\binom{m+n}{[\frac{m}{2}]}} = 2+\sum_{i=1}^{n}\left(\frac{1}{2^i}-\frac{1}{2^{i+t_i-1}}\right) \le 2+\left(1-\frac{1}{2^n}\right) \le 3.$$

Докажем второе утверждение. Пусть $M = 1+\log_2(\frac{1}{\epsilon})$. Положим $n = M$, $t_1 = \cdots = t_n = M$. Тогда

$$\lim_{m\to\infty}\frac{\max_k V(T,m,k)}{\binom{m+n}{[\frac{m}{2}]}} = 2+\sum_{i=1}^{n}\left(\frac{1}{2^i}-\frac{1}{2^{i+t_i-1}}\right)$$
$$= 2+\left(1-\frac{1}{2^M}\right)^2 \ge 3-\frac{1}{2^{M-1}} = 3-\epsilon.$$

Следствие доказано.□

Заметим, что $\lim_{m\to\infty}\frac{V_\Phi(m+n)}{\binom{m+n}{[\frac{m}{2}]}} = 2$. Поэтому содержательно следствие 1 означает, что среди задач рассматриваемого класса не существует примеров, сложность решения которых асимптотически больше, чем $\frac{3}{2}$ сложности решения задачи Финкельштейна с тем же числом переменных, при этом существуют задачи, сложность решения которых как угодно близко приближается асимптотически к $\frac{3}{2}$ сложности решения задачи Финкельштейна с тем же числом переменных.

4.5 Общие оценки сложности метода ветвей и границ для задачи о ранце

В данном разделе для задачи о ранце получены две верхние оценки сложности ее решения методом ветвей и границ, зависящие от входных данных задачи. Выделен частный случай, когда сложность решения задачи о ранце методом ветвей и границ полиномиально ограничена размерностью задачи. Получены также верхняя и нижняя оценки сложности решения методом ветвей и границ задачи о сумме подмножеств, являющейся важным частным случаем задачи о ранце.

Свойства би-деревьев

Установим некоторые свойства деревьев определенного вида, которые будут необходимы в дальнейшем для получения оценок трудоемкости метода ветвей и границ.

Би-деревом будем называть ориентированное бинарное дерево с корнем, удовлетворяющее следующим свойствам:

1. каждая дуга помечена 0 или 1;
2. из каждой внутренней (в том числе корневой) вершины исходит ровно две дуги, помеченные 0 и 1 соответственно;
3. в каждую вершину, отличную от корневой, входит ровно одна дуга.

Пусть a — вершина би-дерева T, Поддерево S с корнем a би-дерева T будем называть *би-поддеревом с корнем a би-дерева T*, если S является би-деревом. Если a при этом является корнем дерева T, то будем называть S *главным би-поддеревом би-дерева T*.

Пусть a — вершина би-дерева T. Би-поддерево S с корнем a би-дерева T будем называть *максимальным*, если любое би-поддерево с корнем a би-дерева T является также би-поддеревом дерева S.

Определим следующие подмножества вершин би-дерева:
$V(T)$ — множество вершин би-дерева T;
$V_*(T)$ — множество концевых вершин би-дерева T;
$V_*^0(T)$ — множество концевых вершин би-дерева T, в которые входит дуга, помеченная 0;
$V_*^1(T)$ — множество концевых вершин би-дерева T, в которые входит дуга, помеченная 1.
Введем также следующие обозначения для мощностей перечисленных множеств: $v(T) = |V(T)|$, $v_*(T) = |V_*(T)|$, $v_*^0(T) = |V_*^0(T)|$, $v_*^1(T) = |V_*^1(T)|$. Если a —

некоторая вершина би-дерева T, то последовательность дуг, соединяющую корень дерева T с этой вершиной, будем обозначать через $\pi(a)$. Число дуг в этой последовательности, помеченных 0, обозначим через $e_0(a)$, а число дуг, помеченных 1, — через $e_1(a)$.

Будем называть (c_0, c_1)-*деревом* би-дерево T, такое что

1. $e_0(x) = c_0$ для всех вершин x, принадлежащих $V_*^0(T)$;

2. $e_1(x) = c_1$ для всех вершин x, принадлежащих $V_*^1(T)$.

Заметим, что (c_0, c_1)-дерево можно построить для любых натуральных c_0, c_1.

Утверждение 4.10 *Пусть T — (c_0, c_1)-дерево. Тогда $v_*(T) = \binom{c_0+c_1}{c_0}$.*

Доказательство. Доказательство проведем индукцией по величине $c_0 + c_1$. В случае, когда $c_0 = 1$ или $c_1 = 1$, утверждение очевидно. Пусть для некоторого натурального $c > 3$ утверждение справедливо для всех (c_0, c_1)-деревьев таких, что $c_0 + c_1 < c$. Рассмотрим произвольное (c_0, c_1)-дерево T такое, что $c_0, c_1 > 1$ и $c_0 + c_1 = c$. Пусть a_0 и a_1 — вершины, соединенные с корнем a дерева T дугами, помеченными 0 и 1 соответственно (рис. 4.1). Обозначим через T_0 и T_1 максимальные би-поддеревья би-дерева T с корнями a_0 и a_1 соответственно. Нетрудно заметить, что дерево T_0 является $(c_0 - 1, c_1)$-деревом, а дерево T_1 является $(c_0, c_1 - 1)$-деревом. Поэтому, согласно индуктивному предположению, имеем $v_*(T_0) = \binom{c_0+c_1-1}{c_0-1}$, $v_*(T_1) = \binom{c_0+c_1-1}{c_0}$. Следовательно, $v_*(T) = v_*(T_0) + v_*(T_1) = \binom{c_0+c_1-1}{c_0-1} + \binom{c_0+c_1-1}{c_0} = \binom{c_0+c_1}{c_0}$. Тем самым утверждение доказано.□

Утверждение 4.11 *Пусть T — би-дерево, для которого справедливы следующие соотношения:*

1. $e_0(x) \geq c_0$ *для всех вершин x, принадлежащих $V_*^0(T)$;*

2. $e_1(x) \geq c_1$ *для всех вершин x, принадлежащих $V_*^1(T)$.*

Тогда $v_*(T) \geq \binom{c_0+c_1}{c_0}$.

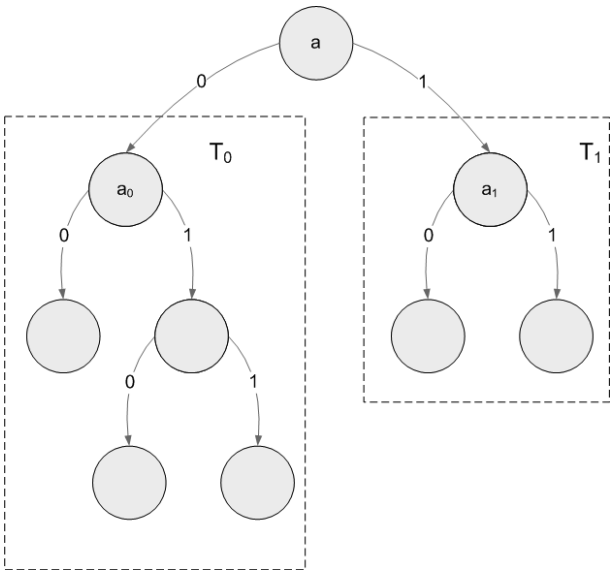

Рис. 4.1: Декомпозиция би-дерева T на би-деревья T_0 и T_1.

Доказательство. Выделим в T главное (c_0, c_1)-поддерево T^* следующим образом. Обозначим через $E_0(T)$ множество $\{x \in V_*^0(T) | e_0(x) > c_0\}$, а через $E_1(T)$ обозначим множество $\{x \in V_*^1(T) | e_1(x) > c_1\}$. Если $E_0(T) = \emptyset$ и $E_1(T) = \emptyset$, то положим $T^* = T$. Пусть $E_0(T) \neq \emptyset$. Рассмотрим вершину $x \in E_0(T)$. Существует вершина x', $x' \in \pi(x)$, такая что $e_0(x) = c_0$ и дуга, входящая в x', помечена 0. Рассмотрим главное би-поддерево T' дерева T, получаемое удалением из T максимального би-поддерева с корнем x'. Дерево T' удовлетворяет соотношениям 1,2 доказываемого утверждения, так как все его концевые вершины, кроме x', также являются концевыми вершинами дерева T, а для вершины x' справедливо $e_0(x') = c_0$. Заметим, что $|E_0(T')| \leq |E_0(T)| - 1$. Поэтому, повторяя данную процедуру не более чем $|E_0(T)|$ раз, получим поддерево T'' би-дерева T, удовлетворяющее соотношениям 1,2 доказываемого утверждения и условию $|E_0(T'')| = 0$. Аналогично построим поддерево T''' би-дерева T'', удовлетворяющее соотношениям 1,2 доказываемого утверждения и условиям $E_0(T''') = \emptyset$, $E_1(T''') = \emptyset$, и положим $T^* = T'''$. Согласно утверждению 1, $v_*(T^*) = \binom{c_0+c_1}{c_0}$. Так как T^* — поддерево дерева T, то $v_*(T) \geq v_*(T^*)$. Следовательно, $v_*(T) \geq \binom{c_0+c_1}{c_0}$. □

Утверждение 4.12 *Пусть T – би-дерево, для которого справедливы следую-*

щие условия:

1. $e_0(x) \leq c_0$ *для всех вершин* x*, принадлежащих* $V_*^0(T)$*;*

2. $e_1(x) \leq c_1$ *для всех вершин* x*, принадлежащих* $V_*^1(T)$*.*

Тогда $v_*(T) \leq \binom{c_0+c_1}{c_0}$*.*

Доказательство. Для любого би-дерева T, удовлетворяющего условиям 1, 2, обозначим через $E_0(T)$ множество $\{x \in V_*^0(T) | e_0(x) < c_0\}$ и через $E_1(T)$ множество $\{x \in V_*^1(T) | e_1(x) < c_1\}$. Покажем, что дерево T является главным поддеревом некоторого (c_0, c_1)-дерева T^*. Если $E_0(T) = \emptyset$ и $E_1(T) = \emptyset$, то $T^* = T$. Пусть $E_0(T) \neq \emptyset$. Рассмотрим вершину $x \in E_0(T)$. Пусть y — вершина, из которой исходит дуга, ведущая в x. Очевидно, что $e_1(x) = e_1(y)$. Поскольку из y исходит также дуга, помеченная 1, имеем $e_1(y) < c_1$. Поэтому $e_1(x) < c_1$. Имеем также $e_0(x) < c_0$. Положим $d_0 = c_0 - e_0(x)$, $d_1 = c_1 - e_1(x)$. Пусть S — произвольное (d_0, d_1)-дерево. Построим би-дерево T' добавлением к дереву T дерева S, совместив x с корнем дерева S. В результате концевая вершина x в T заменится $v_*(S)$ концевыми вершинами в T', и для каждой вершины x' из этих концевых вершин выполняется $e_0(x') = c_0$, если $x' \in E_0(T')$, и $e_1(x') = c_1$, если $x' \in E_1(T')$. Следовательно, $|E_0(T')| = |E_0(T)| - 1$ и $|E_1(T')| = |E_1(T)|$. Повторяя данную процедуру $|E_0(T)|$ раз для каждой вершины из $E_0(T)$, получим би-дерево T'', содержащее T в качестве главного поддерева и удовлетворяющее условиям 1, 2 и соотношению $|E_0(T'')| = 0$. Аналогичную процедуру можно также применить для каждой вершины из $E_1(T)$. В результате получим (c_0, c_1)-дерево T^*, содержащее дерево T в качестве главного поддерева. Очевидно, что $v_*(T) \leq v_*(T^*)$. Согласно утверждению 1 имеем $v_*(T^*) = \binom{c_0+c_1}{c_0}$. Поэтому $v_*(T) \leq \binom{c_0+c_1}{c_0}$. $\square$

Пусть веса предметов в задаче (4.5) упорядочены в порядке убывания: $w_1 \geq \cdots \geq w_n$. Определим величины s и t следующим образом:

$$s = \min\left\{k \in N : \sum_{i=1}^{k} w_i > C\right\}. \tag{4.32}$$

$$t = \min\left\{k \in N : \sum_{i=n-k+1}^{n} w_i > C\right\}. \tag{4.33}$$

Утверждение 4.13 *Пусть T — дерево ветвления для решения задачи (4.5) ослабленным вариантом МВГ. Тогда для любой вершины x из множества $V_*^1(T)$ справедливо $s \leq e_1(x) \leq t$ и для любой вершины y из множества $V_*^0(T)$ справедливо $n+1-t \leq e_0(y) \leq n+1-s$, где величины s и t определяются соотношениями (4.32, 4.33).*

Доказательство. Пусть x — произвольная вершина из $V_*^1(T)$, и соответствующая этой вершине подзадача P имеет вид (4.6). Введем обозначения для множеств индексов фиксированных переменных, принимающих значения 0 и 1: $I_0 = \{i \in I | \theta_i = 0\}, I_1 = \{i \in I | \theta_i = 1\}$. Обозначим через $\hat{P}$ подзадачу, из которой была получена подзадача P в результате декомпозиции. Так как $x \in V_*^1(T)$, то подзадача P не может удовлетворять условию отсева **C2′**, поскольку в противном случае этому же условию удовлетворяет подзадача $\hat{P}$. Следовательно, подзадача P должна удовлетворять условию отсева **C1**, т.е. $\sum_{i\in I} \theta_i w_i > C$, что равносильно $\sum_{i\in I_1} w_i > C$. Так как $w_1 \geq \cdots \geq w_n$, то

$$\sum_{i=1}^{|I_1|} w_i \geq \sum_{i\in I_1} \theta_i w_i > C.$$

Согласно (4.32) получаем $e_1(x) = |I_1| \geq s$.

Пусть x_j — переменная ветвления для подзадачи $\hat{P}$. Так как $x \in V_*^1(T)$, то $\theta_j = 1$. Обозначим через J_1 множество $I_1 \setminus \{j\}$. Так как подзадача $\hat{P}$ не удовлетворяет условию отсева **C1**, то $\sum_{i\in J_1} w_i \leq C$. Так как $w_1 \geq \cdots \geq w_n$,

то $\sum_{i=n-|J_1|+1}^{n} w_i \leq \sum_{i\in J_1} w_i \leq C$. Согласно (4.33) получаем $e_1(x) = |I_1| = |J_1| + 1 \leq t$.

Пусть теперь подзадача P вида (4.6) соответствует произвольной вершине y из $V_*^0(T)$. Пусть также подзадача P получена в результате декомпозиции из подзадачи $\hat{P}$. Так как $y \in V_*^0(T)$, то подзадача P не может удовлетворять условию отсева **C1**, поскольку в противном случае этому же условию удовлетворяет подзадача $\hat{P}$. Следовательно, подзадача P удовлетворяет условию отсева **C2′**. Поэтому $\sum_{i\in I} \theta_i w_i + \sum_{i\in N\setminus I} w_i \leq C$, что эквивалентно неравенству $\sum_{i\in N\setminus I_0} w_i \leq C$. Так как $w_1 \geq \cdots \geq w_n$, то $\sum_{i=n-|N\setminus I_0|+1}^{n} w_i \leq \sum_{i\in N\setminus I_0} w_i \leq C$. Согласно (4.33) справедливо $n - e_0(y) = |N \setminus I_0| \leq t - 1$, следовательно $e_0(y) \geq n + 1 - t$.

Так как $y \in V_*^0(T)$, то $\theta_j = 0$, где j — индекс переменной ветвления для подзадачи $\hat{P}$. Обозначим через J_0 множество $I_0 \setminus \{j\}$. Так как для подзадачи $\hat{P}$ не выполнено условие отсева **C2′**, то $\sum_{i\in N\setminus J_0} w_i > C$. Так как $w_1 \geq \cdots \geq w_n$, то $\sum_{i=1}^{|N\setminus J_0|} w_i \geq \sum_{i\in N\setminus J_0} w_i > C$. Согласно (4.32) $n - e_0(y) + 1 = |N \setminus J_0| \geq s$. Таким образом, $e_0(y) \leq n + 1 - s$. □ Из доказанного утверждения и утверждений 2, 3 вытекает следующее

Утверждение 4.14 *Приведенная сложность V решения задачи (4.5) ослабленным вариантом МВГ удовлетворяет соотношениям* [3]

$$\binom{n+1-t+s}{s} \leq V \leq \binom{n+1-s+t}{t}, \tag{4.34}$$

где величины s и t определяются соотношениями (4.32), (4.33).

Несложно заметить, что в соотношениях (4.34) верхняя оценка для V совпадает с нижней тогда и только тогда, когда $s = t$.

Получим теперь оценку сложности, которая опирается на отношение максимального и минимального весов предметов. Будем использовать следующие обозначения для минимального и максимального весов: $\underline{w} = \min_{i\in N} w_i, \overline{w} =$

[3] Положим $\binom{a}{b} = 1$ при $a < b$.

$\max_{i \in N} w_i$. Пусть T — дерево ветвления для решения задачи (4.5) ослабленным вариантом МВГ, x — концевая вершина дерева T, которой соответствует подзадача P, имеющая вид (4.6). Обозначим через $\tilde{\sigma}(x) = (\sigma_1(x), \ldots, \sigma_n(x))$ булев набор длины n, определяемый следующим образом:

$$\sigma_i(x) = \begin{cases} \theta_i, \text{ если } i \in I; \\ 0, \text{ если } i \notin I \text{ и } x \in V_*^1(T); \\ 1, \text{ если } i \notin I \text{ и } x \in V_*^0(T); \end{cases} \qquad i = 1, \ldots, n.$$

Заметим, что для любых различных концевых вершин x, y дерева T наборы $\tilde{\sigma}(x)$, $\tilde{\sigma}(y)$ различаются в компоненте, номер которого равен индексу переменной ветвления для подзадачи, соответствующей корню наименьшего биподдерева в дереве T, содержащего обе вершины x и y. Таким образом, если $x \neq y$, то $\tilde{\sigma}(x) \neq \tilde{\sigma}(y)$. Будем обозначать через $W(x)$ сумму $\sum_{i=1}^{n} \sigma_i(x) \cdot w_i$ и через $\|\tilde{\sigma}(x)\|$ число единичных компонент набора $\tilde{\sigma}(x)$. Очевидно, что $W(x) > C$ для любой вершины x из $V_*^1(T)$ и $W(y) \leq C$ для любой вершины y из $V_*^0(T)$. Таким образом, для любых $x \in V_*^1(T)$, $y \in V_*^0(T)$ выполняется $W(y) < W(x)$, тем самым либо $\tilde{\sigma}(y) < \tilde{\sigma}(x)$, либо наборы $\tilde{\sigma}(x)$, $\tilde{\sigma}(y)$ несравнимы.

Утверждение 4.15 *Пусть T — дерево ветвления для решения задачи (4.5) ослабленным вариантом МВГ. Тогда для любых вершин x, y, $x \in V_*^1(T), y \in V_*^1(T)$ или $x \in V_*^0(T), y \in V_*^0(T)$, таких что $\tilde{\sigma}(x) \leq \tilde{\sigma}(y)$, выполняется $\|\tilde{\sigma}(y)\| - \|\tilde{\sigma}(x)\| < \frac{\overline{w}}{\underline{w}}$.*

Доказательство. Пусть $x \in V_*^1(T)$, соответствующая вершине x подзадача P имеет вид (4.6), $\hat{P}$ — подзадача, из которой получена в результате декомпозиции подзадача P, и x_j — переменная ветвления для подзадачи $\hat{P}$. Так как подзадача $\hat{P}$ не удовлетворяет условию отсева **C1**, то $\sum_{i \in I \setminus \{j\}} \theta_i w_i \leq C$. Следовательно, $W(x) = \sum_{i \in I} \theta_i w_i \leq C + w_j \leq C + \overline{w}$. Кроме того, как отмечено выше, $W(x) > C$. Таким образом, для любой вершины $x \in V_*^1(T)$

имеем $C < W(x) \leq C + \overline{w}$. Поэтому для любых $x, y \in V_*^1(T)$ выполняется $|W(x) - W(y)| < \overline{w}$. С другой стороны, если $\tilde{\sigma}(x) \leq \tilde{\sigma}(y)$, то, очевидно, $W(y) - W(x) \geq (\|\tilde{\sigma}(y)\| - \|\tilde{\sigma}(x)\|) \cdot \underline{w}$. Таким образом, в этом случае имеем $(\|\tilde{\sigma}(y)\| - \|\tilde{\sigma}(x)\|) \cdot \underline{w} < \overline{w}$, т.е. $\|\tilde{\sigma}(y)\| - \|\tilde{\sigma}(x)\| < \overline{w}/\underline{w}$.

Пусть теперь $x \in V_*^0(T)$. Подзадача $\hat{P}$ не удовлетворяет условию отсева **C2′**, поэтому $w_j + \sum_{i \in I \setminus \{j\}} \theta_i w_i + \sum_{i \in N \setminus I} w_i > C$. Следовательно, $W(x) = \sum_{i \in I \setminus \{j\}} \theta_i w_i + \sum_{i \in N \setminus I} w_i > C - w_j \geq C - \overline{w}$. Кроме того, как отмечено выше, $W(x) \leq C$. Таким образом, для любой вершины $x \in V_*^0(T)$ имеем $C - \overline{w} < W(x) \leq C$. Следовательно, для любых $x, y \in V_*^0(T)$ выполняется $|W(x) - W(y)| < \overline{w}$. Поэтому, аналогично случаю вершин из $V_*^1(T)$, из $\tilde{\sigma}(x) \leq \tilde{\sigma}(y)$ следует, что $\|\tilde{\sigma}(y)\| - \|\tilde{\sigma}(x)\| < \overline{w}/\underline{w}$. □

Из доказанного утверждения вытекает, что как любая цепочка попарно сравнимых наборов $\tilde{\sigma}(x)$ таких, что $x \in V_*^1(T)$, так и любая цепочка попарно сравнимых наборов $\tilde{\sigma}(x)$ таких, что $x \in V_*^0(T)$, содержит не более $\lceil \overline{w}/\underline{w} \rceil$ наборов. Поэтому множество всех наборов $\tilde{\sigma}(x)$ таких, что $x \in V_*^1(T)$ ($x \in V_*^0(T)$), можно покрыть не более чем $\lceil \overline{w}/\underline{w} \rceil$ непересекающимися антицепями. Следовательно, множество всех наборов $\tilde{\sigma}(x)$ таких, что $x \in V_*(T)$, можно покрыть не более чем $2\lceil \overline{w}/\underline{w} \rceil$ непересекающимися антицепями. С другой стороны, нетрудно убедиться, что максимальная суммарная мощность произвольного числа непересекающихся антицепей в n-мерном единичном кубе равна максимальной суммарной мощности такого же числа различных слоев этого куба. Следовательно справедливо утверждение

Утверждение 4.16 *Приведенная сложность решения задачи (4.5) ослабленным вариантом МВГ не превосходит* $\sum_{i=\lfloor n/2 \rfloor - \lceil \overline{w}/\underline{w} \rceil + 1}^{\lfloor n/2 \rfloor + \lceil \overline{w}/\underline{w} \rceil} \binom{n}{i}$.

Отметим, что любому стандартному варианту $\mathcal{A}$ метода ветвей и границ можно сопоставить ослабленный вариант $\mathcal{A}^*$, получающийся из $\mathcal{A}$ заменой условия

отсева **C2** на **C2′**. Очевидно, что сложность решения задачи методом $\mathcal{A}$ не превосходит сложности решения задачи методом $\mathcal{A}^*$. Поэтому справедлива

Теорема 4.3 *Приведенная сложность V решения задачи (4.5) стандартным вариантом МВГ удовлетворяет следующим неравенствам:*

$$V \leq \binom{n+1-s+t}{t}, \tag{4.35}$$

$$V \leq \sum_{i=\lfloor n/2 \rfloor - \lceil \overline{w}/\underline{w} \rceil + 1}^{\lfloor n/2 \rfloor + \lceil \overline{w}/\underline{w} \rceil} \binom{n}{i}, \tag{4.36}$$

где величины s и t определяются соотношениями (4.32, 4.33).

Из оценки (4.36) вытекает, что сложность решения задачи о ранце стандартным вариантом МВГ не превосходит по порядку $\frac{\overline{w}}{\underline{w}} \cdot \frac{2^n}{\sqrt{n}}$. Из оценки (4.35) следует, что если величина t ограничена, то сложность решения задачи о ранце стандартным вариантом МВГ ограничена сверху полиномом от числа переменных задачи. Ограниченность t может иметь место, например если отношение $C/\underline{w}$ ограничено сверху константой при росте n, так как $t \leq C/\underline{w}$.

Следствие 4.2 *Если величина t ограничена сверху константой t_0, то сложность решения задачи (4.5) стандартным вариантом МВГ ограничена сверху полиномом $P(n) = \frac{1}{t_0!} \prod_{i=1}^{t_0} (n+1-s+i)$ степени t_0, зависящим от размерности n данной задачи.*

4.6 Верхняя оценка сложности мажоритарного алгоритма для задачи о ранце

В данном разделе рассмотрим вариант метода ветвей и границ для задачи о ранце, называемый *мажоритарным*. Отметим, что поскольку при решении задачи о сумме подмножеств мажоритарным методом ветвей и границ в качестве переменной ветвления всегда выбирается свободная переменная с наи-

большим весом, без ограничения общности мы будем полагать, что все переменные задачи (4.10) упорядочены в порядке не возрастания их весов, т.е. $w_1 \geq w_2 \geq \ldots \geq w_n$. В таком случае в качестве переменной ветвления всегда выбирается свободная переменная с наименьшим номером, т.е. для каждой рассматриваемой в процессе решения подзадачи вида (4.6) выполняется $I = \{1, 2, \ldots, s\}$, где $0 \leq s < n$, при этом x_{s+1} является переменной ветвления данной подзадачи.

Концевым вершинам дерева ветвления соответствуют подзадачи, удовлетворяющие условиям **C1** и **C2'**. Каждой вершине, удовлетворяющей условию **C1**, сопоставим двоичный набор длины n, являющийся 0-дополнением множества значений фиксированных переменных соответствующей подзадачи. Такие наборы будем называть *0-наборами*. Каждой вершине, удовлетворяющей условию **C2'**, сопоставим двоичный набор длины n, который является 1-дополнением набора значений фиксированных переменных соответствующей подзадачи. Такие наборы будем называть *1-наборами*.

Введем в рассмотрение множество B^n двоичных наборов длины n. Определим в этом множестве частичный порядок следующим образом: $\tilde{\alpha} \leq \tilde{\beta}$, если $\alpha_i \leq \beta_i$ для всех $i \in N$. Если соотношение $\tilde{\alpha} \leq \tilde{\beta}$ не выполнено, то будем писать $\tilde{\alpha} \not\leq \tilde{\beta}$.

Утверждение 4.17 *Все 0-наборы образуют антицепь (множество попарно несравнимых элементов) в* B^n.

Доказательство. Проведем доказательство методом "от противного". Пусть $\tilde{\alpha}$ и $\tilde{\beta}$ — два различных 0-набора, при этом $\tilde{\alpha} \leq \tilde{\beta}$. Тогда найдется $j \in N$, такое что $\alpha_j = 0, \beta_j = 1$. Согласно определению 0-набора, $\sum_{i \in N} \alpha_i w_i > C$. Поэтому

$$\sum_{i \in N} \beta_i w_i > \sum_{i \in N} \alpha_i w_i + w_j > C + w_j. \tag{4.37}$$

С другой стороны, если подзадача P_β была получена в результате разбиения подзадачи P по переменной x_k, то согласно определению метода ветвей и гра-

ниц, $w_k = \min_{i \in I_\beta} w_i$, где I_β — множество фиксированных переменных подзадачи P_β, т.е. $w_k \leq w_j$. Так как P не удовлетворяет условию **C1**, то $\sum_{i \in I_\beta \setminus \{k\}} \beta_i w_i < C$. Следовательно

$$\sum_{i \in N} \beta_i w_i = \sum_{i \in I_\beta} \beta_i w_i < C + w_k \leq C + w_j. \tag{4.38}$$

Неравенства (4.37) и (4.38) противоречат друг другу, поэтому сделанное предположение неверно. Тем самым утверждение доказано. □

Аналогичным образом устанавливается справедливость следующего

Утверждение 4.18 *Все 1-наборы образуют антицепь в* B^n.

Антицепь, образованную 0-наборами будем называть *0-антицепью*. Антицепь, образованную 1-наборами — *1-антицепью*. Следующее утверждение устанавливает связь между элементами этих антицепей.

Утверждение 4.19 *Если* $\tilde{\alpha}$ *— 0-набор и* $\tilde{\beta}$ *— 1-набор, то* $\tilde{\alpha} \not\leq \tilde{\beta}$.

Доказательство. Очевидно, что для любых двух наборов $\tilde{\alpha}$ и $\tilde{\beta}$ таких, что $\tilde{\alpha} \leq \tilde{\beta}$, выполняется $f(\tilde{\alpha}) \leq f(\tilde{\beta})$. Так как 1-набор является допустимым решением задачи (4.10), то $f(\tilde{\beta}) \leq C$. Так как 0-набор кодирует подзадачу, удовлетворяющую условию **C1**, то $f(\tilde{\alpha}) > C$. Тогда $f(\tilde{\beta}) < f(\tilde{\alpha})$, откуда следует, что $\tilde{\alpha} \not\leq \tilde{\beta}$. □

Число единичных компонент набора $\tilde{\alpha}$ из B^n называется *весом* набора $\tilde{\alpha}$ и обозначается через $\|\tilde{\alpha}\|$. Обозначим через B^n_+ множество всех двоичных наборов длины n, у которых единичных компонент больше, чем нулевых. Пусть $\tilde{\alpha} = (\alpha_1, \ldots, \alpha_n)$ — произвольный двоичный набор из B^n_+. *Поднабором* набора $\tilde{\alpha}$ будем называть двоичный набор $(\alpha_{i_1}, \ldots, \alpha_{i_k})$, где $1 \leq i_1 < \ldots < i_k \leq n$, $k \geq 1$, составленный из произвольного непустого подмножества компонент набора $\tilde{\alpha}$ (при этом для каждой компоненты поднабора сохраняется ее номер в исходном наборе $\tilde{\alpha}$).

Для $1 \le i, j \le n$ через $\tilde{\alpha}[i : j]$ будем обозначать поднабор $(\alpha_i, \ldots, \alpha_j)$ в случае $i \le j$ и поднабор $(\alpha_1, \ldots, \alpha_j, \alpha_i, \ldots, \alpha_n)$ в случае $i > j$. Такие поднаборы будем называть *сегментами*. Будем говорить что сегмент $\tilde{\alpha}[i : j]$ *предшествует* компоненте α_{j+1} в случае $j < n$ и компоненте α_1 в случае $j = n$. Два сегмента назовем *изолированными*, если никакие компоненты из одного из сегментов не являются соседними с компонентами из другого сегмента (при этом компоненты α_1 и α_n также считаются соседними). В случае $i \le j$ под *префиксом* сегмента $\tilde{\alpha}[i : j]$ будем понимать любой сегмент $\tilde{\alpha}[i : j']$, где $i \le j' \le j$, в противном случае под *префиксом* сегмента $\tilde{\alpha}[i : j]$ понимается любой сегмент $\tilde{\alpha}[i : j']$, где $i \le j' \le n$ либо $1 \le j' \le j$.

Пусть $\tilde{\beta} = (\alpha_{i_1}, \ldots, \alpha_{i_k})$ — произвольный поднабор набора $\tilde{\alpha}$, содержащий по крайней мере одну нулевую и одну единичную компоненты. Пусть α_{i_j} — имеющая минимальный номер нулевая компонента поднабора $\tilde{\beta}$ такая, что $\alpha_{i_{j+1}} = 1$ (при этом полагаем $i_{k+1} = i_1$). Введем операцию *редукции набора* ∇: обозначим через $\nabla(\tilde{\beta})$ поднабор набора $\tilde{\alpha}$, полученный из поднабора $\tilde{\beta}$ удалением компонент α_{i_j} и $\alpha_{i_{j+1}}$.

Пусть t — число нулевых компонент набора $\tilde{\alpha}$, $t = n - \|\tilde{\alpha}\| < n/2$. Тогда операцию редукции к набору $\tilde{\alpha}$ можно применить последовательно t раз подряд. Рассмотрим последовательность $\tilde{\beta}^{(0)}, \tilde{\beta}^{(1)}, \ldots, \tilde{\beta}^{(t)}$ поднаборов набора $\tilde{\alpha}$ такую, что $\tilde{\beta}^{(0)} = \tilde{\alpha}$ и $\tilde{\beta}^{(i)} = \nabla(\tilde{\beta}^{(i-1)})$ для $i = 1, \ldots, t$. Будем говорить, что две компоненты набора $\tilde{\alpha}$ *образуют связанную пару* в этом наборе, если они были удалены в результате применения операции ∇ к одному из наборов $\tilde{\beta}^{(0)}, \tilde{\beta}^{(1)}, \ldots, \tilde{\beta}^{(t-1)}$. Все компоненты, образующие связанные пары в $\tilde{\alpha}$, будем также называть *связанными* в $\tilde{\alpha}$. Оставшиеся компоненты набора $\tilde{\alpha}$ будем называть *несвязанными* в $\tilde{\alpha}$.

Положим $\mathcal{R}(\tilde{\alpha}) = \tilde{\beta}^{(t)}$. Очевидно, что $\mathcal{R}(\tilde{\alpha})$ состоит из всех несвязанных в $\tilde{\alpha}$ компонент. Заметим также, что поднабор $\mathcal{R}(\tilde{\alpha})$ содержит только единичные

компоненты набора $\tilde{\alpha}$, т.е. все несвязанные в рассматриваемом двоичном наборе компоненты являются единичными.

Нетрудно заметить, что множество всех связанных в $\tilde{\alpha}$ компонент однозначным образом разбивается на попарно изолированные сегменты такие, что в каждом из сегментов содержится поровну нулевых и единичных компонент и любые две компоненты, образующие связанную пару в $\tilde{\alpha}$, содержатся в одном и том же сегменте. Мы будем называть данные сегменты *сегментами связанности* набора $\tilde{\alpha}$. Индукцией по числу компонент в сегменте связанности нетрудно также доказать следующий факт.

Утверждение 4.20 *В любом префиксе сегмента связанности число единичных компонент не превосходит числа нулевых компонент.*

Имеет место следующий критерий.

Лемма 4.1 *Единичная компонента набора $\tilde{\alpha}$ является связанной тогда и только тогда, когда в $\tilde{\alpha}$ найдется предшествующий данной компоненте сегмент, в котором число нулевых компонент превосходит число единичных компонент.*

Доказательство. Пусть α_i — связанная в $\tilde{\alpha}$ единичная компонента, принадлежащая сегменту связанности $\tilde{\alpha}[i' : j']$. Согласно утверждению 4.20 имеем $\alpha_{i'} = 0$, т.е. $i \neq i'$ и в префиксе $\tilde{\alpha}[i' : i]$ сегмента $\tilde{\alpha}[i' : j']$ число единичных компонент не превосходит числа нулевых компонент. Следовательно, в префиксе сегмента $\tilde{\alpha}[i' : j']$, предшествующем компоненте α_i, число нулевых компонент превосходит число единичных компонент. С другой стороны, пусть существует некоторый предшествующий компоненте α_i сегмент, в котором число нулевых компонент превосходит число единичных компонент. Поскольку все нулевые компоненты являются связанными в $\tilde{\alpha}$, в этом сегменте найдется по крайней мере одна нулевая компонента, которая образует связанную пару в $\tilde{\alpha}$ с единич-

ной компонентой, находящейся вне этого сегмента. В таком случае компонента α_i должна быть связанной в $\tilde{\alpha}$. $\square$

Из леммы 4.1 нетрудно получить

Следствие 4.3 *Пусть* $\tilde{\alpha} = (\alpha_1, \ldots, \alpha_n)$, $\tilde{\alpha}' = (\alpha'_1, \ldots, \alpha'_n)$ *— два набора из* B^n_+ *таких, что* $\tilde{\alpha} \leq \tilde{\alpha}'$, *и* α_i *является несвязанной в* $\tilde{\alpha}$ *единичной компонентой. Тогда* α'_i *является несвязанной в* $\tilde{\alpha}'$ *единичной компонентой.*

Определим операцию модификации $\mathcal{D}$ набора: обозначим через $\mathcal{D}(\tilde{\alpha})$ двоичный набор, получающийся из набора $\tilde{\alpha}$ заменой на ноль несвязанной в $\tilde{\alpha}$ единичной компоненты с максимальным номером.

Из следствия 4.3 непосредственно вытекает

Следствие 4.4 *Пусть* $\tilde{\alpha} = (\alpha_1, \ldots, \alpha_n)$, $\tilde{\alpha}' = (\alpha'_1, \ldots, \alpha'_n)$ *— два набора из* B^n_+ *таких, что* $\tilde{\alpha} = \mathcal{D}(\tilde{\alpha}')$, *и* α_i *является несвязанной в* $\tilde{\alpha}$ *единичной компонентой. Тогда* α'_i *является несвязанной в* $\tilde{\alpha}'$ *единичной компонентой.*

Докажем инъективность операции $\mathcal{D}$.

Лемма 4.2 *Для любых наборов* $\tilde{\alpha}', \tilde{\alpha}'' \in B^n_+$ *из* $\tilde{\alpha}' \neq \tilde{\alpha}''$ *следует* $\mathcal{D}(\tilde{\alpha}') \neq \mathcal{D}(\tilde{\alpha}'')$.

Доказательство. Предположим, что для некоторых двух различных наборов $\tilde{\alpha}' = (\alpha'_1, \ldots, \alpha'_n)$, $\tilde{\alpha}'' = (\alpha''_1, \ldots, \alpha''_n)$ из B^n_+ имеет место $\mathcal{D}(\tilde{\alpha}') = \mathcal{D}(\tilde{\alpha}'')$. Пусть $\mathcal{D}(\tilde{\alpha}')$ получен из $\tilde{\alpha}'$ заменой на ноль единичной компоненты $\alpha'_{i'}$, а $\mathcal{D}(\tilde{\alpha}'')$ получен из $\tilde{\alpha}''$ заменой на ноль единичной компоненты $\alpha''_{i''}$. Так как $\tilde{\alpha}' \neq \tilde{\alpha}''$, то $i' \neq i''$. Не ограничивая общности, будем считать, что $i' > i''$. Поскольку $\alpha''_{i''}$ является несвязанной в $\tilde{\alpha}''$ компонентой с максимальным номером, компоненты $\alpha''_{i''+1}, \ldots, \alpha''_n$ являются связанными в $\tilde{\alpha}''$, т.е. сегмент $\tilde{\alpha}''[i''+1 : n]$ является префиксом некоторого сегмента связанности в $\tilde{\alpha}''$. Следовательно, согласно утверждению 4.20, в наборе $\tilde{\alpha}''$ между компонентами $\alpha''_{i''}$ и $\alpha''_{i'}$ число единичных

компонент не превосходит числа нулевых компонент (этот факт, очевидно, также имеет место в случае, когда компоненты $\alpha''_{i''}$, $\alpha''_{i'}$ являются соседними). Так как $\mathcal{D}(\tilde{\alpha}') = \mathcal{D}(\tilde{\alpha}'')$, все компоненты набора $\tilde{\alpha}'$ кроме $\alpha'_{i'}$ и $\alpha'_{i''}$ должны совпадать с соответствующими компонентами набора $\tilde{\alpha}''$. Поэтому в наборе $\tilde{\alpha}'$ между компонентами $\alpha'_{i''}$ и $\alpha'_{i'}$ число единичных компонент также не превосходит числа нулевых компонент. Кроме того, из $\mathcal{D}(\tilde{\alpha}') = \mathcal{D}(\tilde{\alpha}'')$ вытекает, что $\alpha'_{i''} = 0$. Таким образом, в сегменте $\tilde{\alpha}'[i'' : i' - 1]$ набора $\tilde{\alpha}'$ число нулевых компонент превосходит число единичных компонент. Следовательно, согласно лемме 4.1 компонента $\alpha'_{i'}$ является связанной в $\tilde{\alpha}'$, что противоречит определению данной компоненты как несвязанной в $\tilde{\alpha}'$ компоненты с максимальным номером.□

Введем в рассмотрение набор $\tilde{\gamma}_s = (\underbrace{0, \ldots, 0}_{n-s}, \underbrace{1, \ldots, 1}_{s})$, где $s > n/2$.

Лемма 4.3 *Если для набора* $\tilde{\alpha} \in B^n_+$, $\|\tilde{\alpha}\| \geq n/2 + 1$, *справедливо* $\tilde{\alpha} \not\leq \tilde{\gamma}_s$, *то* $\mathcal{D}(\tilde{\alpha}) \not\leq \tilde{\gamma}_s$.

Доказательство. Пусть $\tilde{\alpha} = (\alpha_1, \ldots, \alpha_n)$. Предположим, что $\alpha_1 = 0$. Рассмотрим в $\tilde{\alpha}$ единичную компоненту с минимальным номером. Пусть такой компонентой является компонента α_i. Очевидно, что α_i образует с α_{i-1} связанную пару в $\tilde{\alpha}$, т.е. α_i является связанной в $\tilde{\alpha}$. Следовательно, α_i совпадает с соответствующей компонентой набора $\mathcal{D}(\tilde{\alpha})$, т.е. i-я компонента набора $\mathcal{D}(\tilde{\alpha})$ является единичной. Заметим, что $i \leq n - s$, поскольку в противном случае $\tilde{\alpha} \leq \tilde{\gamma}_s$. Поэтому $\mathcal{D}(\tilde{\alpha}) \not\leq \tilde{\gamma}_s$. Предположим теперь, что $\alpha_1 = 1$. Покажем, что тогда первая компонента набора $\mathcal{D}(\tilde{\alpha})$ также является единичной, поэтому соотношение $\mathcal{D}(\tilde{\alpha}) \leq \tilde{\gamma}_s$ не выполняется. Это очевидно в случае, если α_1 является связанной в $\tilde{\alpha}$. Пусть α_1 не является связанной в $\tilde{\alpha}$. Заметим, что из условия $\|\tilde{\alpha}\| > n/2 + 1$ вытекает, что по крайней мере две компоненты являются несвязанными в $\tilde{\alpha}$, поэтому α_1 не может быть несвязанной компонентой с максимальным номером. Следовательно, и в этом случае первая компонента набора $\mathcal{D}(\tilde{\alpha})$ совпадает с α_1,

тем самым лемма доказана.□

Лемма 4.4 *Пусть для набора* $\tilde{\alpha} \in B_+^n$ *выполняются соотношения* $\tilde{\alpha} \not\leq \tilde{\gamma}_s$ *и* $\mathcal{D}(\tilde{\alpha}) \leq \tilde{\gamma}_s$. *Тогда не существует набора* $\tilde{\alpha}' \in B^n$ *такого, что* $\tilde{\alpha} = \mathcal{D}(\tilde{\alpha}')$.

Доказательство. Пусть $\tilde{\alpha} = (\alpha_1, \ldots, \alpha_n)$. Из леммы 4.3 вытекает, что условия леммы 4.4 могут быть выполнены лишь в случае, когда n — нечетное число и $\|\tilde{\alpha}\| = (n+1)/2$. Соответственно, в этом случае в $\tilde{\alpha}$ имеется только одна несвязанная компонента α_i. Тогда α_i должна быть единственной единичной компонентой набора $\tilde{\alpha}$, удовлетворяющей условию $i \leq n - s$. Поэтому при $i > 1$ данная компонента должна образовывать связанную пару с нулевой компонентой α_{i-1}, т.е. должна быть связанной в $\tilde{\alpha}$. Следовательно, возможен только случай $i = 1$. Предположим, что существует набор $\tilde{\alpha}' = (\alpha'_1, \ldots, \alpha'_n)$ такой, что $\tilde{\alpha} = \mathcal{D}(\tilde{\alpha}')$. Тогда, согласно следствию 4.4, компонента α'_1 является несвязанной в $\tilde{\alpha}'$. Пусть α'_j — несвязанная компонента с максимальным номером в $\tilde{\alpha}'$. Так как в $\tilde{\alpha}'$ должно быть три несвязанных компоненты, то $j \neq 1$. При этом либо $j = n$, либо $\tilde{\alpha}'[j+1 : n]$ является сегментом связанности в $\tilde{\alpha}'$, т.е. в $\tilde{\alpha}'$ правее компоненты α'_j содержится поровну нулевых и единичных компонент. Поскольку набор $\tilde{\alpha}$ отличается от $\tilde{\alpha}'$ только в j-ой компоненте, в $\tilde{\alpha}$ правее компоненты α_j также должно содержаться поровну нулевых и единичных компонент. Так как $\tilde{\alpha}$ получается из $\tilde{\alpha}'$ заменой j-ой компоненты на 0, то $\alpha_j = 0$. Таким образом, в сегменте $\tilde{\alpha}[j : n]$ нулевых компонент должно быть на одну больше, чем единичных. Следовательно, согласно лемме 4.1, компонента α_1 должна быть связанной в $\tilde{\alpha}$, что противоречит тому, что α_1 является несвязанной в $\tilde{\alpha}$ компонентой. Таким образом, лемма доказана. □

Пусть T', T'' — две антицепи в B^n такие, что для любых двух наборов $\tilde{\alpha}' \in T'$, $\tilde{\alpha}'' \in T''$ справедливо $\tilde{\alpha}' \not\geq \tilde{\alpha}''$. В таком случае будем писать, что $T' < T''$ (отметим, что из $T' < T''$ вытекает $T' \cap T'' = \emptyset$). Для $s > n/2$ обозначим через $\mathcal{A}_s$ множество всех пар (T', T'') антицепей в B^n таких, что $T' < T''$ и $\tilde{\gamma}_s \in T'$.

Под мощностью пары непересекающихся антицепей будем понимать суммарную мощность этих антицепей.

Пару антицепей (T', T'') из $\mathcal{A}_s$ будем называть *правильной снизу*, если в множестве $(T' \cup T'') \setminus \{\tilde{\gamma}_s\}$ все наборы, имеющие минимальный вес, содержатся в T', и *правильной сверху*, если в множестве $(T' \cup T'') \setminus \{\tilde{\gamma}_s\}$ все наборы, имеющие максимальный вес, содержатся в T''. Заметим, что из любой пары антицепей $(T', T'') \in \mathcal{A}_s$, содержащих наборы с весом меньшим, чем s, мы можем получить правильную снизу пару антицепей из $\mathcal{A}_s$, переместив все имеющие минимальный вес наборы из $T' \cup T'' \setminus \{\tilde{\gamma}_s\}$ в антицепь T'. Полученную таким образом пару антицепей будем называть *нижним исправлением* исходной пары (T', T''). Аналогичным образом, из любой пары антицепей $(T', T'') \in \mathcal{A}_s$ мы можем получить правильную сверху пару антицепей из $\mathcal{A}_s$, которую будем называть *верхним исправлением* исходной пары (T', T''), переместив все имеющие максимальный вес наборы из $(T' \cup T'') \setminus \{\tilde{\gamma}_s\}$ в антицепь T''. Отметим, что антицепи как нижнего, так и верхнего исправлений пары антицепей состоят в совокупности из тех же наборов, что и исходные антицепи.

Лемма 4.5 *Для любого $s > n/2$ в $\mathcal{A}_s$ существует имеющая максимальную мощность пара антицепей, состоящих из наборов с весом не меньшим, чем $\lfloor n/2 \rfloor$.*

Доказательство. Рассмотрим произвольную пару антицепей (T', T'') из $\mathcal{A}_s$, имеющую максимальную мощность. Предположим, что минимальный вес наборов из $T' \cup T''$ равен $r < \lfloor n/2 \rfloor$. Пусть (T'_0, T''_0) — нижнее исправление пары (T', T''). Поскольку, очевидно, $T'_0 \cup T''_0 = T' \cup T''$, пара антицепей (T'_0, T''_0) также имеет максимальную мощность в $\mathcal{A}_s$, и минимальный вес наборов из $T'_0 \cup T''_0$ также равен r. При этом, поскольку пара антицепей (T'_0, T''_0) является правильной снизу, все наборы из $T'_0 \cup T''_0$ с весом r содержатся в T'_0. Обозначим множество всех таких наборов через V. Кроме того, обозначим через U множество всех

наборов из T_0'' с весом $r+1$. Обозначим также через V' множество всех наборов из B^n с весом $r+1$, сравнимых с хотя бы одним набором из V, и через U' множество всех наборов из B^n с весом $r+2$, сравнимых с хотя бы одним набором из U. Отметим, что каждый набор веса r соединен ровно $n-r$ ребрами с наборами веса $r+1$, а каждый набор веса $r+1$ соединен ровно $r+1$ ребрами с наборами веса r. Исходя из этих соображений, нетрудно получить, что $|V'| \geq \frac{n-r}{r+1}|V|$. Аналогичным образом получаем, что $|U'| \geq \frac{n-r-1}{r+2}|U|$. Отметим также, что, поскольку T_0' и T_0'' являются антицепями, множество V' не пересекается с T_0', а множество U' не пересекается с T_0''. Рассмотрим сначала случай $r < n/2-1$, т.е. $r \leq n/2-3/2$. В этом случае имеем $|V'| \geq \frac{n-r}{r+1}|V| > |V|$ и $|U'| \geq \frac{n-r-1}{r+2}|U| \geq |U|$. Положим $T_1' = (T_0' \setminus V) \cup V'$ и $T_1'' = (T_0'' \setminus U) \cup U'$. Нетрудно убедиться, что T_1' и T_1'' являются непересекающимися антицепями, образующими пару из $\mathcal{A}_s$. Кроме того, имеем $|T_1'| = |T_0'| + |V'| - |V| > |T_0'|$ и $|T_1''| = |T_0''| + |U'| - |U| \geq |T_0''|$. Следовательно, $|T_1' \cup T_1''| = |T_1'| + |T_1''| > |T_0'| + |T_0''| = |T_0' \cup T_0''|$, что противоречит тому, что пара антицепей (T_0', T_0'') имеет максимальную мощность в $\mathcal{A}_s$. Таким образом, случай $r < n/2-1$ невозможен. Рассмотрим теперь оставшийся случай $r = n/2-1$. Отметим, что это возможно лишь в случае, когда n — четное число, т.е. $n = 2k$, и $r = \lfloor n/2 \rfloor - 1 = k-1$. Обозначим через V'' множество $V' \cup U$. Отметим, что множество V'', так же как и множество V', не пересекается с T_0'. Положим $T_2' = (T_0' \setminus V) \cup V''$ и $T_2'' = (T_0'' \setminus U) \cup U'$. Нетрудно убедиться, что T_2' и T_2'' также являются непересекающимися антицепями, образующими пару из $\mathcal{A}_s$. Кроме того, имеем $|V''| \geq |V'| \geq \frac{n-r}{r+1}|V| = \frac{k+1}{k}|V|$, т.е. $|V| \leq \frac{k}{k+1}|V''|$. Имеем также $|U'| \geq \frac{n-r-1}{r+2}|U| = \frac{k}{k+1}|U|$. Таким образом, учитывая, что $|U| \leq |V''|$, получаем $(|V''| - |V|) + (|U'| - |U|) \geq \frac{1}{k+1}|V''| - \frac{1}{k+1}|U| \geq 0$. Следовательно, $|T_2' \cup T_2''| = |T_2'| + |T_2''| = |T_0'| + |T_0''| + (|V''| - |V|) + (|U'| - |U|) \geq |T_0'| + |T_0''| = |T_0' \cup T_0''|$, мощность пары антицепей (T_2', T_2'') не меньше мощности пары антицепей (T_0', T_0''). Таким образом, пара антицепей (T_2', T_2'') также имеет

максимальную мощность в $\mathcal{A}_s$, и антицепи T_2', T_2'', очевидно, состоят из наборов с весом не меньшим, чем $\lfloor n/2 \rfloor$. Тем самым лемма доказана.$\square$

Лемма 4.6 *Для любого $s > n/2$ в $\mathcal{A}_s$ существует имеющая максимальную мощность пара антицепей таких, что вес всех входящих в эти антицепи наборов, кроме набора $\tilde{\gamma}_s$, не больше, чем $(n+3)/2$, и не меньше, чем $\lfloor n/2 \rfloor$.*

Доказательство. Согласно лемме 4.5, в $\mathcal{A}_s$ существует некоторая имеющая максимальную мощность пара антицепей (T', T''), состоящих из наборов с весом не меньшим, чем $\lfloor n/2 \rfloor$. Предположим, что максимальный вес содержащихся в этих антицепях наборов, отличных от $\tilde{\gamma}_s$, равен $r > (n+3)/2$. Отметим, что из $r > (n+3)/2$, очевидно, вытекает $r \geq n/2 + 2$. Чтобы доказать лемму 4.5, достаточно показать, что тогда найдется имеющая максимальную мощность пара антицепей таких, что вес всех входящих в эти антицепи наборов, кроме набора $\tilde{\gamma}_s$, не меньше, чем $\lfloor n/2 \rfloor$, и не больше, чем $r-1$. Для этого рассмотрим верхнее исправление (T_0', T_0'') пары (T', T''). Поскольку антицепи T_0', T_0'' состоят в совокупности из тех же наборов, что и антицепи T', T'', пара антицепей (T_0', T_0'') также имеет максимальную мощность в $\mathcal{A}_s$ и вес всех входящих в эти антицепи наборов, кроме набора $\tilde{\gamma}_s$, не меньше, чем $\lfloor n/2 \rfloor$, и не больше, чем r. Кроме того, все входящие в эти антицепи наборы, отличные от $\tilde{\gamma}_s$ и имеющие максимальный вес r, содержатся в T_0''. Обозначим множество всех таких наборов через U. Кроме того, обозначим через V множество всех наборов веса $r-1$, содержащихся в T_0' и отличных от $\tilde{\gamma}_s$. Обозначим также через $\mathcal{D}(U)$ и $\mathcal{D}(V)$ множества $\{\mathcal{D}(\tilde{\alpha}) \mid \tilde{\alpha} \in U\}$ и $\{\mathcal{D}(\tilde{\alpha}) \mid \tilde{\alpha} \in V\}$ соответственно. Из леммы 4.2 вытекает, что $|\mathcal{D}(U)| = |U|$ и $|\mathcal{D}(V)| = |V|$. Кроме того, поскольку T_0' и T_0'' являются антицепями, множество $\mathcal{D}(V)$ не пересекается с T_0', а множество $\mathcal{D}(U)$ не пересекается с T_0''. Положим $T_1' = (T_0' \setminus V) \cup \mathcal{D}(V)$ и $T_1'' = (T_0'' \setminus U) \cup \mathcal{D}(U)$. Пользуясь леммой 4.3, нетрудно проверить, что множество T_1' является антицепью, содержащей набор $\tilde{\gamma}_s$. Кроме того, очевидно, что T_1'' также является антицепью.

В силу леммы 4.3 никакой набор $\tilde{\alpha}$ из T_1'' не может удовлетворять соотношению $\tilde{\alpha} \leq \tilde{\gamma}_s$. Учитывая этот факт, нетрудно убедиться, что $T_1' < T_2''$. Таким образом, $(T_1', T_1'') \in \mathcal{A}_s$. Из равенств $|\mathcal{D}(U)| = |U|$, $|\mathcal{D}(V)| = |V|$ вытекает, что $|T_1'| = |T_0'|$, $|T_1''| = |T_0''|$, поэтому пара антицепей (T_1', T_1'') также имеет максимальную мощность в $\mathcal{A}_s$. Для завершения доказательства леммы остается только отметить, что вес любого отличного от $\tilde{\gamma}_s$ набора из $T_1' \cup T_1''$ не меньше, чем $\lfloor n/2 \rfloor$, и не больше, чем $r - 1$. $\square$

Теорема 4.4 *При $s > n/2$ мощность любой пары антицепей из $\mathcal{A}_s$ не превосходит $1 + \binom{n+1}{\lfloor n/2 \rfloor + 1} - \binom{s+1}{\lfloor n/2 \rfloor + 1}$.*

Доказательство. Рассмотрим сначала случай, когда n четно, т.е. $n = 2k$. Тогда, согласно лемме 4.6, в $\mathcal{A}_s$ существует имеющая максимальную мощность пара антицепей (T', T'') таких, что все наборы из $|T' \cup T''|$, кроме набора $\tilde{\gamma}_s$, имеют вес равный либо k, либо $k+1$. Заметим, что тогда ни для какого набора $\tilde{\alpha}$ множества T'' не может быть выполнено соотношение $\tilde{\alpha} \geq \tilde{\gamma}_s$, поскольку вес набора $\tilde{\gamma}_s$ равен $s \geq k+1$. Следовательно, никакой набор из T'' не может быть сравнимым с $\tilde{\gamma}_s$. Поскольку T' является антицепью, никакой набор из T' также не может быть сравнимым с $\tilde{\gamma}_s$. Таким образом, все наборы из $T' \cup T'' \setminus \{\tilde{\gamma}_s\}$ являются несравнимыми с $\tilde{\gamma}_s$. Очевидно, что существует $\binom{n}{k+1} - \binom{s}{k+1}$ несравнимых с $\tilde{\gamma}_s$ наборов веса $k+1$ и $\binom{n}{k} - \binom{s}{k}$ наборов веса k. Поэтому

$$|T' \cup T'' \setminus \{\tilde{\gamma}_s\}| \leq \left(\binom{n}{k+1} - \binom{s}{k+1}\right) + \left(\binom{n}{k} - \binom{s}{k}\right) = \binom{n+1}{k+1} - \binom{s+1}{k+1}.$$

Следовательно, $|T' \cup T''| \leq 1 + \binom{n+1}{k+1} - \binom{s+1}{k+1}$. Поскольку пара (T', T'') имеет максимальную мощность в $\mathcal{A}_s$, получаем, что в этом случае мощность любой пары антицепей из $\mathcal{A}_s$ не превосходит $1 + \binom{n+1}{k+1} - \binom{s+1}{k+1} = 1 + \binom{n+1}{\lfloor n/2 \rfloor + 1} - \binom{s+1}{\lfloor n/2 \rfloor + 1}$.

Рассмотрим теперь случай, когда n нечетно, т.е. $n = 2k+1$. Согласно лемме 4.6, в этом случае в $\mathcal{A}_s$ существует имеющая максимальную мощность пара антицепей (T', T'') таких, что все наборы из $T' \cup T''$, кроме набора $\tilde{\gamma}_s$, имеют

вес равный либо k, либо $k+1$, либо $k+2$. Пусть (T_0', T_0'') — верхнее исправление пары (T', T''). Поскольку $T_0' \cup T_0'' = T' \cup T''$, пара антицепей (T_0', T_0'') также имеет максимальную мощность в $\mathcal{A}_s$, и все наборы из $T_0' \cup T_0''$, кроме набора $\tilde{\gamma}_s$, имеют вес равный либо k, либо $k+1$, либо $k+2$. Кроме того, все наборы из $T_0' \cup T_0'' \setminus \{\tilde{\gamma}_s\}$, имеющие вес $k+2$, содержатся в T_0''. Обозначим множество всех таких наборов через U. Положим $\mathcal{D}(U) = \{\mathcal{D}(\tilde{\alpha}) \mid \tilde{\alpha} \in U\}$, $V = \mathcal{D}(U) \cap T_0'$, и $\mathcal{D}(V) = \{\mathcal{D}(\tilde{\alpha}) \mid \tilde{\alpha} \in V\}$. Поскольку T_0', T_0'' являются антицепями, имеем $\mathcal{D}(U) \cap T_0'' = \emptyset$, $\mathcal{D}(V) \cap T_0' = \emptyset$. Обозначим через T_1' множество $(T_0' \setminus V) \cup \mathcal{D}(V)$ и через T_1'' множество $(T_0'' \setminus U) \cup \mathcal{D}(U)$. Нетрудно заметить, что $T_1' \cap T_1'' = \emptyset$ и любой набор из $T_1' \cup T_1'' \setminus \{\tilde{\gamma}_s\}$ имеет вес либо k, либо $k+1$. Учитывая, что вес набора $\tilde{\gamma}_s$ равен $s \geq k+1$, получаем, что ни для какого набора $\tilde{\alpha}$ из $T_1' \cup T_1'' \setminus \{\tilde{\gamma}_s\}$ не может быть выполнено соотношение $\tilde{\alpha} \geq \tilde{\gamma}_s$. Кроме того, соотношение $\tilde{\alpha} \leq \tilde{\gamma}_s$ также не может быть выполнено ни для какого набора $\tilde{\alpha}$ из T_1'' в силу леммы 4.3 и ни для какого набора $\tilde{\alpha}$ из $T_1' \setminus \{\tilde{\gamma}_s\}$ в силу леммы 4.4. Таким образом, никакой набор из $T_1' \cup T_1'' \setminus \{\tilde{\gamma}_s\}$ не является сравнимым с $\tilde{\gamma}_s$. Поэтому, аналогично рассмотренному ранее случаю для четного n, получаем, что $|T_1' \cup T_1'' \setminus \{\tilde{\gamma}_s\}| \leq \binom{n+1}{k+1} - \binom{s+1}{k+1}$. Следовательно, $|T_1' \cup T_1''| \leq 1 + \binom{n+1}{k+1} - \binom{s+1}{k+1} = 1 + \binom{n+1}{\lfloor n/2 \rfloor + 1} - \binom{s+1}{\lfloor n/2 \rfloor + 1}$. Из справедливых в силу леммы 4.2 равенств $|\mathcal{D}(U)| = |U|$, $|\mathcal{D}(V)| = |V|$ вытекает, что $|T_1'| = |T_0'|$, $|T_1''| = |T_0''|$, поэтому $|T_0' \cup T_0''| = |T_1' \cup T_1''|$. Следовательно, $|T_0' \cup T_0''| \leq 1 + \binom{n+1}{\lfloor n/2 \rfloor + 1} - \binom{s+1}{\lfloor n/2 \rfloor + 1}$. Поскольку пара антицепей (T_0', T_0'') имеет максимальную мощность в $\mathcal{A}_s$, получаем, что теорема 4.4 справедлива и в этом случае.□

Следствие 4.5 *При $s > n/2$ пара антицепей (T', T''), где T' состоит из набора $\tilde{\gamma}_s$ и всех наборов с весом $\lfloor n/2 \rfloor$, несравнимых с $\tilde{\gamma}_s$, а T'' состоит из всех наборов с весом $\lfloor n/2 \rfloor + 1$, несравнимых с $\tilde{\gamma}_s$, имеет максимальную мощность в множестве $\mathcal{A}_s$.*

Для $t \leq \lfloor n/2 \rfloor + 1$ обозначим через $\mathcal{A}_t'$ множество всех пар (T', T'') непересе-

кающихся и состоящих из наборов с весом не большим, чем t, антицепей в B^n таких, что $T' < T''$.

Теорема 4.5 *При* $t \leq \lfloor n/2 \rfloor + 1$ *мощность любой пары антицепей из* $\mathcal{A}'_t$ *не превосходит* $\binom{n+1}{t}$.

Доказательство. Пусть (T', T'') — произвольная пара антицепей из $\mathcal{A}'_t$. Рассмотрим сначала случай $t \leq \frac{n+1}{2}$. В множестве B^n имеется $n!$ максимальных цепей и каждая из этих цепей содержит не более одного набора из T' и не более одного набора из T''. Поэтому, если для каждого набора из $T' \cup T''$ рассмотреть множество всех содержащих этот набор максимальных цепей, то суммарная мощность этих множеств не превосходит $2n!$. Упорядочим все наборы из $T' \cup T''$ в порядке возрастания числа содержащих эти наборы максимальных цепей. Набор веса p содержится в $p!(n-p)!$ максимальных цепях, поэтому среди наборов из $T' \cup T''$ может быть не более $\binom{n}{t}$ наборов, содержащихся в $t!(n-t)!$ максимальных цепях, и каждый из остальных наборов должен содержаться в не менее чем $(t-1)!(n-t+1)!$ максимальных цепях. Следовательно, суммарная мощность множеств всех максимальных цепей, содержащих первые $\binom{n}{t}$ наборов из $T' \cup T''$, не меньше, чем $n!$, а каждый из остальных наборов из $T' \cup T''$ содержится в не менее чем $(t-1)!(n-t+1)!$ максимальных цепях. Таким образом, суммарная мощность множеств всех максимальных цепей, содержащих остальные наборы из $T' \cup T''$, не превосходит $n!$ и, следовательно, число таких наборов не превосходит $\binom{n}{t-1}$. Поэтому $|T' \cup T''| \leq \binom{n}{t} + \binom{n}{t-1} = \binom{n+1}{ts}$. Для случая $t = n/2 + 1$, возможного только при четном n, теорему 4.5 можно доказать аналогичным образом, принимая во внимание, что в этом случае в $T' \cup T''$ имеется не более чем $\binom{n}{n/2}$ наборов, содержащихся в $((n/2)!)^2$ максимальных цепях, а каждый из остальных наборов содержится в не менее чем $(n/2-1)!(n/2+1)!$ максимальных цепях. $\square$

Следствие 4.6 *При $s \leq \lfloor n/2 \rfloor + 1$ пара антицепей (T', T''), где T' состоит из всех наборов с весом $t-1$, а T'' состоит из всех наборов с весом t, имеет максимальную мощность в множестве $\mathcal{A}'_t$.*

Обозначим через T_0 (T_1) 0-антицепь (1-антицепь) для задачи (4.10). Определим величины t и s следующим образом:

$$t = \min\left\{k \in N : \sum_{i=n-k+1}^{n} w_i > C\right\}, \quad s = t - 1. \tag{4.39}$$

Лемма 4.7 *Если подзадача (4.6) удовлетворяет условию* **C2'**, *то 1-дополнение набора $\{\theta_i | i \in I\}$ будет оптимальным решением подзадачи (4.6).*

Доказательство. Пусть подзадача (4.6) удовлетворяет условию **C2'**. Рассмотрим 1-дополнение $\tilde{\theta}^{(1)}$ набора $\tilde{\theta} = \{\theta_i | i \in I\}$. Согласно определению условия **C1**, $\sum_{i\in I}(1-\theta_i)w_i \geq W - C$. Следовательно $W - \sum_{i\in I}(1-\theta_i)w_i \leq C$. С другой стороны,

$$W - \sum_{i\in I}(1-\theta_i)w_i = \sum_{i\in I}\theta_i w_i + \sum_{i\in N\setminus I} w_i = \sum_{i\in N}\theta_i^{(1)} w_i.$$

Следовательно $\sum_{i\in N}\theta_i^{(1)} w_i \leq C$. Таким образом доказана допустимость решения $\tilde{\theta}^{(1)}$. Его оптимальность является очевидным следствием определения 1-дополнения. Тем самым справедливость доказываемого утверждения установлена. □

Справедливо следующее утверждение

Утверждение 4.21 *Вес любого элемента 0-антицепи T_0 и 1-антицепи T_1 не превосходит t и $\tilde{\gamma}_s \in T_1$.*

Доказательство Рассмотрим 1-набор $\tilde{\alpha} = (\alpha_1, \ldots, \alpha_n)$. Согласно утверждению 4.7 справедливо $\sum_{i=1}^{n}\alpha_i w_i \leq C$. Так как $w_1 \geq \cdots \geq w_n$, то $\sum_{i=n-\|\tilde{\alpha}\|+1}^{n} w_i \leq \sum_{i=1}^{n}\alpha_i w_i$, поэтому $\sum_{i=n-\|\tilde{\alpha}\|+1}^{n} w_i \leq C$. Следовательно, $\|\tilde{\alpha}\| < t$. Рассмотрим теперь 0-набор $\tilde{\beta} = (\beta_1, \ldots, \beta_n)$. Пусть $j = \max\{i \in N : \beta_i = 1\}$. Согласно

определению 0-набора имеем $\sum_{i=1}^{j-1} \beta_i w_i \leq C$. Из неравенств $w_1 \geq \cdots \geq w_n$ вытекает, что $\sum_{i=n-\|\tilde{\beta}\|+2}^{n} w_i \leq \sum_{i=1}^{j-1} \beta_i w_i$. Поэтому $\sum_{i=n-\|\tilde{\beta}\|+2}^{n} w_i \leq C$. Следовательно, $\|\tilde{\beta}\| - 1 < t$, т. е. $\|\tilde{\beta}\| \leq t$.

Докажем теперь принадлежность набора $\tilde{\gamma}_s$ множеству T_1. Рассмотрим подзадачу P с множеством фиксированных переменных $I = \{1, \ldots, n-s-1\}$ и набором значений фиксированных переменных $\{\theta_i | i \in I\}$ таким, что $\theta_i = 0$ для любого $i \in I$. Для данной подзадачи имеем $\sum_{i \in I}(1-\theta_i)w_i = \sum_{i=1}^{n-s-1} w_i = W - \sum_{i=n-t+1}^{n} w_i < W - C$. Таким образом, подзадача P не удовлетворяет условию **C2'**. Очевидно, что P также не удовлетворяет условию **C1**. Таким образом, соответствующая данной подзадаче вершина содержится в дереве ветвления, но не является концевой. Рассмотрим теперь подзадачу P' с множеством фиксированных переменных $I' = \{1, \ldots, n-s\}$ и набором значений фиксированных переменных $\{\theta'_i | i \in I'\}$ таким, что $\theta'_i = 0$ для любого $i \in I'$. Для данной подзадачи имеем $\sum_{i \in I'}(1-\theta'_i)w_i = \sum_{i=1}^{n-s} w_i = W - \sum_{i=n-s+1}^{n} w_i \geq W - C$. Таким образом, подзадача P' удовлетворяет условию **C1** и, очевидно, содержится в разбиении подзадачи P. Следовательно, подзадача P' соответствует концевой вершине дерева ветвления, удовлетворяющей условию **C1**. Набор $\tilde{\gamma}_s$ является 1-дополнением набора значений фиксированных переменных подзадачи P'. Тем самым $\tilde{\gamma}_s$ является 1-набором, т. е. принадлежит множеству T_1.

Из утверждения 4.21, учитывая утверждение 4.19, получаем, что пара антицепей (T_1, T_0) принадлежит множеству $\mathcal{A}_s$ в случае, если $t > \lfloor n/2 \rfloor + 1$, и множеству $\mathcal{A}'_t$ в случае, если $t \leq \lfloor n/2 \rfloor + 1$. Поэтому из теорем 4.4, 4.5 вытекает

Теорема 4.6 *Приведенная сложность $\mathcal{V}$ решения задачи (4.10) мажоритар-*

ным методом ветвей и границ удовлетворяет следующим соотношениям:

$$\mathcal{V} \;\le\; \tbinom{n+1}{t},\ \textit{при}\ t \le \lfloor n/2 \rfloor + 1,$$

$$\mathcal{V} \;\le\; 1 + \tbinom{n+1}{\lfloor n/2 \rfloor + 1} - \tbinom{t}{\lfloor n/2 \rfloor + 1},\ \textit{при}\ t > \lfloor n/2 \rfloor + 1,$$

где t определяется согласно (4.39).

Следствие 4.7 *Сложность $\mathcal{S}$ решения задачи (4.10) мажоритарным методом ветвей и границ удовлетворяет следующим соотношениям:*

$$\begin{aligned} \mathcal{S} \;&\le\; 2\tbinom{n+1}{t} - 1,\ \textit{при}\ t \le \lfloor n/2 \rfloor + 1, \\ \mathcal{S} \;&\le\; 2\left(\tbinom{n+1}{\lfloor n/2 \rfloor + 1} - \tbinom{t}{\lfloor n/2 \rfloor + 1}\right) + 1,\ \textit{при}\ t > \lfloor n/2 \rfloor + 1, \end{aligned} \tag{4.40}$$

где t определяется согласно (4.39).

Сравним три варианта оценки сложности мажоритарного алгоритма. Верхние оценки (4.35) и (4.36) не зависят от способа выбора переменной для ветвления. В работе [156] показано, что выражение (4.15) является верхней оценкой для варианта МВГ при котором для ветвления всегда выбирается переменная, соответствующая предмету с максимальным весом из числа оставшихся, т.е. для мажоритарного алгоритма. Несложно заметить, что для мажоритарного алгоритма верхняя оценка (4.36) всегда будет уступать верхней оценке (4.15). Также легко показать, что оценка (4.40) всегда не хуже (4.15) и при $t \le \lfloor n/2 \rfloor$ не хуже оценки (4.35).

Таблица 4.1 содержит результаты экспериментального сравнения точности этих оценок, усредненных по 1000 вариантам задачи о сумме подмножеств, в которых коэффициенты w_i являются случайными числами в диапазоне $[1, 100]$, а C — случайным числом в диапазоне $[1, \sum_{i=1}^{n} w_i]$. Средняя сложность мажоритарного алгоритма, вычисляемая как отношение суммарного числа итераций 2-4, выполненных для всех задач из серии, к общему числу задач, составила 2114.02. В таблице 4.1 представлены следующие величины.

Среднее значение: значение оценки, усредненное по всем задачам из серии.

Мин. (Макс.) отклонение: минимальное и максимальное отклонение оценки, вычисляемое по следующей формуле: $r = \frac{S'-S}{S}$, где S — экспериментально получено значение сложности, а S' — значение соответствующей верхней оценки.

Лучшая оценка: количество раз, когда данная оценка совпадала с наименьшим значением из трех оценок.

Точная оценка: количество раз, когда оценка в точности совпадала с истинным значением сложности.

Таблица 4.1: Сравнение различных оценок

Оценка	(4.15)	(4.35)	(4.40)
Среднее значение	25739	859985.552	20257.82
Мин. отклонение	0.908	0	0
Макс. отклонение	1224.667	7578.711	761.619
Лучшая оценка	64	72	931
Точная оценка	0	15	3

Выполненное сравнение показывает, что оценка (4.40) превосходит оценки (4.15) и (4.35) по среднему значению, максимальной точности и в подавляющем большинстве случаев была лучшей из всех оценок. В то же время, оценка (4.35) чаще, чем все остальные была точной.

4.7 Основные результаты главы

В данном разделе были рассмотрены вопросы сложности метода ветвей и границ применительно к различным оптимизационным задачам. Были получены следующие результаты:

1. разработан подход к оценке сложности метода неравномерных покрытий, с помощью которого получены верхние оценки сложности различных вариантов этого метода для различных классов задач;

2. показано, что при выборе дробной переменной в задаче релаксации в качестве переменной для ветвления, максимальная вычислительная сложность может превосходить сложность примера Финкельштейна в полтора раза более;

3. получены две верхние оценки максимальной сложности метода ветвей и границ для задачи о ранце;

4. получена верхняя оценка сложности для мажоритарного алгоритма задачи о ранце;

5. проведено экспериментальное сравнение точности различных оценок.

Глава 5

Параллельная вычислительная сложность метода ветвей и границ

5.1 Постановка задачи и обзор существующих результатов

В настоящее время параллельные вычисления [162, 163] получили широкое распространение. Поэтому актуальным становится вопрос эффективного портирования основных вычислительных схем на параллельные архитектуры. В данной главе будет рассмотрена одна из основных вычислительных схем организации параллельных вычислений — метод ветвей и границ. Параллельной реализации МВГ и родственных схем посвящено много литературы. Следует отметить работы Ю.Г. Евтушенко [48], Р.Г. Стронгина, В.П. Гергеля, Я.Д. Сергеева [164–166], Б.О. Онищенко [167]. Также большое число работ по данной тематике опубликовано зарубежными авторами, обзор которых можно посмотреть в работах [168–170].

Вопросам параллельной сложности тоже уделялось внимание в различных публикациях. Укажем статью А.В. Плясунова [85], в которой рассмотрены теоретические основы параллельной сложности и исследована параллельная сложность решения некоторых оптимизационных задач эвристическими методами. Отметим также ряд работ зарубежных авторов [171, 172].

В данной главе изучается одна из возможных параллельных реализаций ме-

тода ветвей и границ (МВГ), называемая *фронтальным алгоритмом*. Из всего множества процессоров выделяется *управляющий процессор*, который на первом этапе выполняет некоторое число шагов МВГ. На втором этапе полученные подзадачи рассылаются *рабочим процессорам* — по одной на процессор. Процессоры решают подзадачи методом ветвей и границ полностью. Управляющий процессор принимает результаты расчетов и выбирает из них оптимум. Описанная схема организации вычислений идеально подходит для высокопроизводительных систем с ограниченными коммуникациями. К такому классу относятся различные грид-системы, получившие широкое распространение сегодня [173]. Ограниченные возможности взаимодействия параллельных потоков также имеют место в специализированных графических вычислителях общего назначения (GP-GPU).

Возможен также вариант, когда начальный этап выполняется не только на управляющем, но и повторяется на всех рабочих процессорах. Далее каждый из рабочих процессоров выбирает нужную ему подзадачу в соответствии с некоторым правилом. При таком подходе не требуется начальной рассылки подзадач рабочим процессорам. Этот вариант целесообразно применять в системах, в которых все процессоры аллокируются одновременно на некоторый интервал времени. В таких системах во время работы управляющего процессора рабочие процессоры простаивают. Примером подобных систем являются графические вычислители, а также кластеры общего доступа, в которых группа процессоров аллокируется на некоторый заданный пользователем промежуток времени.

5.2 Фронтальный параллельный вариант реализации метода ветвей и границ

Рассмотрим следующую реализацию МВГ для многопроцессорных систем, которую назовем *фронтальным параллельным МВГ* или просто *фронтальным*

алгоритмом. Предположим, что имеется бесконечное число одинаковых процессоров. Выделим один процессор и назовем его *управляющим*. Остальные процессоры будем называть *рабочими*. На начальном этапе управляющий процессор выполняет некоторое количество итераций МВГ. Пусть после завершения начального этапа на управляющем процессоре получено v вершин, соответствующих задачам-кандидатам. Далее v вершин рассылается с управляющего процессора v рабочим процессорам (по одной на процессор). Каждый рабочий процессор, получивший подзадачу, полностью решает ее. После завершения расчетов рабочие процессоры пересылают наилучшие найденные решения управляющему процессору, который, сопоставляя полученные результаты, выбирает оптимальный.

Определим теперь понятие сложности фронтального параллельного МВГ. Пусть управляющий процессор проделал u итераций МВГ. Обозначим через w общее число вершин в соответствующем дереве ветвления, а через v — число подзадач, переданных для обработки рабочим процессорам. Тогда $u = w - v$. Пусть максимальная из сложностей подзадач, переданных рабочим процессорам, равна q. Время решения задачи на рассматриваемой параллельной системе складывается из времени работы управляющего процессора, пропорционального числу итераций u, выполненных на нем, и максимального времени решения разосланных подзадач на рабочих процессорах. Для более точной оценки необходимо также учесть время, необходимое для рассылки подзадач, и время, которое требуется на сбор результатов расчетов. Рассмотрим идеальную ситуацию, когда это время пренебрежимо мало по сравнению со временем t одного шага МВГ. Тогда общее время решения задачи на рассмотренной параллельной системе составит $L \cdot t$, где $L = u + q$. Исходя из проведенных рассуждений, естественно положить *параллельную сложность фронтального алгоритма* равной введенной величине L.

5.3 Проблемно-независимая оценка сложности фронтального алгоритма

Справедлива следующая лемма

Лемма 5.1 *Пусть дано дерево с общим числом вершин, равным l, из которых k вершин — концевые. Пусть также D — максимальная степень вершины в этом дереве, $D \geq 3$. Тогда справедливо следующее соотношение:*

$$l \geq \frac{k(D-1)-2}{D-2}. \tag{5.1}$$

Доказательство. Известно, что число ребер дерева с l вершинами составляет $l-1$ и что сумма степеней всех вершин в графе вдвое больше числа его ребер. Обозначим через $p(u)$ - степень вершины u. Пусть V — множество всех вершин графа. Тогда справедливо соотношение $\sum_{u \in V} p(u) = 2(l-1)$. Так как из концевых вершин выходит в точности одно ребро, а степень остальных вершин ограничена величиной D, то $\sum_{u \in V} p(u) \leq k + (l-k)D$. Следовательно, $2(l-1) \leq k + (l-k)D$. Преобразуя данное неравенство, получаем соотношение $l(D-2) > (D-l)k - 2$, из которого непосредственно вытекает справедливость доказываемого утверждения. □.

Если процесс разбиения подзадач устроен таким образом, что в результате разбиения образуется не более d новых подзадач, тогда, согласно лемме , имеет место неравенство $w \geq (vd-2)/(d-1)$, откуда следует:

$$v \leq \frac{w(d-1)+2}{d}. \tag{5.2}$$

Далее будем предполагать, что общее число S вершин в дереве ветвления не изменяется в результате параллельного выполнения. Очевидно, что общее число вершин, обработанных рабочими процессорами, составляет $S-(w-v)$. Учитывая, что максимальная сложность одной из v подзадач, разосланных рабочим процессорам, составляет q, получим $qv \geq S - w + v$. Следовательно,

справедливо соотношение $q \geq (S - w + v)/v$:

$$q \geq (S - w + v)/v = \frac{S-w}{v} + 1 \geq \frac{(S-w)d}{w(d-1)+2} + 1 = \tag{5.3}$$

$$= (Sd - w + 2)/(w(d-1) + 2).$$

Следовательно,

$$p = w - v + q \geq w - \frac{w(d-1)+2}{d} + (Sd - w + 2)/(w(d-1) + 2) =$$

$$= \frac{w-2}{d} + (Sd - w + 2)/(w(d-1) + 2).$$

Введем обозначение $z(w) = \frac{w-2}{d} + (Sd - w + 2)/(w(d-1) + 2)$.

Для нахождения минимума функции $z(w)$ вычислим производную

$$\frac{dz(w)}{dw} = \frac{1}{d} - \frac{d(S(d-1)+2)}{(w(d-1)+2)^2}.$$

Уравнение $\frac{dz(w)}{dw} = 0$ имеет единственный положительный корень w_*:

$$w_* = \left(d\sqrt{S(d-1)+2} - 2\right)/(d-1).$$

Следовательно, минимум z_* функции $z(w)$ достигается в точке w_*. Для вычисления этого минимума, представим $z(w)$ в виде суммы двух функций $z_1(w) = (w-2)/d$ и $z_2(w) = (Sd - w + 2)/(w(d-1) + 2)$. Вычислим значения обоих слагаемых в точке w_*:

$$z_1(w_*) = (w_* - 2)/d = \left(\left(d\sqrt{S(d-1)+2} - 2\right)/(d-1) - 2\right)/d =$$

$$= \left(d\sqrt{S(d-1)+2} - 2 - 2d + 2\right)/(d(d-1)) =$$

$$= \left(\sqrt{S(d-1)+2} - 2\right)/(d-1).$$

$$z_2(w_*) = \frac{Wd - w_* + 2}{w_*(d-1)+2} = \frac{Sd - \left(d\sqrt{S(d-1)+2} - 2\right)/(d-1) + 2}{d\sqrt{S(d-1)+2}} =$$

$$= \frac{Sd(d-1) - d\sqrt{S(d-1)+2} + 2 + 2d - 2}{d(d-1)\sqrt{S(d-1)+2}} = \frac{S(d-1) - \sqrt{S(d-1)+2} + 2}{(d-1)\sqrt{S(d-1)+2}}.$$

Суммируя полученные выражения, получим:

$$z(w_*) = \frac{1}{d-1}\left(\sqrt{S(d-1)+2} - 2 + \frac{S(d-1)-\sqrt{S(d-1)+2}+2}{\sqrt{S(d-1)+2}}\right) =$$

$$= \frac{2(S(d-1)+2)-3\sqrt{S(d-1)+2}}{(d-1)\sqrt{S(d-1)+2}} = \frac{2\cdot\sqrt{S(d-1)+2}-3}{d-1}.$$

Таким образом, для сложности L параллельного фронтального МВГ имеет место следующая нижняя оценка

$$L \geq \frac{2\sqrt{S(d-1)+2}-3}{d-1}. \tag{5.4}$$

В распространенном частном случае дихотомического ветвления $d = 2$ соотношение (5.4) приобретает более простой вид:

$$L \geq 2\sqrt{S+2} - 3. \tag{5.5}$$

Таким образом, в предположении, что число вершин дерева ветвления не меняется при переходе к параллельному варианту, справедливо следующее утверждение.

Утверждение 5.1 *Сложность фронтального алгоритма L ограничена снизу величиной $2\sqrt{S+2}-3$, где S — сложность последовательного алгоритма, а ускорение Sp подчиняется неравенству $Sp \leq \frac{S}{2\sqrt{S+2}-3}$.*

5.4 Оценка параллельной сложности ярусного фронтального алгоритма для задачи о сумме подмножеств

В разделе 4 показано, что сложность, определяемая числом шагов МВГ совпадает с общим количеством вершин в дереве ветвлений.

Глубиной вершины в дереве ветвлений будем называть число *дуг* в пути, соединяющем корень дерева ветвлений с данной вершиной. Корневая вершина имеет нулевую глубину. *Глубиной дерева* будем называть максимальную из глубин его вершин. Множество вершин с одинаковой глубиной назовем *ярусом*

дерева. Дерево, все концевые вершины которого имеют одинаковую глубину d, назовем *фронтальным*.

Фронтальным деревом будем называть дерево ветвлений, в котором вершина обозначена белым цветом тогда и только тогда, когда она имеет глубину, равную глубине дерева.

Рассмотрим следующую реализацию метода ветвей и границ для многопроцессорных систем, которую назовем *ярусным фронтальным алгоритмом*. Предположим, что имеется бесконечное число одинаковых процессоров. Выделим один процессор и назовем его *управляющим*. Остальные процессоры будем называть *рабочими*. На начальном этапе управляющий процессор выполняет некоторое количество итераций МВГ, таким образом, чтобы текущее дерево ветвлений после последней итерации было фронтальным деревом заданной глубины $s > 0$. Пусть на управляющем процессоре получено v вершин, соответствующих задачам-кандидатам. Каждый рабочий процессор, получивший подзадачу, полностью решает ее. После завершения расчетов рабочие процессоры пересылают наилучшие найденные решения управляющему процессору, который, сопоставляя данные решения, находит оптимум.

Определим теперь понятие сложности фронтального ярусного алгоритма. Обозначим число шагов метода ветвей и границ, выполненных на управляющем процессоре, через L_1, а максимальную сложность решения любой из v сгенерированных задач-кандидатов — через L_2. Время решения задачи на рассматриваемой параллельной системе складывается из времени решения на управляющем процессоре, пропорционального L_1, и максимального из времен решения v разосланных подзадач на рабочих процессорах, пропорционального L_2. Для более точной оценки необходимо также учесть время, необходимое для рассылки подзадач, и время, которое требуется на сбор результатов расчетов. Рассмотрим идеальную ситуация, когда время коммуникаций пренебрежимо мало по

сравнению со временем t одного шага метода ветвей и границ. Тогда общее время решения задачи на рассмотренной параллельной системе составит $L \cdot t$, где $L = L_1 + L_2$. Исходя из проведенных рассуждений естественно положить *сложность фронтального ярусного алгоритма* равной введенной величине L.

Следуя [174] будем говорить, что последовательность a_n *эквивалентна* последовательности b_n при $n \to \infty$ и писать $a_n \sim b_n$, если найдется такая последовательность c_n, что $\lim_{n\to\infty} c_n = 1$ и $a_n = c_n b_n$ для всех $n \in \mathbb{N}$. Будем говорить также, что последовательности a_n и b_n одного порядка и писать $a_n \asymp b_n$, если $a_n = O(b_n)$ и $b_n = O(a_n)$.

В дальнейшем нам потребуются следующие свойства биномиальных коэффициентов (см. например [175]). Здесь и далее через $\lfloor a \rfloor$ обозначается наибольшее целое число, не превосходящее a.

Утверждение 5.2 *Последовательность чисел* $\left\{\binom{n}{k}\right\}$, $k = 0, 1, \ldots, n$ *унимодальна. Положим* $k_n = \lfloor n/2 \rfloor$. *При четном* n *максимум достигается в одной точке* $k_n = n/2$, *а при нечетном — в двух точках* $k_n = \lfloor n/2 \rfloor$ *и* $k_n + 1$.

Утверждение 5.3 $\sum_{i=0}^{n} \binom{n}{i} = 2^n$.

Из утверждений 5.2 и 5.3 и того, что $\binom{n}{k} = \binom{n}{n-k}$ при $k = 0, 1, \ldots, n$, следует что

$$2^{n-1} \leq \sum_{i=0}^{l} \binom{n}{i} \leq 2^n \text{ для любого } l, \;\; n/2 \leq l \leq n. \tag{5.6}$$

Задача о сумме подмножеств [157] является частным случаем задачи о булевом ранце с одним ограничением и формулируется следующим образом:

$$\begin{cases} f(x) = \sum_{i=1}^{n} w_i x_i \to max; \\ \text{при условиях} \\ \sum_{i=1}^{n} w_i x_i \leq C; \end{cases} \tag{5.7}$$

где $x = (x_1, \ldots, x_n)$ — вектор булевых переменных.

В данной постановке через w_i обозначен вес i-го предмета, помещаемого в ранец грузоподъемностью C. Задача состоит в определении набора предметов максимального суммарного веса, который можно разместить в ранце. При этом обычно предполагаются справедливыми неравенства $w_i \leq C$, $\sum_{i=1}^{n} w_i > C$.

Стандартный метод ветвей и границ для задачи о ранце подробно описан в [159]. Декомпозиция (ветвление) состоит в разбиении подзадачи на две путем присваивания одной из переменных, называемой переменной ветвления, значений 0 и 1 соответственно. Здесь и далее ограничимся рассмотрением варианта метода, при котором значения переменным присваиваются в порядке их нумерации. Пусть некоторая подзадача получена из исходной задачи последовательным присвоением переменным $x_1, \ldots, x_r$ булевых значений $\theta_1, \ldots, \theta_r$. Вектор $\tilde{\theta} = (\theta_1, \ldots, \theta_r)$ назовем *вектором фиксированных переменных* для данной подзадачи. Для удобства описания метода ветвей и границ введем в рассмотрение более общую постановку задачи (5.7) , в которой предполагается, что первые r переменных принимают фиксированные значения:

$$\begin{cases} \sum\limits_{i=1}^{n} w_i x_i \to max; \\ \text{при условиях} \\ \sum\limits_{i=1}^{n} w_i x_i \leq C; \\ x_i = \theta_i, 1 \leq i \leq r, \\ x_i \in \{0,1\}, r+1 \leq i \leq n. \end{cases} \tag{5.8}$$

Отметим, что для задача (5.7)является частным случаем задачи (5.8) , когда $r = 0$.

Помимо ветвления в процессе работы алгоритма происходит обновление рекордного решения $\hat{x}$. Первоначально полагают $\hat{x} = \{0, \ldots, 0\}$. Если x — допустимая (т.е. удовлетворяющая ограничению) точка задачи (5.7), то рекорд обновляется по следующему правилу

правило **REC-UPDATE:** если $f(x) > f(\hat{x})$, то положить $\hat{x} := x$.

Обозначим через $W = \sum\limits_{i=1}^{n} w_i$ — суммарный вес всех предметов, $R = W - C$. Сформулируем следующие правила исключения вершин из дальнейшего поиска:

C1: $\sum\limits_{i=1}^{r} w_i\theta_i \geq C$;

C2: $\sum\limits_{i=1}^{r} w_i(1-\theta_i) \geq R$;

C3: $r = n$.

Первое правило означает, что суммарный вес предметов, соответствующих фиксированным переменным, уже превзошел грузоподъемность C, и поэтому рассматриваемая подзадача не имеет допустимых точек, а следовательно и не содержит оптимального решения. При этом набор x, где

$$x_i = \begin{cases} \theta_i, 1 \leq i \leq r-1, \\ 0, r \leq i \leq n, \end{cases}$$

является допустимой точкой исходной задачи и рекорд может быть обновлен по правилу **REC-UPDATE**. Если выполнено правило C2, то

$$\sum_{i=1}^{r} w_i\theta_i + \sum_{i=r+1}^{n} w_i = W - \sum_{i=1}^{r} w_i(1-\theta_i) \leq W - R = C.$$

Это означает, что набор x, где

$$x_i = \begin{cases} \theta_i, 1 \leq i \leq r-1, \\ 1, r \leq i \leq n, \end{cases}$$

будет оптимальным решением подзадачи (5.8) и допустимым решением исходной задачи. Поэтому дальнейшее исследование данной подзадачи нецелесообразно и его следует прекратить, обновив рекорд по правилу **REC-UPDATE**.

Правило отсева C3 означает, что все переменные фиксированы и дальнейшее ветвление невозможно. Заметим, что это правило может быть исключено.

Действительно, если не выполнены правила С1, С2, то

$$\sum_{i=1}^{r} w_i < C + R = W.$$

Следовательно, $r < n$. Значит, $r = n$ только если выполнено хотя бы одно из правил 1,2. Несложно показать, что после завершения работы МВГ с описанными выше операциями, найденное рекордное решение $\hat{x}$ будет оптимумом задачи (5.7). Особенностью задачи о сумме подмножеств является то, что если значение целевой функции $f(\hat{x})$ в рекордной точке $\hat{x}$ стало равным C, то процесс решения может быть прекращен, так как $\hat{x}$ очевидно будет оптимальным решением задачи (5.7).

Оценку сложности проведем для следующей серии задач:

$$\begin{cases} \sum\limits_{i=1}^{n} ax_i \to \max, \\ \text{при условиях} \\ \sum\limits_{i=1}^{n} ax_i \le ka + 1, \\ x_i \in \{0,1\}, i = 1, \ldots, n, \end{cases} \tag{5.9}$$

где a — произвольное действительное число, большее 1. Эта серия была выбрана для исследования в силу ее относительной простоты и хорошей изученности. Равенство коэффициентов позволяет получить точные формулы для сложностей решения задач из этой серии. Сложность решения задачи (5.9) составляет $2\binom{n+1}{k+1} - 1$. Для задачи (5.9) введенные величины L_1, L_2 и L полностью определяются глубиной s начального дерева ветвлений и могут быть рассмотрены как функции этого параметра.

Для задачи (5.9) определим величины L_* и s_* как минимальное значение функции $L(s)$ и аргумент, на котором этот минимум достигается:

$$L_* = L(s_*) = \min_{s=1,\ldots,n} L(s).$$

Заметим, что s_* может определяется неоднозначно. Далее в работе оценивается порядок роста с увеличением n величины L_* при условии $n/3 \le k \le n/2$.

Правила отсева для данной задачи записываются следующим образом:

C1: $r_1 \ge k+1$,

C2: $r_0 \ge n-k$,

где r — число фиксированных переменных данной задачи, r_1 — количество единиц в фиксированном наборе значений переменных, r_0 — количество нулей, $r_0 + r_1 = r$.

Так как глубина дерева ветвлений на управляющем процессоре составляет s, то у всех задач-кандидатов зафиксированы значения первых s переменных. Задачи-кандидаты на уровне s имеют по $n-s$ переменных, а ограничение по весу выражается формулой $C_j = ja + 1$, где $k - s \le j \le k$. Сложность задачи с ограничением C_j составляет $2\binom{n-s+1}{j+1} - 1$. Таким образом $L_2(s) = \max_{k-s \le j \le k} \left(2\binom{n-s+1}{j+1} - 1\right)$.

В дальнейшем будет использоваться следующая лемма.

Лемма 5.2 *Если* $s \le \min(k+1, n/2)$, *то* $L_1(s) = 2^s - 1$.

Доказательство. Для всех подзадач на ярусах $0, \ldots, s-1$ справедливо $r_1 \le r \le s-1 \le k$ и $r_0 \le r \le s-1 \le n/2-1 < n/2 < n-k$. Таким образом, к любой подзадаче на ярусе $0 \le s-1 \le k$ правило отсева не применимо, следовательно, $L_1(s) = 2^s - 1$. $\square$

Рассмотрим два случая: $s \le n-2k$ и $s \ge n-2k$.

Случай 1. Пусть $s \le n-2k$. В этом случае поскольку $n/3 \le k \le n/2$, то $s \le n-2k \le n-2n/3 = n/3 \le k$. Поэтому, согласно лемме 5.2, $L_1(s) = 2^s - 1$. Покажем, что $L_2(s) = 2\binom{n-s+1}{k+1} - 1$. Для этого рассмотрим два подслучая: $s =$

$n-2k$ и $s \le n-2k-1$. Согласно утверждению 5.2, при $s = n-2k$ выполняется

$$\begin{aligned} L_2(s) &= \max_{k-s\le j\le k}\left(2\binom{n-s+1}{j+1}-1\right) = \max_{k-s\le j\le k}\left(2\binom{2k+1}{j+1}-1\right) \\ &= 2\binom{2k+1}{k+1}-1 = 2\binom{n-s+1}{k+1}-1. \end{aligned}$$

При $s \le n-2k-1$ справедливо $k+1 \le (n-s+1)/2$. Из утверждения 5.2 следует, что в этом случае $L_2(s) = \max_{k-s\le j\le k}\left(2\binom{n-s+1}{j+1}-1\right) = 2\binom{n-s+1}{k+1}-1$. Таким образом, при $s \le n-2k$ выполняется $L(s) = 2^s + 2\binom{n-s+1}{k+1} - 2$.

Лемма 5.3 *Для любого целого s, $1 \le s \le n-2k$ выполняется неравенство $L(s-1) \ge L(s)$ и $s_* \ge n-2k$.*

Доказательство. Рассмотрим величину $\Delta(s) = L(s) - L(s-1)$. Для нее справедливы соотношения

$$\Delta(s) = 2^{s-1} + 2\left(\binom{n-s+1}{k+1} - \binom{n-s+2}{k+1}\right) = 2^{s-1} - 2\binom{n-s+1}{k}.$$

Заметим, что

$$\binom{n-s+1}{k} = \frac{(n-s+1)!}{k!(n-s+1-k)!} = \prod_{i=1}^{k}\frac{n-s+1-k+i}{i} \ge \left(\frac{n-s+1}{k}\right)^k.$$

Так как $s \le n-2k$, то $n-s+1 \ge 2k$. Поскольку $n/3 \le k \le n/2$, то $s \le n-2k \le n-2n/3 = n/3 \le k$. Таким образом, $\left(\frac{n-s+1}{k}\right)^k \ge 2^s$. Следовательно, $\Delta(s) \le 0$. $\square$.

Согласно лемме 5.3 имеет место монотонное не возрастание функции $L(s)$ в интервале $1 \le s \le n-2k$ и, следовательно, $L(n-2k) = \min_{1\le s\le n-2k} L(s)$. Поэтому $s_* \ge n-2k$.

Случай 2. Пусть $n-2k \le s \le n/2$. Так как $s \ge n-2k+1$, то $k+1 \ge (n-s)/2+1 > (n-s+1)/2$. Согласно утверждению 5.2 получим

$$L_2(s) = \max_{k-s\le j\le k}\left(2\binom{n-s+1}{j+1}-1\right) = 2\binom{n-s+1}{\lfloor (n-s+1)/2\rfloor}.$$

Рассмотрим два подслучая: $s \le k+1$ и $k+2 \le s \le n/2$. Если $s \le k+1$, то, согласно лемме 1, $L_1(s) = 2^s - 1$. Пусть $s > k+1$. Обозначим через B число черных вершин в дереве T_1. При $k+2 \le s \le n/2$ черные вершины соответствуют подзадачам, у которых вектор фиксированных переменных имеет длину, не превосходящую $s-1$, содержит ровно $k+1$ единичных компонент, при этом последняя компонента является единичной (отсев по правилу C2 не происходит, так как $j_0 \le s-1 < n/2 \le n-k$). Нетрудно убедиться, что число таких векторов равно $\binom{s-1}{k+1}$. Поэтому $B = \binom{s-1}{k+1}$. Число W белых вершин в T_1 может быть вычислено по формуле $W = 2G'$, где G' — число серых вершин на ярусе $s-1$. Серые вершины на уровне $s-1$ соответствуют подзадачам, векторы фиксированных переменных которых имеют длину $s-1$ и содержат не более k единичных компонент. Поэтому

$$G' = \sum_{i=0}^{k} \binom{s-1}{i}.$$

Общее число G серых (внутренних) вершин в дереве T_1 определяется формулой $G = W + B - 1$. Таким образом $L_1(s) = G + B = W + 2B - 1$. Подставляя в эту формулу найденные выражения для W и B, получим

$$L_1(s) = 2\sum_{i=0}^{k+1} \binom{s-1}{i} - 1.$$

Таким образом, при $n - 2k \le s \le n/2$

$$L(s) = \begin{cases} 2^s + 2\binom{n-s+1}{\lfloor (n-s+1)/2 \rfloor} - 2 \text{ при } s \le k+1, \\ \\ 2\sum\limits_{i=0}^{k+1} \binom{s-1}{i} + 2\binom{n-s+1}{\lfloor (n-s+1)/2 \rfloor} - 2 \text{ в противном случае.} \end{cases}$$

Так как $k+1 > n/3 > (s-1)/2$, то согласно (5.6), имеем

$$2^{s-2} \le \sum_{i=0}^{k+1} \binom{s-1}{i} \le 2^{s-1}.$$

Таким образом $L(s) = \theta(s)2^s + 2\binom{n-s+1}{\lfloor (n-s+1)/2 \rfloor} - 2$, где $\frac{1}{2} \le \theta(s) \le 1$.

По формуле Валлиса

$$\binom{n-s+1}{\lfloor (n-s+1)/2 \rfloor} \sim \frac{2^{n-s+1}}{\sqrt{\pi(n-s+1)/2}} \sim \frac{2}{\sqrt{\pi/2}} \cdot \frac{2^{n-s}}{\sqrt{n-s}}.$$

Таким образом при достаточно больших n

$$L(s) = \theta(s)2^s + \phi(s)\frac{2^{n-s}}{\sqrt{n-s}},$$

где $\frac{1}{2} \le \theta(s) \le 1$, $3 \le \phi(s) \le 4$.

Рассмотрим введенные ранее величины s_* и L_* как функции от числа переменных n и покажем, что

$$L_*(n) \asymp \frac{2^{n/2}}{\sqrt[4]{n}}, \quad s_*(n) = n/2 - \frac{1}{4}\log_2 n + O(1),$$

Подставим выражение $t_1(n) = \lfloor n/2 - \frac{1}{4}\log_2 n \rfloor$ в формулу для $L(s)$:

$$L(t_1(n)) \le \theta(t_1(n))2^{n/2-\frac{1}{4}\log_2 n} + \phi(t_1(n))\frac{2^{n/2+\frac{1}{4}\log_2 n+1}}{\sqrt{n/2}} =$$
$$= \frac{2^{n/2}}{\sqrt[4]{n}}\left(\theta(t_1(n)) + 2\sqrt{2}\,\phi(t_1(n))\right) \le 13\frac{2^{n/2}}{\sqrt[4]{n}}.$$

Докажем, что для любого целого s из множества $[1, t_1(n)-3] \cup [t_1(n)+6, n/2]$ выполнено $L(s) > L(t_1(n))$. Пусть $s \le \lfloor n/2 - \frac{1}{4}\log_2 n \rfloor - 3$. Тогда

$$L(s) > \phi(s)\frac{2^{n-s}}{\sqrt{n}} > 3\frac{2^{\frac{n}{2}+\frac{1}{4}\log_2 n+3}}{\sqrt{n}} = 24\frac{2^{n/2}}{\sqrt[4]{n}}.$$

Если $s \ge \lfloor n/2 - \frac{1}{4}\log_2 n \rfloor + 6$, то имеем

$$L(s) > \theta(s)2^s > \frac{1}{2}2^{n/2-\frac{1}{4}\log_2 n+5} = 16\frac{2^{n/2}}{\sqrt[4]{n}}.$$

Рассмотрим теперь случай $s \ge n/2$, для которого справедливо $L_1(s) \ge L_1(\lfloor \frac{n}{2} \rfloor) \ge 2^{n/2-2}$. Полученное значение по порядку роста превосходит величину $\frac{2^{n/2}}{\sqrt[4]{n}}$. Таким образом верна следующая теорема.

Теорема 5.1 *При $n/3 \le k \le 2n/3$ справедливы соотношения:*

$$L_*(n) \asymp \frac{2^{n/2}}{\sqrt[4]{n}},$$

$$s_*(n) = n/2 - \frac{1}{4}\log_2 n + O(1).$$

Сложность решения задачи (5.9) на одном процессоре составляет $Z(n) = 2\binom{n+1}{k+1} - 1$. Для оценки асимптотического поведения положим $k = \lfloor \alpha n \rfloor$, $1/3 \leq \alpha \leq 1/2$. Из формулы Стирлинга [175] получим:

$$\binom{n}{k} \asymp \frac{\sqrt{n}}{\sqrt{2\pi k(n-k)}} \cdot \frac{n^n}{k^k (n-k)^{n-k}} = \frac{1}{\sqrt{2\pi\alpha(1-\alpha)n}} \cdot \frac{1}{(\alpha^\alpha (1-\alpha)^{1-\alpha})^n}.$$

Так как $\binom{n+1}{k+1} = \frac{n+1}{k+1} \cdot \binom{n}{k} \asymp \frac{1}{\alpha}\binom{n}{k}$, то $Z(n) \asymp \frac{\beta^n}{\sqrt{n}}$, где $\beta = 1/(\alpha^\alpha (1-\alpha)^{1-\alpha})$.

По порядку ускорение оптимального параллельного варианта по отношению к последовательному составляет

$$\mathbb{S}(n) = \frac{Z(n)}{L_*(n)} \asymp \left(\frac{\beta}{\sqrt{2}}\right)^n n^{-\frac{1}{4}}.$$

Несложно показать, что при $\alpha \in [1/3, 1/2]$ выполняется неравенство $2 \geq \beta > \sqrt{2}$. Следовательно, при любом α ускорение растет с ростом размерности задачи. При этом максимум $\beta = 2$ достигается в единственной точке $\alpha = 1/2$.

Количество процессоров $p(n)$, задействованных в вычислениях, совпадает с числом белых вершин в момент завершения последовательной части алгоритма:

$$p(n) = 2\sum_{i=0}^{k} \binom{s_*(n) - 1}{i}.$$

Так как $k \geq n/3 > (s_*(n) - 1)/2$, то, согласно (5.6), имеем $p(n) \asymp 2^{s_*(n)} \asymp 2^{n/2} \cdot n^{-1/4}$. Рассмотрим выражение для параллельной эффективности

$$\mathbb{E}(n) = \frac{\mathbb{S}(n)}{p(n)} = (\beta/2)^n.$$

При $\beta = 2$ параллельная эффективность максимальная и по порядку является константой $\mathbb{E}(n) = O(1)$. При любом другом значении α параллельная эффективность убывает с ростом n. Таким образом, эффективность использования вычислительного ресурса будет падать при росте размерности для любых α, отличных от $1/2$.

Полученные в данной работе асимптотические формулы показывают, что сложность фронтального алгоритма по порядку одинакова для разных задач из серии (5.9) с одним и тем же числом переменных. Из этих формул также вытекает, что с ростом размерности задачи при фиксированном отношении вместимости ранца к сумме весов эффективность сохраняется постоянной только тогда, когда это отношение равно $1/2$, в других же случаях она падает.

Заметим, что при $k = \lfloor n/2 \rfloor$ сложность $S(n)$ последовательного варианта имеет асимптотику $S(n) \asymp \frac{2^n}{\sqrt{n}}$, а параллельная сложность $L_*(n) \asymp \frac{2^{n/2}}{\sqrt[4]{n}}$. Следовательно, $L_*(n) \asymp \sqrt{S(n)}$, т.е. нижняя оценка из теоремы 5.1 не может быть асимптотически улучшена для данной задачи.

5.5 Основные результаты главы

В данной главе исследован вопрос параллельной вычислительной сложности одного из распространенных вариантов параллельной реализации метода ветвей и границ. Для этого рассмотрена гипотетическая система с неограниченным параллелизмом (число процессоров неограниченно) и бесконечной скоростью обмена данными. Исследован фронтальный параллельный алгоритм и его частный случай — ярусный параллельный алгоритм. Получена нижняя оценка параллельной сложности фронтального алгоритма для любого варианта метода ветвей и границ.

Получена асимптотика минимальной параллельной вычислительной сложности ярусного фронтального алгоритма, примененного для решения задачи о сумме подмножеств. Рассмотрена серия задач с равными коэффициентами и различной правой частью в ограничении. Показано, что масштабируемость имеет место только для частного случая, когда правая часть ограничения равняется примерно половине суммы коэффициентов. В остальных случаях имеет место деградация параллельной эффективности с ростом размерности задачи.

Глава 6

Программные комплекс для решения задач оптимизации на последовательных и параллельных вычислительных системах

6.1 Постановка задачи и обзор существующих результатов

Данная глава посвящена разработке и реализации программного комплекса для решения задач оптимизации на однопроцессорных и многопроцессорных ЭВМ, который был использован для практической апробации подходов и методов, предложенных в диссертационной работе. Целью разработки программного комплекса являлось создание универсальной расширяемой программной инфраструктуры, обладающей следующими свойствами для решения задач глобальной оптимизации.

Тематике разработки программных комплексов для решения задач глобальной оптимизации уделяется большое внимание в настоящее время. Нередко оптимизационные программы разрабатываются специально для решения конкретных прикладных [176–179] или специальных достаточно узких классов типовых модельных [48,180–182] задач. Такой подход оправдан, когда цель исследования ограничивается необходимостью решения одной конкретной задачи или проведением расчетов для задач определенного класса, например задачи о ранце. В

случае, когда требуется решать различные задачи, то разрабатываются методы более широкого назначения, которые реализуются в рамках программных комплексов, поддерживающих достаточно широкий спектр методов оптимизации. В дальнейшем в обзоре уделим основное внимание детерминированным алгоритмам решения задач глобальной оптимизации, в частности, подходам, основанным на методе ветвей и границ.

Значительная часть оптимизационных пакетов, использующих парадигму метода ветвей и границ, ориентированы на решение целочисленных или частично-целочисленных задач линейного программирования. Программная платформа ABACUS [183] предназначена для решения задач комбинаторной оптимизации с помощью различных вариантов методов ветвей и отсечений (branch-and-cut, branch-cut-and-price). Используя механизм наследования, система позволяет разработчику не только задавать исходные данные задачи, но и определять свои методы декомпозиции подзадач, контейнеры для хранения подзадач и отсечений, использовать различные программные пакеты для решения задач линейной релаксации. Была разработана параллельная версия системы ABACUS [184], предназначенная для параллельных систем с общей памятью. При этом использовалась асинхронная децентрализованная стратегия балансировки нагрузки между параллельными потоками. Если судить по публикациями, данная разработка не получила дальнейшего развития и современная версия пакета ABACUS является последовательной. Библиотека MINTO [185] представляет собой другой программный пакет для решения задач частично-целочисленного линейного программирования методами ветвей и границ. Также как и ABACUS, библиотека MINTO допускает расширение пользовательскими процедурами декомпозиции задачи на подзадачи, построения отсечений и некоторых других типов действий. Отметим также инструмент CPLEX [186], предоставляющий возможности для решения задач частично-целочисленного

линейного и квадратичного программирования. Обзор других оптимизационных пакетов для решения задач частично-целочисленного линейного программирования можно найти в работе [187].

Также разрабатывались библиотеки программ, предназначенные для решения более общего класса задач оптимизации. Программный комплекс ДИСО [188], разработанный в Вычислительном центре Академии наук СССР, был ориентирован на решения задач конечномерной оптимизации широкого класса. Библиотека включала в себя методы решения задач безусловной оптимизации, нелинейного программирования [189] и многокритериальной оптимизации [190]. В нем были реализованы как локальные, так и глобальные методы поиска экстремума. В системе были предусмотрены пакетный и диалоговый режимы работы. Поскольку ДИСО появился до эпохи широкого внедрения параллельных вычислительных систем, то в нем поддерживался только последовательный вариант проведения расчетов.

Начиная с 90-х годов прошлого века, многопроцессорные вычислительные комплексы стали активно использоваться в научных и технических расчетах. Эра многоядерных процессоров сделала параллельные вычисления повсеместно распространенными. Поэтому многие современные библиотеки методов оптимизации ориентированы как на последовательные, так и на параллельные системы. Рассмотрим примеры таких пакетов.

Библиотека DAKOTA [191] предназначен для решения задач инженерной оптимизации. В большинстве случаев в задачах инженерной оптимизации целевые функции и ограничения не заданы аналитически. Поэтому основные методы в составе пакета относятся к классу “черного ящика”. В состав пакета входит большой набор приближенных методов оптимизации. Также большое внимание уделено проблемам устойчивости найденного решения, оптимизации в условиях неопределенности. В пакете DAKOTA предусмотрена возможность параллель-

ного выполнения. Поддерживается концепция иерархического параллелизма, когда на верхнем уровне распараллеливаются итерации метода, а на нижнем процедуры вычисления целевой функции и ограничений [191].

Библиотека DAKOTA входит в состав программного комплекса Acro [192]. Другой универсальной расширяемой объектно-ориентированной библиотекой, входящей в состав Acro, является PICO [193]. В основе PICO лежит метод ветвей и границ. Пользователь наследует классы, отвечающие за операции метода ветвей и границ, применительно к конкретному методу. Общая каркасная схема метода ветвей и границ уже реализована в библиотеке и конкретный решатель получается объединением унаследованных классов и каркаса. Разработаны каркасы для последовательной и для параллельной реализаций. Параллельная реализация предназначена для выполнения на системах с распределенной памятью и поддерживает иерархическую балансировку нагрузки, в которой "мастер-процесс" управляет "процессами-хабами", каждый из которых в свою очередь управляет некоторым количеством подчиненных процессов. Есть возможность настройки параметров балансировщика нагрузки. На данный момент в состав PICO входят решатели для задачи о булевом ранце, коммивояжере, и частично-целочисленной задачи линейного программирования.

Система ALPS [194] построена во многом аналогично PICO. В ней также предусмотрен иерархический двухуровневый алгоритм балансировки нагрузки. На основе ALPS построено два других пакета оптимизации BiCePs и BLIS [195], предназначенные для решения задач линейного и частично-целочисленного линейного программирования соответственно.

Библиотека MALLBA [196] предоставляет расширяемую программную инфраструктуру для широкого класса методов глобальной оптимизации. Поддерживаются точные методы, эвристические и гибридные методы. Реализация основана на принципе каркасов (скелетонов), которые представляют собой шаб-

лонные реализации основных типовых схем оптимизационных алгоритмов. В частности, поддерживаются схемы для метода ветвей и границ и динамического программирования. Разработаны каркасные реализации этих схем для систем с распределенной и общей памятью, а также для распределенных систем, объединяющих несколько географически удаленных суперкомпьютеров. С использование MALLBA были реализованы несколько решателей для оптимизационных задач различного типа, начиная

Можно перечислить еще несколько систем PEBBL [197], APPSPACK [198], BOBPP [199], предназначенных для решения задач оптимизации на многопроцессорных вычислительны системах с общей и распределенной памятью.

В связи с ростом популярности распределенных систем, объединяющих несколько суперкомпьютеров в единый вычислительный комплекс. В работе [200] рассматривается инструмент FATCOP, направленный на решение задач частично-целочисленного программирования с помощью ветвей и границ на сети рабочих станций. FATCOP построен на основе коммуникационной библиотеки Condor-PVM. Эта библиотека является частью системы Condor [201], предназначенной для распределенных вычислений на сетях персональных компьютеров и рабочих станций. Два наиболее примечательных особенностей Condor являются способность обнаруживать неработающие рабочие станции и обеспечить миграцию задач между хостами. FATCOP использует стратегию управляющий-рабочие и простой централизованный алгоритм балансировки нагрузки. Для того чтобы справиться с отказами рабочих процессов, основной процесс сохраняет задания и в случае отказа переназначает задание другому процессу.

Другой оптимизационный пакет MW на основе Condor, также использующий парадигму управляющий-рабочие, предложен в [202]. Эта структура была успешно применена в различных оптимизационных задачах [203]. Пакет MW использует парадигму управляющий-рабочие.

Решение для распределенных систем, состоящих из нескольких кластеров , соединенных через глобальные сети (WAN) предлагается в работе [204]. Метод ветвей и границ реализован с использованием промежуточного программного обеспечения Ninf-G [205], обеспечивающего надежное соединение через глобальные и локальные сети. Система эффективно использует иерархическую природу распределенных систем, распределение работы выполняется на двух уровнях. На верхнем уровне работа присваивается главным процессом между управляющими процессами, которые уже распределяют работу между рабочими процессами.

Большинство из рассмотренных систем следуют принципу разделения проблемно-зависимой и проблемно-независимой функциональности. При таком подходе реализация солвера для нового класса оптимизационных задач заключается в разработке проблемно-зависимых модулей для данной задачи, общая схема (последовательность шагов метода) реализована в библиотеке. Таким образом происходит экономия усилий разработчиков при реализации новых задач и методов в рамках общей схемы метода ветвей и границ, поддерживаемой библиотекой.

В рамках работы над диссертацией был реализован программный комплекс BNB-Solver для реализации методов ветвей и границ, некоторых классов эвристических алгоритмов и гибридных методов. В рамках программного комплекса поддерживаются как платформы с общей, так и с распределенной памятью. От перечисленных программных инструментов BNB-Solver отличается в основном двумя чертами. Во-первых, поддерживается очень широкий класс задач оптимизации от булевого программирования до задач многокритериальной оптимизации с нелинейными целевыми функциями и ограничениями. Во-вторых, управление процессом вычислений в последовательном и параллельном вариантах выделена в отдельный модуль и для описания процесса управления раз-

работан специальный протокол. Такой подход позволяет реализовывать разные схемы балансировки нагрузки между процессами параллельной программы, не меняя другие классы библиотеки, что очень удобно, если пользователь захочет изменить алгоритм балансировки, не затрагивая инфраструктуру библиотеки.

6.2 Общие сведения об архитектурах современных параллельных и распределенных систем

6.2.1 Последовательные архитектуры

Классическая последовательная архитектура, предполагающая один поток команд и данных, в настоящее время не встречается даже на персональных компьютерах. Современные микропроцессоры, как правило, имеют несколько вычислительных ядер. При этом, каждое ядро можно рассматривать как вычислитель с последовательной архитектурой. При планировании процессов и потоков операционная система рассматривает вычислительные ядра как независимые процессоры, не делая принципиальных различий между многоядерными и классическими архитектурами с общей памятью. Несмотря на то, что в чистом виде последовательные архитектуры микропроцессоров практически не встречаются, существенная доля приложений использует именно модель последовательного выполнения. При этом основным средством разработки программ для решения вычислительных задач являются традиционные языки программирования C, C++, FORTRAN.

6.2.2 Многопроцессорные системы с общей памятью

В системах с общей памятью предполагается наличие нескольких вычислительных ядер, которые имеют доступ к единому пространству памяти. Обмен данными между процессорами производится следующим образом: один процессор записывает данные по некоторому адресу, а другой считывает их. К

этому классу относятся многопроцессорные рабочие станции и сервера, суперкомпьютеры с общей памятью (например HP Superdome), а также, получившие широкое распространение в последнее время многоядерные процессоры. При доступе к общей памяти необходима синхронизация для обеспечения эксклюзивного доступа к данным, которая реализуется с помощью таких механизмов, как мьютексы, семафоры, условные переменные. Основным инструментарием разработчика программ для архитектур этого типа являются библиотека для организации многопоточных вычислений POSIX Threads, пакет OpenMP, новые расширения Си++ для многопоточного программирования, введенные в последнем стандарте.

6.2.3 Многопроцессорные системы с распределенной памятью

Системы с общей памятью удобны с точки зрения разработки параллельных программ и обеспечивают высокую производительность, но при большом числе процессоров очень дороги. Менее затратной альтернативой являются системы с распределенной памятью. В таких системах каждый процессор имеет доступ только к своей локальной памяти, а между собой процессоры взаимодействуют с помощью передачи сообщений по сети. К этому классу относятся высокопроизводительные вычислительные кластеры, а также, локальные сети. Не редки также ЭВМ с гибридной архитектурой, в которой узлы с общей памятью соединяются посредством сетевого оборудования. Основной парадигмой программирования для систем с распределенной памятью является передача сообщений. Наиболее распространенным средством разработки приложений в таких системах является MPI (Message Passing Interface). Библиотека MPI предоставляет набор функций для передачи сообщений между процессорами.

6.3 Реализация метода ветвей и границ для многопроцессорных систем с распределенной памятью

6.3.1 Общие принципы реализации

Метод ветвей и границ применяется для решения различных задач. Общая схема метода хорошо известна и состоит из следующих шагов:

1. инициализировать список подзадач $\mathbb{P} = \{P_0\}$, записав туда исходную задачу P_0;
2. выбрать подзадачу Q из списка $\mathbb{P}$;
3. произвести действия, направленные на вычисление верхних и нижних оценок и декомпозицию задачи Q на подзадачи $\{Q_1, \ldots, Q_k\}$;
4. добавить полученные подзадачи $\{Q_1, \ldots, Q_k\}$ к списку $\mathbb{P}$;
5. проверить условие остановки и, если оно выполнено, то заврешить алгоритм, в противном случае перейти к шагу 2.

Сделаем несколько замечаний в отношении данной схемы. Действия, производимые на шаге 3, зависят от решаемой задачи и применяемого варианта МВГ. Нередко для вычисления оценок применяются различные эвристические алгоритмы, в результате чего обновляется так называемый *рекорд* — наилучшее найденное на данный момент значение целевой функции. Рекорд выполняет роль верней оценки (в случае задачи на минимум). При решении задач многокритериальной оптимизации рекордом является не одна точка, а совокупность точек, которая по окончании выполнения алгоритма станет множеством ε-Парето. Допускается наличие нескольких алгоритмов улучшения рекорда, которые можно переключать до начала либо по ходу вычислений. Алгоритмы улучшения рекорда могут отличаться как по трудоемкости, так и по эффективности нахождения рекорда. Поэтому, как-правило, присутствует альтернатива — использо-

вать более сильный и трудоемкий алгоритм или обойтись простым, но менее эффективным. Вычисление нижних оценок и применение правил отсева также может выполняться различными способами. Также как и в случае верхних оценок, возникает выбор между эффективностью и трудоемкостью вычисления оценки.

Также варьироваться может способ выбора подзадачи для декомпозиции (разбиения, ветвления). Выделяют следующие стандартные способы выбора подзадачи:

1. *ветвление в глубину* — для разбиения выбирается одна из подзадач, максимально удаленная от корня дерева ветвлений;
2. *ветвление в ширину* — для разбиения выбирается одна из подзадач, минимально удаленная от корня дерева ветвлений;
3. *ветвление по наилучшей оценке* — для разбиения выбирается одна из подзадач, имеющая минимальную нижнюю оценку (в случае задач скалярной минимизации).

Возможны также и другие варианты выбора подзадачи для ветвления, специфичные для конкретного алгоритма и решаемой задачи. Декомпозиция подзадачи также может осуществляться различными способами. Например, при решении задачи о булевом ранце для ветвления могут выбираться различные переменные, а в методе неравномерных покрытий могут использоваться различные варианты декомпозиции параллелепипеда.

Таким образом, алгоритм решения задачи оптимизации методом ветвей и границ управляется набором параметров. В 1990-м году И.Х. Сигалом было предложено [206] понятие *стратегии решения задачи* как способа управления перечисленными параметрами в процессе решения задачи с учетом ресурсов ЭВМ. И.Х. Сигалом были предложены различные стратегии для задач типа

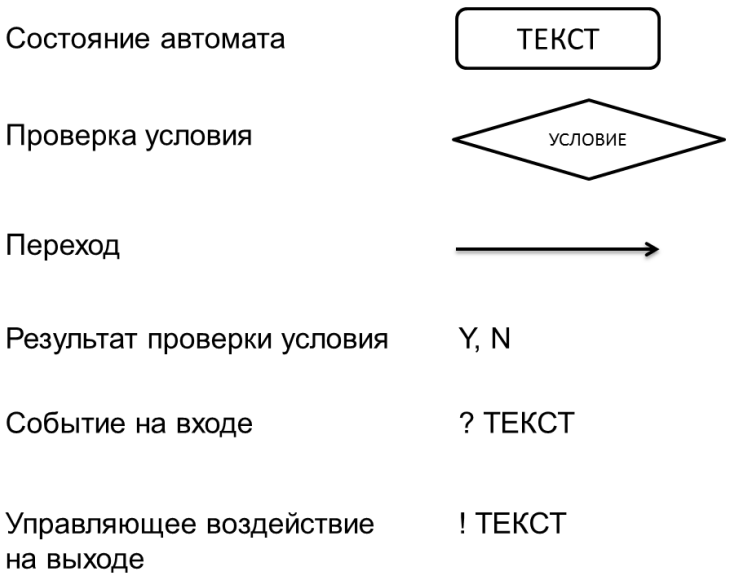

Рис. 6.1: Элементы формального описания стратегии решения задачи

коммивояжера и задач ранцевого типа.

Реализация на параллельных архитектурах подразумевает распределение вычислений между вычислительными ядрами (процессорами). В наиболее простом варианте работа распределяется статически до начала вычислений. При параллельной реализации метода ветвей и границ используются методы динамической балансировки нагрузки, при которой происходит передача подзадач между процессами параллельной программы в процессе вычислений с целью выравнивания загрузки процессоров и предотвращения простоя вычислительных мощностей. Стратегия решения оптимизационной задачи в параллельном случае заключается в управлении не только стандартным набором параметров, характерным для метода ветвей и границ, но и балансировкой нагрузки.

6.3.2 Формализация описания управления процессом вычислений

Процесс решения оптимизационной задачи комбинированным методом ветвей и границ можно представить как взаимодействие *модуля управления*, реализующего стратегию решения, и *вычислительного модуля*, выполняющего шаги метода. Для наглядного и строгого описания логики управления процессом ре-

шения в диссертационной работе было предложено использовать расширенный формализм конечных автоматов с двумя лентами. Основные элементы, входящие в формальное описание такого автомата, приведены на Рис. 6.1. Автомат выполняет переходы между состояниями, реагируя на события, приходящие от вычислительного модуля и генерирует управляющие воздействия (команды), которые воспринимаются и выполняются вычислительным модулем.

Переход автомата, обозначаемый стрелкой, имеет два атрибута: условие, по которому осуществляется переход и генерируемая в результате перехода команда. Любой из этих атрибутов может быть опущен. Если переходу не сопоставлено условие, то оно считается тождественно истинным и переход осуществляется всегда независимо от событий, генерируемых системой. Аналогично, если переходу не сопоставлена команда, то выполнение перехода не приводит к генерации команды.

Условие перехода может быть двух типов: результат операции сравнения или событие, пришедшее от вычислительного модуля. В первом случае переход помечается соответствующей буквой (Y, N), которые обозначают положительный или отрицательный результат сравнения соответственно. Во втором случае переходу приписывается метка “? ТЕКСТ”, где ТЕКСТ содержит название события, по реакции на которое выполняется переход. Генерируемая команда обозначается как приписанная переходу метка вида “! ТЕКСТ”, где ТЕКСТ обозначает генерируемую команду.

Команды как и события могут иметь аргументы, которые указываются в круглых скобках через запятую. В случае команд аргументы представляют собой константы, либо переменные, значения которых передаются в команду в качестве аргумента. Аргументы событий аргументами являются переменные, которые заполняются решателем.

Основные события и их аргументы перечислены таблице 6.3.2. Перечислен-

Таблица 6.1: Основные события

Название	Аргументы	Описание
ERROR	код ошибки	возникновение ошибки
START		событие, соответствующее началу расчетов
DONE	количество выполненных шагов	выполнено некоторое число шагов метода
SENT	число элементов переданных данных	запрос на посылку данных или команды выполнен
DATA_ARRIVED	процесс, приславший данные	приняты данные
COMMAND_ARRIVED	процесс, приславший команду, команда и ее аргументы	принята управляющая команда
SET_SUCCESS		установка значения запрошенного параметра произведена успешно
CUSTOM_EVENT	произвольные аргументы определяемые расширением	дополнительное событие для создания расширений

ные события являются универсальными и применимы к любому варианту метода ветвей и границ. События, специфичные для конкретного варианта задаются с помощью специального события CUSTOM_EVENT, допускающего произвольное количество аргументов, не превосходящее заданной в системе константы [1].

Основные управляющие воздействия (команды) представлены в таблице 6.3.2. Часть команд предназначены для установки параметров поиска, часть предназначена для организации обмена данными между процессами параллельной программы. В частности, команда SEND_SUBS получает в качестве аргумента число подзадач для передачи. Передается заданное число подзадач или меньшее. Специальная команда CUSTOM_COMMAND дает возможность передавать дополнительные команды, специфичные для конкретной реализации.

Рассмотрим несколько примеров автоматов, описывающих стратегию решения. На Рис. 6.2 приведен пример автомата, описывающего стратегию управления методом ветвей и границ в последовательном случае с переключением между ветвлением в ширину и в глубину. Начальное состояние автомата обо-

[1] На данным момент число аргументов событий и констант не может быть более 8

Таблица 6.2: Основные команды

Название	Аргументы	Описание
SOLVE	число шагов	выполнить определенное число шагов метода
EXIT		завершить выполнение алгоритма
SET_STRATEGY	номер стратегии	установка стратегии поиска
SET_HEURISTIC	номер эвристики	установка эвристики для нахождения рекорда
SET_BOUNDING_METHOD	номер метода получения оценки	установка метода получения оценки
SET_CUSTOM_PARAMETER	номер параметра и его значение	установка значения некоторого параметра
SEND_COMMAND	процесс-получатель, номер команды и ее аргументы	посылка управляющей команды другому процессу
SEND_RECORDS	процесс-получатель	переслать найденные рекордные значения
SEND_SUBS	процесс-получатель, количество подзадач для передачи другому процессу	переслать подзадачи
SEND_SUBS_AND_RECORDS	процесс-получатель, количество подзадач для передачи	переслать подзадачи и рекорды другому процессу
RECV	процесс-отправитель или -1 (прием от всех)	ожидание прибытия данных или команды от другого процесса
CUSTOM_COMMAND	произвольные аргументы определяемая расширением	дополнительная команда для создания расширений

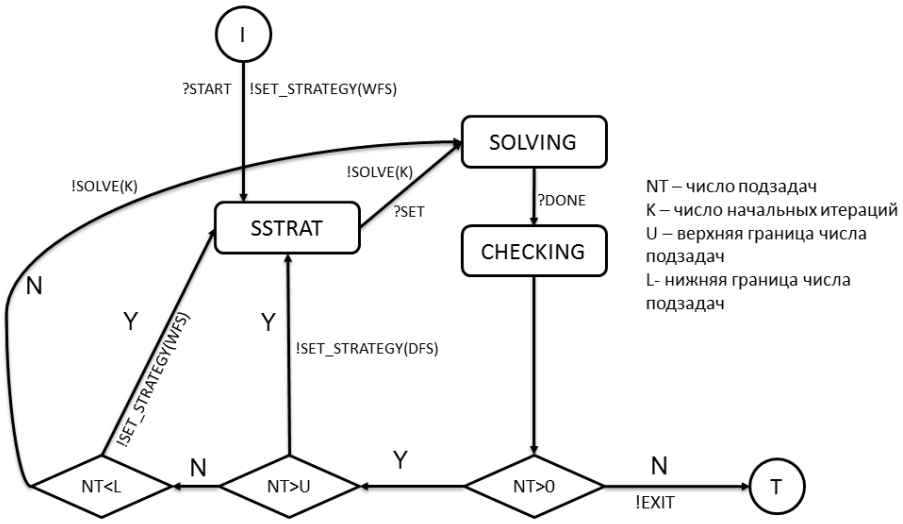

Рис. 6.2: Пример автомата, описывающего стратегию управления процессом оптимизации

значено буквой I, а конечное — буквой T. Способ выбора очередного множества для ветвления сначала соответствует ветвлению в ширину с помощью команды SET_STRATEGY(WFS) и затем меняется в динамике. Если число подзадач в списке превосходит заданную верхнюю границу U, то выдается команда SET_STRATEGY(DFS), переключающая вычислительный модуль на стратегию ветвления в глубину. Если число подзадач в списке становится меньше установленной нижней границы L, то происходит переключение на стратегию ветвления в ширину с помощью команды SET_STRATEGY(WFS). Проверка числа подзадач в списке производится каждый K итераций в состоянии CHECKING.

Перейдем к рассмотрению формализации одного из базовых вариантов реализации метода ветвей и границ. Из множества процессов выделяется один процесс — управляющий. Все остальные являются рабочими. Управляющий процесс распределяет подзадачи между рабочими процессами. Логика работы управляющего процесса задается конечным автоматом, представленным на Рис. 6.3.

На начальном этапе управляющий процесс устанавливает стратегию ветвления “в ширину” и выполняет K итераций метода. Выбор стратегии ветвле-

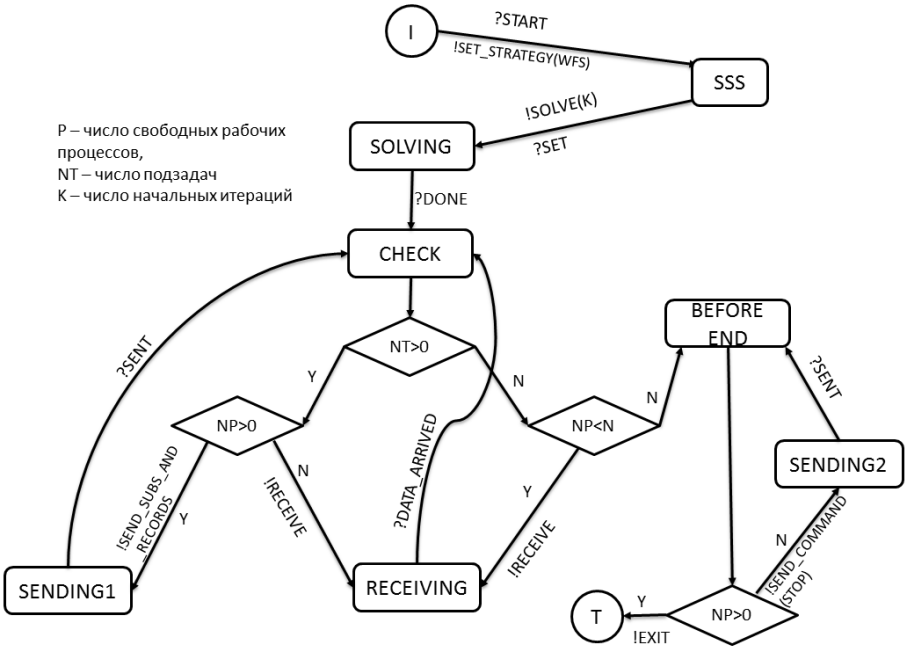

Рис. 6.3: Автомат, определяющий логику работы управляющего процесса

ния “в ширину” дает возможность сгенерировать большее число подзадач, сделав заданное число шагов, чем стратегия “в глубину”. Поэтому эта стратегия используется на управляющем процессе. Затем начинается процесс рассылки сгенерированных подзадач, число которых задается переменной NT, рабочим процессам. Если не все подзадачи разосланы (NT $>$ 0), то производится их рассылка по свободным (не занятых решением задач) рабочим процессам, количество которых задается переменной NP (левая петля в графе автомата). Если же все подзадачи разосланы, то освободившиеся процессы останавливаются с помощью команды STOP (правая петля в графе автомата), после чего управляющий процесс завершает работу.

Логика выполнения рабочего процесса определяется автоматом, представленным на Рис. 6.4. После установки стратегии ветвления “в глубину” рабочий процесс, получив одну или несколько подзадач от управляющего процесса, решает их полностью. Затем, выполняется переход по витку цикла и ожидается следующая задача или управляющая команда. В данном контексте единственная возможная управляющая команда — это STOP, по ее получении выполняется

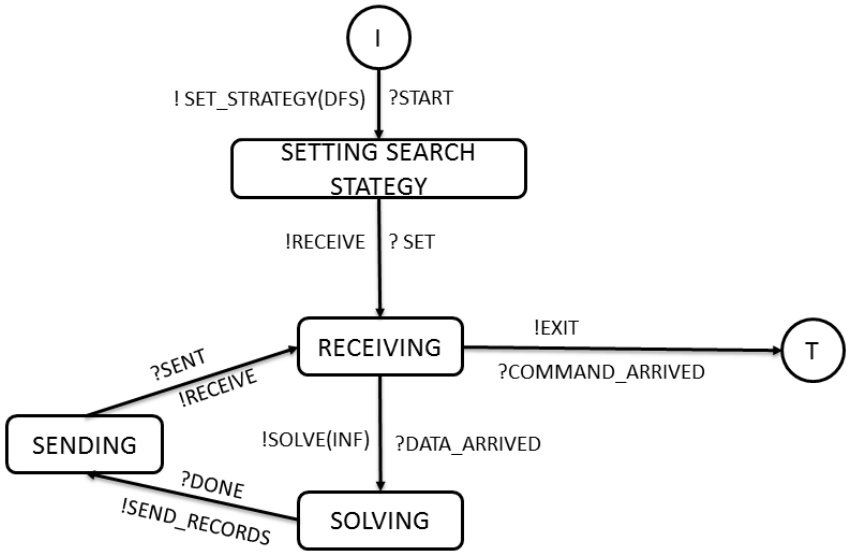

Рис. 6.4: Автомат, определяющий логику выполнения рабочего процесса

завершение рабочего процесса.

6.3.3 Основные компоненты библиотеки BNB-Solver и их взаимосвязь

Библиотека BNB-Solver реализована на языке программирования C++. Язык C++ обладает мощными средствами поддержки объектно-ориентированного программирования, обеспечивающим высокий уровень абстракции, модульности и гибкости программирования. С другой стороны C++ унаследовал от языка C низкоуровневые средства, дающие возможность эффективного использования аппаратных ресурсов. Наряду с Фортраном, C++ является основным средством разработки программ для параллельных систем.

BNB-Solver включает в себя следующие подсистемы и модули:

- модули управления вычислительным процессом;
- проблемно-зависимые компоненты (солверы);
- вспомогательные модули базовых вычислительных и коммуникационных процедур;
- модули интеграции балансировщиков и солверов для последовательных и

параллельных вариантов.

Рассмотрим подробнее эти компоненты системы. Классы управления вычислительным процессом наследуют базовый класс `BNBScheduler`, реализуя при этом абстрактный метод этого класса `virtual void action(const Event& event, const SolverInfo& info, Action& action)`. Эта функция принимает на вход описание событие и информацию о состоянии процесса решения. Результатом работы программы является команда `action`. Таким образом реализуется конечный автомат с двумя лентами.

Модули управления вычислительным процессом поддерживают последовательный сценарий, рассмотренный в предыдущем пункте и две схемы управления параллельными вычислениями. Одна из них рассматривался в предыдущем разделе. Приведем описание другой, более сложной схемы балансировки.

Рабочий процесс управляется двумя параметрами Ц T и S, посылаемыми ему управляющим процессом. Получив подмножество для обработки, рабочий процесс выполняет T разбиений, после чего пересылает S сгенерированных подмножеств на управляющий процесс. По окончании разбиения своего подмножества процесс рабочий процесс получает от управляющего новое подмножество.

Для предотвращения переполнения памяти на управляющем процессе введено два пороговых параметра m и M. Если число подмножеств на управляющем процессе становится больше M, то управляющий процесс рассылает всем рабочим процесса новое значения T, равное -1, и рабочие процессы перестают пересылать подмножества управляющему процессу. Когда число подмножеств на управляющем процессе становится меньшим m, он рассылает рабочим процессам первоначальные значения T, S и передача подмножеств с рабочих процессов управляющему возобновляется. Этот алгоритм балансировки нагрузки хорошо зарекомендовал при решении различных задач глобальной оптимизации с одним и несколькими критериями.

Проблемно-зависимые модули реализуют конкретный вариант метода ветвей и границ. В библиотеке реализованы варианты МВГ для задачи о сумме подмножеств, задачи о ранце, различные варианты метода неравномерных покрытий, описанные в диссертационной работе для задач с одним и несколькими критериями. Классы, реализующие проблемно-зависимые модули, наследуют интерфейс `BNBResolver`, представленный на следующем листинге (оставлены только наиболее важные методы):

```
class BNBResolver {
public:
    /**
     * Скопировать в сериализатор подзадачи в указанном количестве
     */
    virtual void getSubs(int& num, BinarySerializer& ser){
        BNB_ERROR_REPORT("BNBResolver::getSubs
                    method is not implemented");
    }
    /**
     * Скопировать в сериализатор информацию о найденных рекордах
     */
    virtual void getRecords(BinarySerializer& ser) {
        BNB_ERROR_REPORT("BNBResolver::getRecords
                    method is not implemented");
    }

    /**
     * Скопировать из сериализоватора подзадачи
     */
```

```
virtual int putSubs(BinarySerializer& ser) {
    BNB_ERROR_REPORT("BNBResolver::putSubs
                method is not implemented");
}

/**
 * Скопировать рекорды из сериализатора
 */
virtual void putRecords(BinarySerializer& ser) {
    BNB_ERROR_REPORT("BNBResolver::putRecords
                method is not implemented");
}

/**
 * Установить стратегию поиска с заданным номером
 */
virtual void setSearchStrategy(int strategy) {
    BNB_ERROR_REPORT("BNBResolver::setSearchStrategy
                method is not implemented");
}

/**
 * Установить эвристику для улучшения рекорда
 */
virtual void setHeuristic(int heuristic) {
    BNB_ERROR_REPORT("BNBResolver::setHeuristic
                method is not implemented");
```

```
    }

    /**
     * Установить метод получения оценок с заданным номером
     */
    virtual void setBoundingMethod(int method) {
        BNB_ERROR_REPORT("BNBResolver::setBoundingMethod
                    method is not implemented");
    }

    /**
     * Выполнить заданное количество шагов алгоритма. Реально
     * выполненное число шагов возвращается через параметр steps.
     * Это значение может быть меньше, чем запрошенное.
     */
    virtual void solve(long long& steps)    {
        BNB_ERROR_REPORT("BNBResolver::solve
                    method is not implemented");
    }
};
```

Методы, `getSubs`, `putSubs`, `getRecords`, `putRecords` предназначены для обмена подзадачами между процессами параллельной программы. Одним из аргументов этих методов является так называемый сериализатор, который выполняет роль контейнера для пересылки данных. Задачей солвера является обеспечение возможности сохранение и извлечение информации из этого контейнера. Таким образом достигается разделение между проблемно-зависимой частью библиотеки и коммуникационной частью.

Методы `setSearchStrategy, setHeuristic, setBoundingMethod` позволяют устанавливать до начала или переключать в процессе расчетов стратегию ветвления, эвристический алгоритм для улучшения рекорда и способ вычисления оценки. Предполагается, что эти алгоритмы в солвере пронумерованы в порядке возрастания сложности и эффективности, т.е. большим номерам соответствуют алгоритмы с более высокой трудоемкостью и большей эффективностью. Метод `solve` выполняет заданное количество шагов (итераций) метода, используя установленные параметры. Если задача решена за меньшее число шагов, то это число возвращается через параметр `steps`.

Все перечисленные методы являются виртуальными методами класса BNBResolver. Но при этом реализация по умолчанию выдает диагностическое сообщение с помощью макроса `BNB_ERROR_REPORT`. Если солвер будет использоваться только в последовательном контексте, то нет необходимости реализовывать методы для работой с сериализатором. При этом программа будет корректно функционировать, т.к. эти методы не вызываются. При попытке использовать класс в параллельной программе выполнение будет прервано с соответствующей диагностикой.

Вспомогательные модули базовых вычислительных алгоритмов и коммуникационных процедур содержат основные вычислительные алгоритмы и коммуникационные примитивы. К базовым алгоритмам относятся различные функции для работы с векторами, процедуры одномерного и локального поиска экстремума функции. Отдельный модуль предназначен для вычисления интервальных оценок основных алгебраических функций. Также предусмотрены функции для вычисления полиномов и их производных по символьному представлению. Коммуникационные процедуры реализуются в классе, который наследует класс `Communicator`, задающий абстрактный интерфейс для коммуникационных обменов.

```
class Communicator {
public:

    /**
     * Возвращает номер процесса, вызывающего метод
     */
    virtual int rank() const = 0;

    /**
     * Возвращает число процессов в коммуникаторе
     */
    virtual int size() const = 0;

    /**
     * Осуществляет пересылку буфера процессу с заданным номером
     */
    virtual int send(const Buffer& buf, int dest) = 0;

    /**
     * Выполняет проверку наличия пришедшего сообщения от процесса с
     * заданным номером или от любого процесса, если указано -1
     */
    virtual bool probe(int source = -1) = 0;

    /**
     * Выполняет прием сообщения и сохранение
 * в буфер полученных данных
```

```
     * от процесса с заданным номером или от любого процесса,
     * если указано -1
     */
    virtual int recv(Buffer& buf, int source = -1) = 0;

    /**
     * Рассылает всем процессам содержимое буфера
     */
    virtual int bcast(Buffer& buf) = 0;

    /**
     * Возвращает число полученных байтов за все время
     * функционирования
     */
    virtual long long int getRecvBytes() const = 0;

    /**
     * Возвращает число посланных байтов за все время
     * функционирования
     */
    virtual long long int getSentBytes() const = 0;

};
```

Наличие универсального интерфейса коммуникатора позволяет скрыть детали реализации передачи данных от других частей программы, использующих коммуникационные примитивы. На данный момент в библиотеке имеется только одна реализация данного интерфейса — класс `MpiCommunicator`, который

предназначен для выполнения в среде MPI (Message Passing Interface).

В библиотеке реализовано два класса для интеграции солверов и модулей управления. Первый `BNBSeqSolver` предназначен для объединения модулей управлением последовательной реализацией и солверов. Второй класс `BNBDmSolver` может интегрировать три компонента: солвер, коммуникатор и модуль управления параллельными вычислениями. Модуль управления генерирует команды, которые `BNBDmSolver` обрабатывает, вызывая соответствующие методы коммуникатора либо солвера. Типовой код программы, выполняющей решение задачи параллельным алгоритмом, выглядит следующим образом:

```
 /**
  * Создание коммункатора
*/
 MpiCommunicator com;
/**
 * Создание экземпляра класса солвера
 * (в данном случае решается задача
 * многокритериальной
 * оптимизации методом неравномерных покрытий)
 */
 MultiResolver mr(&ms, &ml);
/**
 * Создание экземпляра класса, управляющего параллельным выполнением
 */
 SimpSched sched(size * 8);
/**
 * Создание параллельного решателя с
 * помощью интеграции коммуникатора, планировщика и солвера
```

```
*/
BNBDmSolver sol(&com, &sched, &mr);
```

6.4 Способы повышения эффективности параллельной реализации метода ветвей и границ на примере задачи о ранце

6.4.1 Использование специфики решаемой задачи для эффективной реализации ветвлений

Реализация метода ветвей и границ для конкретной задачи может представлять определенные сложности на практике, т.к. эффективная реализация метода плохо распараллеливается из-за наличия зависимости по данным между итерациями вычислительного процесса. В данном разделе рассматривается вопрос эффективной реализации метода ветвей и границ для задачи об одномерном булевом ранце. Рассматривается комбинирование двух способов ветвления: по наибольшей оценке и одностороннего. Одностороннее ветвление более эффективно с точки зрения экономии памяти и организации вычислений. В частности, вариант одностороннего ветвления, предложенный Горовицем и Сахни [158], для задачи о булевом ранце с одним ограничением позволяет организовать перебор на базе одного бинарного вектора. Такой подход исключительно эффективен на однопроцессорной машине. На параллельной системе при его применении возникают сложности, связанные с необходимостью пересылки подзадач между процессорами. Предлагается модификация этого алгоритма для применения на параллельной вычислительной системе, достигаемая комбинированием с организацией МВГ на базе пула подзадач. Под пулом в данном контексте понимается структура данных, предназначенная для хранения множества подзадач. С помощью экспериментов показывается, что применение такого подхода позволяет эффективно распараллелить метод Горовица-Сахни,

сохранив при этом присущую ему высокую скорость выполнения операций ветвления.

Дальнейшее ускорение решения задач дискретной оптимизации может быть получено с применением эвристических алгоритмов. Под эвристическими понимаются алгоритмы, основанные на правдоподобных, но не обоснованных строго предположениях о свойствах оптимального решения задач. Предлагается подход, основанный на комбинировании параллельного МВГ и параллельного эвристического поиска в одном приложении. В такой схеме эвристические алгоритмы обрабатывают допустимые решения, получаемые по ходу работы МВГ, а МВГ использует для отсева рекордные значения, формируемые эвристиками. В работе приводятся результаты экспериментов, на основании которых анализируется эффективность параллельной реализации комбинированных алгоритмов, и исследуются правила выбора параметров, управляющих работой алгоритмов.

На каждом шаге метода ветвей и границ задача декомпозируется на две подзадачи путем придания некоторой переменной значения 0 в одной из них и 1 — в другой. Естественным образом строится бинарное дерево подзадач, в котором дуги направлены от исходной к задачам, полученным при ветвлении (декомпозиции). Концевые вершины могут быть исключены по правилам отсева. Одна из концевых неотсеянных подзадач подвергается ветвлению и заменяется подзадачами, возникшими при ветвлении. Процесс завершается, когда все концевые вершины отсеяны, т.е. дальнейшее ветвление невозможно.

Известный алгоритм Горовица-Сахни является существенно более эффективным по сравнению с общей схемой, рассмотренной в предыдущем разделе. Этот метод реализует односторонний обход дерева поиска, причем первой всегда обходится вершина, соответствующая присваиванию значения 1 переменной для ветвления. Основным достоинством алгоритма Горовица-Сахни служит эконом-

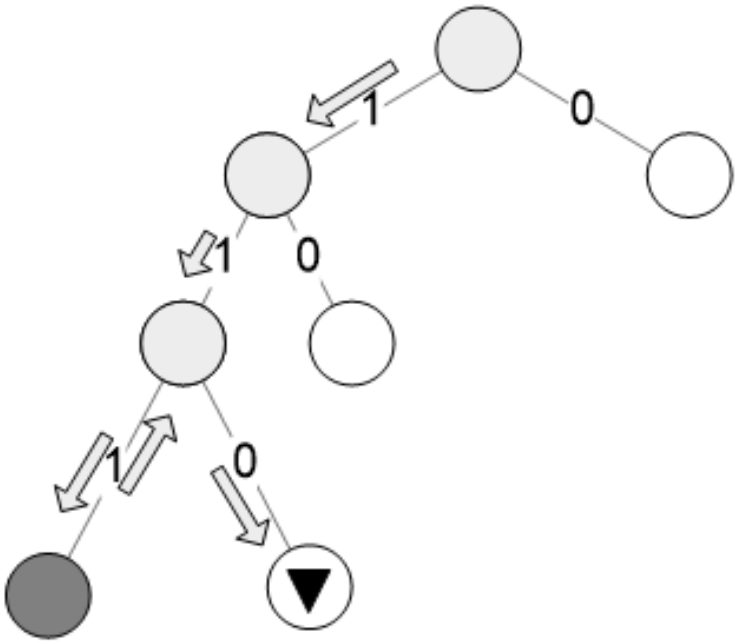

Рис. 6.5: Односторонний обход в алгоритме Горовица-Сахни, текущий вектор (110)

ное использование памяти и отсутствие лишних операций копирования. Это достигается за счет того, что для хранения дерева используется бинарный массив длины n. На рис. 6.5 изображена ситуация частичного обхода дерева. Белые вершины еще не обходились, светло-серые уже подвергались декомпозиции, а темные уже пройдены и не подлежат дальнейшему ветвлению. Треугольный маркер обозначает текущую вершину в дереве. Текущий вектор составлен из нулей и единиц, выписанных вдоль пути от корня дерева к текущей вершине, таким образом, он однозначно задает положение текущей вершины.

Алгоритм последовательно обходит дерево ветвления в одностороннем порядке, используя для этого вектор. На каждом шаге очередной элемент вектора получает значение 1, если при этом не нарушается граница по весу и 0 — в противном случае. Если незаполненные элементы вектора положить равными 0, то получится некоторое допустимое решение x. При условии $f(x) > f_0$, где f_0 — рекорд, $f(x)$ становится новым рекордом, а x — новым рекордным решением. После каждого шага производится вычисление оценки с помощью решения

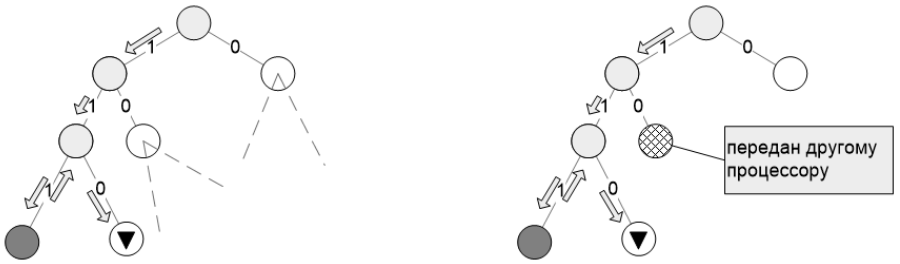

Рис. 6.6: Кодирование события передачи поддерева другому процессу в текущем векторе: а — стандартный обход, б — ситуация, когда подзадача передана для обработки другому процессору.

задачи линейной релаксации. Если оценка не превосходит рекорд, то выполняется обратный ход алгоритма. На этом этапе происходит возврат в обратном направлении по вектору до первой единицы, которая заменяется 0 и алгоритм продолжается со следующей позиции. В такой реализации вектор применяется для хранения информации об обходе дерева и содержит путь от корня дерева до текущей вершины. Особенностью алгоритма Горовица-Сахни является то, что пул подзадач в явном виде не формируется, что позволяет минимизировать требуемый размер памяти и сократить объем вычислений, выполняемых при одной операции декомпозиции. С другой стороны, отсутствие пула подзадач выступает препятствием для параллельной реализации. Действительно, в стандартной реализации подзадача для передачи выбирается из пула. В данном случае это действие невозможно выполнить непосредственно в связи с тем, что пул не используется.

Предлагаемое нами решение данной проблемы основано на модификации алгоритма Горовица-Сахни. Стандартный алгоритм Горовица-Сахни кодирует информацию об обходе в векторе следующим образом: 0 означает, что в соответствующем месте в дереве было выбрано правое ребро, а поддерево, сопоставленное левому ребру, уже пройдено на предыдущих шагах алгоритма. Если в

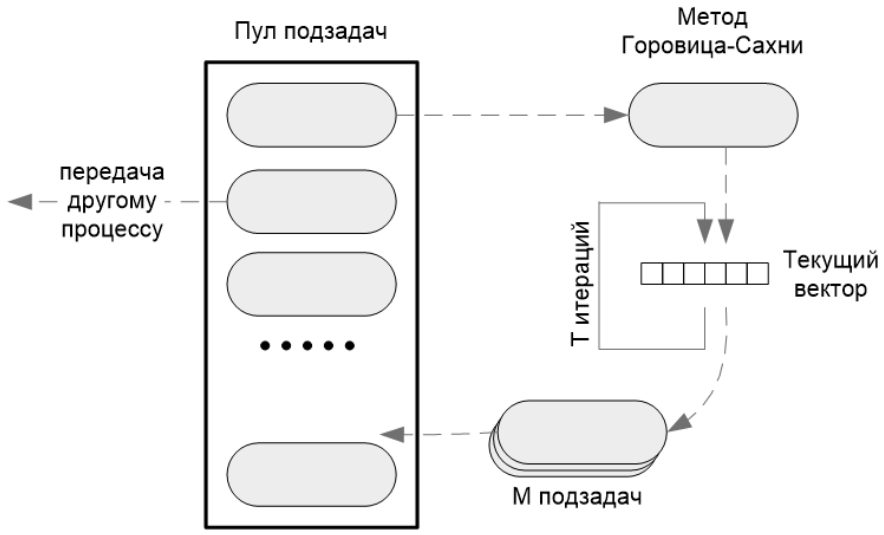

Рис. 6.7: Комбинированный алгоритм на основе схемы Горовица-Сахни

некоторой позиции записана 1, то было выбрано ребро, помеченное 1, а поддерево, соответствующее нулевому ребру, еще не было пройдено. Алгоритм работает так, что обходу не подвергались поддеревья, соединенные с текущим путем нулевыми ребрами. Именно такие поддеревья можно передавать для дальнейшей обработки другим процессам. При передаче необходимо изменить данные алгоритма, стремясь, чтобы переданное поддерево не обходилось процессом, осуществившим передачу. Для этого в соответствующую позицию текущего вектора записывается 2 (рис. 6.6).

При такой реализации значение 2 в некоторой позиции указывает на то, что при обходе было выбрано единичное ребро, а поддерево, соответствующее нулевому ребру, обрабатывается другим процессом. Базовый алгоритм Горовица-Сахни при этом подвергается минимальной модификации: при обратном ходе алгоритма значение 2 обрабатывается аналогично 0.

Рассмотренная модификация метода Горовица-Сахни позволяет построить следующий комбинированный алгоритм (рис. 6.7). На процессе поддерживается пул подзадач. На очередном шаге из него выбирается подзадача, которая решается методом Горовица-Сахни. При этом если алгоритм завершил свою работу

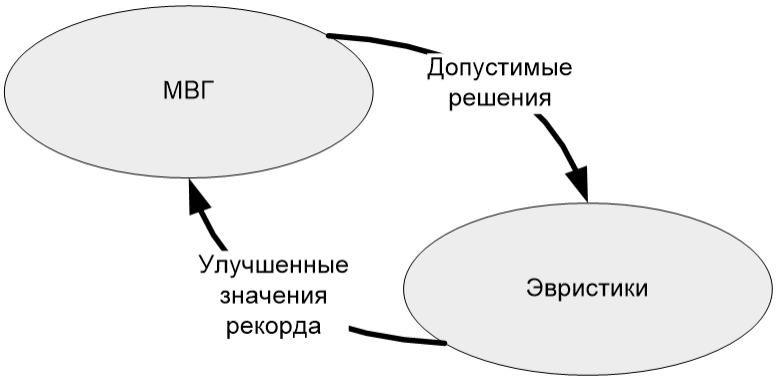

Рис. 6.8: Взаимодействие метода ветвей и границ и эвристических алгоритмов.

за число шагов, меньшее заданного параметра T, то из пула извлекается новая подзадача. В противном случае после T итераций алгоритм прерывается и из сформированного дерева ветвления выделяется не более M подзадач, при этом текущий вектор модифицируется способом, указанным в предыдущем разделе. Эти подзадачи добавляются в пул. Подзадачи, хранимые в пуле, могут быть использованы для передачи другим процессам параллельного приложения.

Правильный выбор параметров T и M является важнейшим условием обеспечения высокой производительности системы. На основании экспериментов было установлено, что наилучшие результаты получаются в ситуации, когда $T \geq n$, где n — размерность задачи, а значение параметра M изменяется динамически так, чтобы число подзадач в пуле не превосходило заданной величины Q. Другими словами, M полагается равным величине $Q - Q'$, где Q' — число задач в пуле в данный момент времени. Величину Q следует выбирать так, чтобы обеспечить необходимую загрузку процессов параллельного приложения.

6.4.2 Использование эвристик для ускорения поиска экстремума

Другим полезным вариантом комбинирования является объединение МВГ и эвристических алгоритмов. Эта технология известна и применяется давно при решении различных задач дискретной оптимизации [159]. В общем виде вза-

имодействие МВГ и эвристического алгоритма происходит в соответствии со следующей схемой (рис. 6.8): в процессе решения МВГ предоставляет эвристическому алгоритму допустимые решения. Эвристический алгоритм обрабатывает полученные решения и, если в результате удается улучшить рекорд, он передается для обработки МВГ.

Применение эвристических алгоритмов для ускорения работы МВГ в последовательном варианте требует значительных затрат вычислительных ресурсов, которые могут свести на нет достигнутый выигрыш в производительности и привести к замедлению работы алгоритма. На многопроцессорном вычислительном комплексе этого удается избежать, так как эвристические алгоритмы могут работать параллельно с МВГ. Кроме того, сама процедура поиска локального экстремума может быть реализована более эффективно с использованием методов параллельных вычислений.

Применение эвристических алгоритмов для ускорения работы МВГ в последовательном варианте требует значительных затрат вычислительных ресурсов, которые могут свести на нет достигнутый выигрыш в производительности и привести к замедлению работы алгоритма. На многопроцессорном вычислительном комплексе этого удается избежать, так как эвристические алгоритмы могут работать параллельно с МВГ. Кроме того, сама процедура поиска локального экстремума может быть реализована более эффективно с использованием методов параллельных вычислений.

С учетом перечисленных особенностей разработана следующая схема организации параллельного приложения, совмещающего эвристические алгоритмы и МВГ (рис. 6.9). На рисунке введены следующие обозначения:

- M — процесс, управляющий работой всего приложения;
- B_1, B_2, B_3 — процессы, которые выполняют операции ветвления в МВГ;

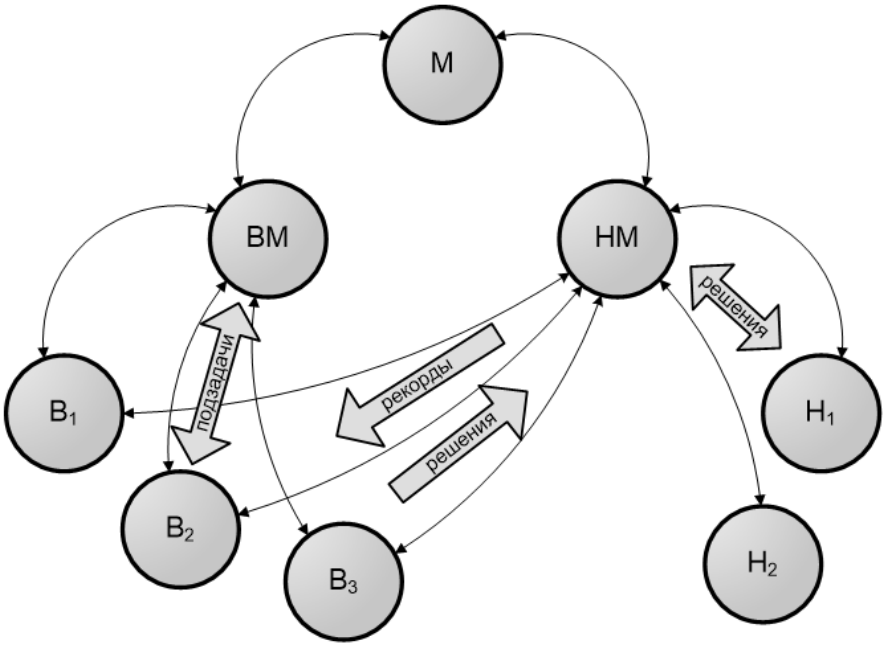

Рис. 6.9: Общая схема параллельной реализации совместной работы МВГ и эвристических алгоритмов

- BM — процесс, координирующий работу процессов B_1, B_2, B_3;
- H_1, H_2 — процессы, которые выполняют эвристический поиск;
- HM — процесс, координирующий процессы, выполняющие эвристический поиск.

В начале работы процесс M считывает исходные данные задачи из файла и рассылает их остальным процессам. МВГ выполняется группой, состоящей из управляющего процесса BM и рабочих процессов B_i. По ходу решения очередной задачи-кандидата на любом из процессов B_i может быть определено допустимое решение. Если принимаемое на нем значение целевой функции улучшает рекорд, то это решение пересылается управляющему процессу эвристического поиска HM. Процесс HM получает допустимые решения от процессов H_i и B_i, обновляет значение рекорда и рассылает его по всем процессам B_i, реализующим МВГ. Далее полученное решение подвергается процедуре локальной оптимизации. Для этого процесс HM направляет допустимое решение всем процессам H_i, выполняющим эвристический поиск. Приняв допустимое решение,

Таблица 6.3: Характеристики оборудования

Название	Процессоры	Сетевое оборудование
MVS-15000BM	1148 процессоров PowerPC 970 2.2 GHz	Myrinet
MVS-100K	1980 процессоров Intel Xeon E5450 (4 ядра) 3 GHz	Infiniband DDR

процесс H_i применяет к нему эвристический алгоритм. Если удается улучшить рекорд, то соответствующее допустимое решение посылается процессу HM. Предлагаемая схема может применяться в двух вариантах: когда различные процессы H_i выполняют различные эвристические алгоритмы, либо когда они совместно образуют один параллельный эвристический алгоритм.

Приведем определение окрестности для задачи о ранце, данное в [159]. Пусть $z = (z_1, \ldots, z_2)$ — допустимое решение. Зафиксируем s, $1 \leq s \leq n$. Окрестностью z назовем множество допустимых решений $G(z)$, отличающихся от z не более чем в s компонентах, расположенных подряд.

Пусть $z = (z_1, \ldots, z_n)$ — некоторое допустимое решение. Предлагаемый алгоритм находит оптимум в окрестности $G(z)$ данного решения. Для этого для каждого $i \in 1, \ldots, n$ решается задача оптимизации $\sum_{j=i}^{k} p_j x_j \to \max, \sum_{j=i}^{k} p_j x_j \leq C'$, где $k = \begin{cases} i+s, \ i+s \leq n, \\ i+s \mod n, i+s > n, \end{cases}$ $C' = C - \sum_{j=i}^{k} w_j$. Из найденных решений выбирается решение с наибольшим значением целевой функции. Если для локальной оптимизации выделены процессы $H_1, , H_k$. Процесс H_i просматривает наборы длины s, начинающиеся с позиций $i, i+k, i+2k, \ldots$. Таким образом, все процессоры анализируют приблизительно равное число различных наборов и работа распределяется равномерно.

Первая серия проведенных экспериментов была направлена на изучение эффективности применения комбинированного параллельного алгоритма на основе метода Горовица-Сахни. Следует отметить, что организация вычислений в соответствии с базовым алгоритмом Горовица-Сахни существенно эффективней подхода, основанного на копировании подзадач. Для сравнения рассмотрим

Таблица 6.4: Время работы (сек.) различных реализаций МВГ

Число переменных в примере	Время решения МВГ на основе пула подзадач	Время решения алгоритмом Горовица-Сахни
20	5.17	0.18
24	77.19	2.56
28	1184.02	37.1

Таблица 6.5: Время решения двух задач различными библиотеками

Число ядер		1	2	4	8	16	32
1	ALPS	16.7	19.2	6.57	3.12	2.55	1.85
	PEBBL	137.2	56.3	35.5	23.6	17.1	15.2
	BNB-Solver	1.32	0.69	0.34	0.18	0.11	0.07
2	ALPS	-	-	-	-	-	-
	PEBBL	743.3	369.5	183.1	99.9	50.2	35.1
	BNB-Solver	2.92	1.86	0.92	0.52	0.50	0.42

вариант задачи о ранце, предложенный Ю.Ю. Финкельштейном [24]:

$$\sum_{i=1}^{n} 2x_i \to \max, \sum_{i=1}^{n} 2x_i \leq 2\lfloor \frac{n}{2} \rfloor + 1, x_i \in \{0,1\}. \tag{6.1}$$

Особенностью этого примера является независимость числа ветвлений от стратегии ветвления. Поэтому его удобно использовать в качестве теста для сравнения эффективности различных вариантов реализации МВГ. Эксперимент проводился на одном из узлов многопроцессорного вычислительного комплекса МВС-15000 БМ, установленном в Межведомственном суперкомпьютерном центре РАН [24]. Характеристики суперкомпьютера приведены в табл. 6.3.

В табл. 6.4 содержатся результаты измерения времени работы для различного числа переменных в рассматриваемом примере, полученные при решении с помощью последовательного алгоритма МВГ на основе пула подзадач и последовательного алгоритма Горовица-Сахни на одном процессоре. Такой результат является ожидаемым, если учесть, что на каждом шаге алгоритма не производится копирования подзадач.

Оценку эффективности параллельной реализации алгоритма на основе схемы Горовица-Сахни проведем на двух примерах. Первый — это задача 6.1 при $n = 24$. Второй пример является задачей со случайными коэффициентами, ко-

Таблица 6.6: Характеристики тестовых примеров

Name	Iters	Time	n	α
RND-1	$0.17 \cdot 10^9$	33.04	10^4	2.9
RND-2	$0.48 \cdot 10^9$	91.23	10^4	2.52
RND-3	$6.2 \cdot 10^9$	1178.9	10^3	857
RND-4	$9.2 \cdot 10^9$	1758.5	10^3	920
RND-5	$2.4 \cdot 10^9$	464.44	10^3	667

торая входит в состав библиотеки PEBBL [197]. В таблице 6.5 приведены времена решения этих примеров с помощью предлагаемого в диссертационной работе подхода (BNB-Solver) и двух других оптимизационных пакетов, при различном числе ядер. Эксперименты проводились на суперкомпьютере MVS-100K. С помощью библиотеки ALPS не удалось решить задачу 2 за разумное время. Из приведенной таблицы становится ясно, что все три библиотеки обеспечивают относительно хорошую масштабируемость. Однако BNB-Solver по крайней мере в 10 раз быстрее. Это объясняется применяемой в BNB-Solver эффективной схемой реализации метода ветвей и границ на базе вектора, тогда как другие библиотеки использовали менее эффективную реализацию на базе пула подзадач.

Рассмотренные тестовые примеры обладают тем свойством, что число вершин в дереве ветвлений не зависит от выбранной стратегии ветвления. Для таких примеров применение эвристик неэффективно. Числовой характеристикой, приближенно отражающей возможный эффект от применения эвристик, является отношение числа итераций алгоритма Горовица-Сахни в ситуации, когда оптимум неизвестен заранее, к числу итераций этого же алгоритма, когда перед первым шагом значение рекорда полагается равным оптимуму. В последнем случае работа алгоритма сводится к доказательству оптимальности. Будем называть эту величину коэффициентом восприимчивости к эвристикам. Коэффициент восприимчивости не может быть меньше единицы. Значения этого коэффициента, близкие к единице, указывают на невысокую вероятность того,

что применение эвристик позволит существенно ускорить решение задачи. Напротив, задачи с высоким коэффициентом восприимчивости могут значительно выиграть от применения эвристических алгоритмов. Очевидно, что использование комбинированного параллельного алгоритма, сочетающего эвристики и точные методы, даст ожидаемый эффект только для задач с достаточно высоким коэффициентом восприимчивости. В таблице 6.6 приведены следующие характеристики пяти таких задач:

Name — название задачи;

Iters — приближенное количество итераций последовательного алгоритма Горовица-Сахни;

Time — время решения на одном процессоре методом Горовица-Сахни;

n — число переменных задачи;

α — коэффициент восприимчивости к рекорду.

Задачи RND-1, RND-2 были сгенерированы по следующему правилу: $n = 10^4, w_i = 2(10^3+\xi), p_i = 2(10^3+\zeta), C = 10^7+1$ а задачи RND-3, RND-4, RND-5 — следующим образом: $n = 10^3, w_i = 2(10^4+\xi), p_i = 2(10^4+\zeta), C = 10^7+1$, где ξ и ζ — равномерно распределенные случайные величины в диапазоне $[-2000, 2000]$.

Для экспериментов было выделено 16 процессоров вычислительного комплекса MVS-100K. Число процессоров, выполняющих эвристический поиск и задействованных в МВГ, варьируется. Рассмотрим подробно результаты измерений для задачи RND-1, приведенные в табл. 6.7. В таблице приведены результаты измерений времени работы параллельного алгоритма и суммарного (по всем процессорам, выполняющим МВГ) числа итераций. Приведенные результаты показывают, что суммарное количество итераций увеличивается с ростом доли процессов, выполняющих МВГ. Это объясняется тем, что применение эвристик позволяет ускорить процесс нахождения оптимума, и задержками в передаче рекорда. При этом, если все 16 процессоров задействованы в выполнении

Таблица 6.7: Результаты экспериментов

МВГ/Эвристики	Время	Итерации МВГ
1/15	12.24	$0.061 \cdot 10^9$
2/14	6.37	$0.062 \cdot 10^9$
3/13	4.74	$0.063 \cdot 10^9$
4/12	4.27	$0.064 \cdot 10^9$
5/11	4.05	$0.065 \cdot 10^9$
6/10	4.00	$0.072 \cdot 10^9$
7/9	3.86	$0.074 \cdot 10^9$
8/8	3.66	$0.076 \cdot 10^9$
9/7	3.45	$0.079 \cdot 10^9$
10/6	3.24	$0.08 \cdot 10^9$
11/5	3.27	$0.082 \cdot 10^9$
12/4	3.48	$0.093 \cdot 10^9$
13/3	3.39	$0.093 \cdot 10^9$
14/2	3.63	$0.104 \cdot 10^9$
15/1	5.25	$0.203 \cdot 10^9$
16/0	17.16	$0.927 \cdot 10^9$

МВГ, то суммарное число итераций превосходит число итераций в последовательном варианте в 5.4 раза.

Можно сделать следующее наблюдение: при большой доле процессоров, выполняющих эвристический поиск, рекорд находится быстро, что приводит к существенному сокращению общего числа ветвлений. При этом не удается достичь значительного ускорения, так как на МВГ не выделяется достаточного процессорного ресурса. Напротив, когда большая часть процессоров задействована в выполнении МВГ, не хватает процессоров для эффективной работы эвристических алгоритмов, что приводит к увеличению числа ветвлений и снижает эффективность. Оптимальное соотношение числа процессоров, задействованных в выполнении МВГ и эвристическом поиске, является одним из важнейших параметров алгоритма и определяется особенностями конкретной задачи.

В табл. 6.8 приведены минимальное время работы алгоритма и распределение процессов между эвристиками и МВГ, при котором достигается минимум времени работы. Вычисления выполнялись на 16 процессорах вычислительного

Таблица 6.8: Оптимальное соотношение эвристик и МВГ

Задача	Время работы (с)	Оптимальное соотношение МВГ/эвристики
RND-1	3.21	10/6
RND-2	8.85	12/4
RND-3	0.46	11/5
RND-4	1.13	8/8
RND-5	0.69	14/2

комплекса MVS-100K.

6.5 Программный комплекс для решения задач оптимизации большой размерности в распределенной вычислительной среде

6.5.1 Архитектура программного комплекса

В настоящее время в распоряжении отдельных исследователей и научных коллективов, как правило, имеются ресурсы следующих типов: рабочие станции, небольшие многопроцессорные комплексы, доступные в монопольном режиме, суперкомпьютеры коллективного доступа. Программный комплекс BNB-Grid предназначен для решения задач оптимизации на распределенных системах, состоящих из узлов перечисленного типа. BNB-Grid запускает и организует взаимодействие параллельных приложений, решающих задачу с помощью библиотеки BNB-Solver на различных вычислительных узлах. В результате формируется иерархическая распределенная система: на верхнем уровне части работы распределяются между параллельными приложениями, а далее они распределяются по процессорам средствами библиотеки BNB-Solver.

Ядро системы реализовано на языке программирования Java с использованием промежуточного программного обеспечения Internet Communication Engine (ICE) [208] — аналога CORBA. Каждый вычислительный узел представлен в системе распределенным экземпляром объекта типа CE-Manager — Computing Element Manager (Рис. 6.10). Этот объект предоставляет интерфейс для запуска и остановки приложений на вычислительном узле. Координация работы объ-

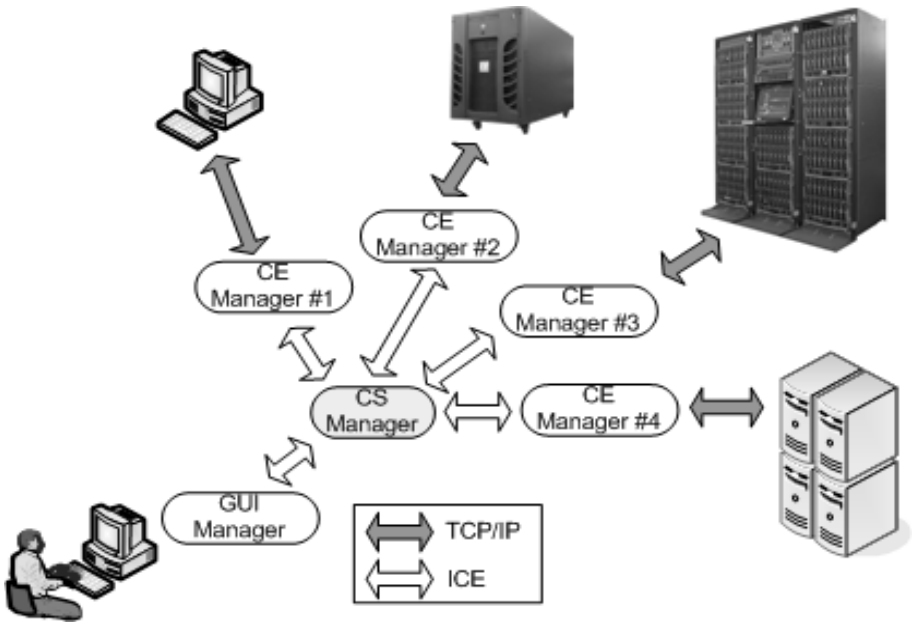

Рис. 6.10: Организация вычислений в BNB-Grid

ектов типа CE-Manager осуществляется с помощью другого распределенного объекта типа CS-Manager — Computing Space Manager. Графический интерфейс пользователя также реализован как ICE-объект, обозначенный на рисунке GUI Manager.

ICE-объекты размещаются либо на одном, либо на нескольких компьютерах в приделах локальной сети. Доступ к удаленному вычислительному узлу осуществляется по следующей схеме (Рис. 6.11). Объект CE-Manager устанавливает SSH-соединение с внешним управляющим модулем узла, запускает на нем процесс CE-Server и устанавливает с ним TCP/IP соединение. Если удаленный узел защищен сетевыми экранами, то применяется механизм туннелирования сетевого соединения через SSH-канал. Процесс CE-Server информирует CE-Manager о состоянии вычислительного узла. При обрыве соединения или перезагрузке узла CE-Server перезапускается. При получении запроса на запуск параллельного приложения CE-Manager создает объект APP-Manager, CE-Server создает процесс APP-Proxy, который запускает параллельное приложение BNB-Solver на узле. При этом CS-Manager взаимодействует с APP-Manager

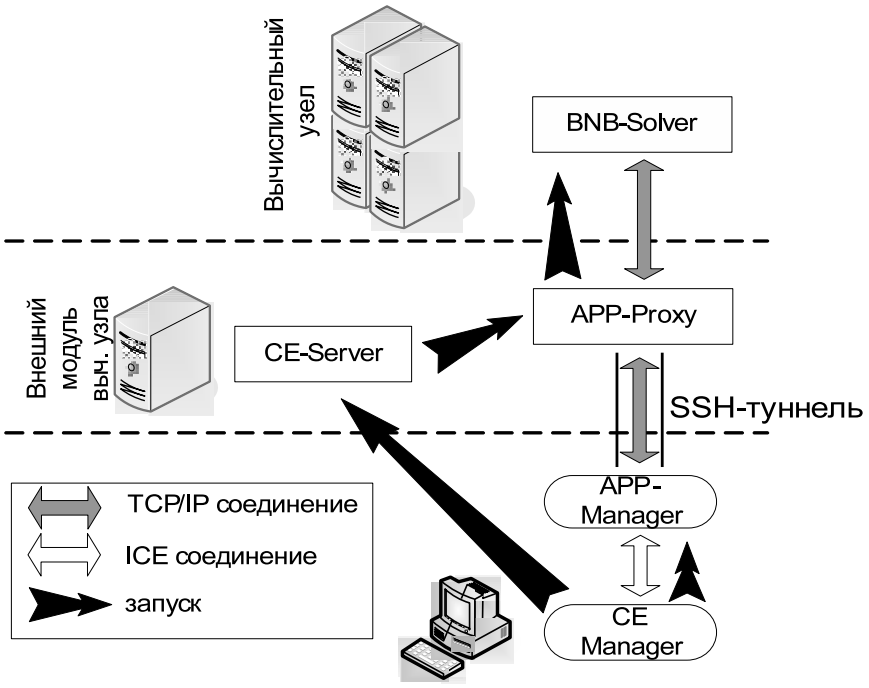

Рис. 6.11: Запуск приложений на удаленном вычислительном узле и установление соединения

средствами ICE, а APP-Manager, в свою очередь, устанавливает TCP/IP соединение с BNB-Solver через APP-Proxy. Процесс APP-Proxy необходим, так как на суперкомпьютерах общего доступа непосредственный доступ из внешней сети к вычислительным модулям, как правило, невозможен. На одном вычислительном узле может быть запущено несколько экземпляров приложения BNB-Solver. Такой подход часто оказывается целесообразным для суперкомпьютеров коллективного доступа, работающих под управлением систем пакетной обработки. На таких системах приложение, запросившее достаточно большое число процессоров может быть надолго помещено в очередь в ожидании соответствующего кокнань в расписании заданий. В то же время, приложение, запросившее существенно меньшее число процессоров, может быть запущено ранее. Этот эффект объясняется тем, что системе пакетной обработки проще эффективно размещать небольшие задания. Поэтому, в проводимых экспериментах на суперком-

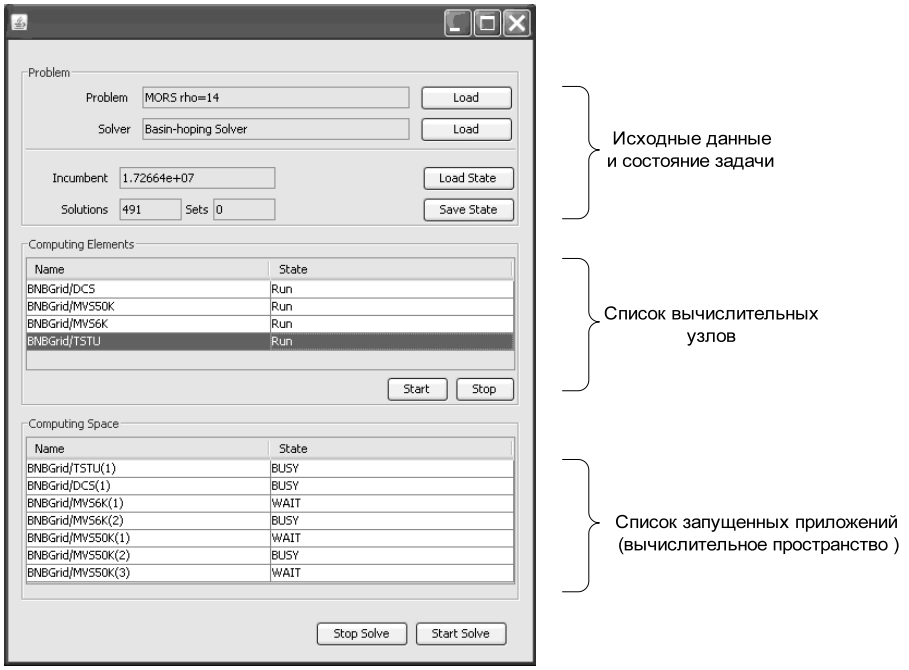

Рис. 6.12: Общий графический интерфейс пользователя системы BNB-Grid

пьютерах коллективного доступа запускалось от пяти до двадцати приложений.

Выделение графического интерфейса пользователя в отдельный ICE-объект преследует две цели. Во-первых, для работы в системе BNB-Grid пользователю достаточно установить на своем персональном компьютере только этот компонент. Во-вторых, при таком подходе графический интерфейс становится легко заменяемым компонентом, который не является неотъемлемой частью системы. Общий графический интерфейс (Рис. 6.12) предоставляет возможности для загрузки исходных данных задачи, управления вычислительным пространством, загрузки и сохранения состояния решения задачи.

Пример специализированного графического интерфейса, ориентированного на рассматриваемую далее задачу поиска энергетически оптимальной конформации молекулярного кластера, представлен на Рис. 6.13. Он позволяет просматривать и модифицировать исходные и полученные в результате расчетов молекулярные кластеры. Такие возможности необходимы для исследования геометрических свойств энергетически-оптимальных кластеров. Вместе с системой хранения конформаций этот интерфейс образует развитую оболочку для иссле-

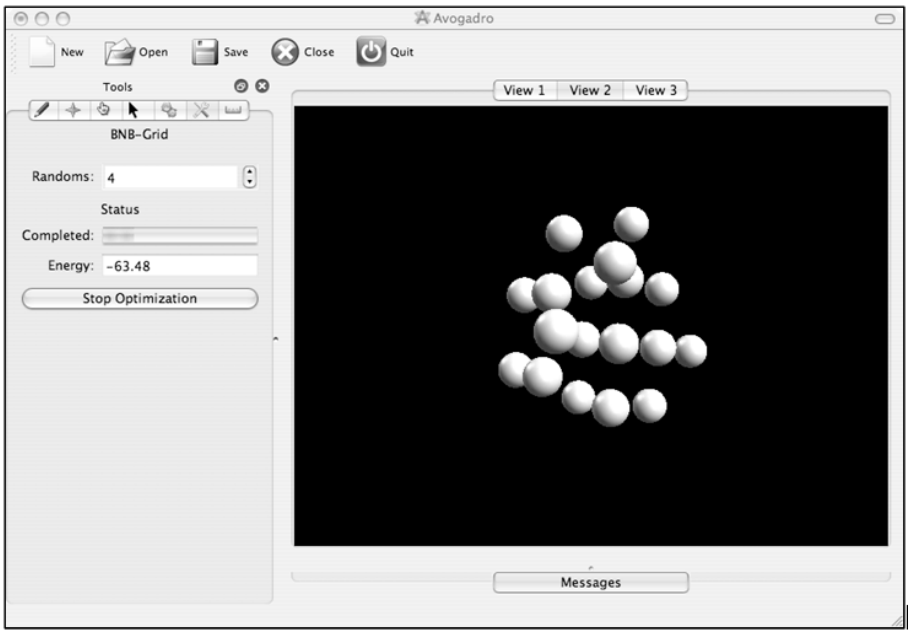

Рис. 6.13: Версия графического интерфейса пользователя системы BNB-Grid для решения задачи поиска энергетически-минимальных конформаций молекулярных кластеров

дования данной задачи. Более подробно эта система рассмотрена в работе [209].

Управляющие модули BNB-Grid и запущенных экземпляров библиотеки BNB-Solver образуют иерархическую систему управления распределением вычислительной нагрузки (Рис. 6.14). На верхнем уровне части работы распределяются ICE-объектом CS-Manager между экземплярами приложений BNB-Solver, а на нижнем — управляющим процессом (Master Process) приложения BNB-Solver между его рабочими процессами (Worker Process). Понятие части работы зависит от применяемого алгоритма: для методов ветвей и границ частями будут подмножества пространства допустимых решений, для эвристических алгоритмов — допустимые решения.

Из-за принудительного завершения приложения системой пакетной обработки, сетевого или аппаратного сбоя, экземпляр BNB-Solver может не завершить обработку выделенной ему части работы. Для предотвращения потери данных в такой ситуации, CS-Manager сохраняет резервные копии передаваемых заданий для каждого из экземпляров запущенных приложений. В случае аварийного завершения любого из них, соответствующая резервная копия возвращается

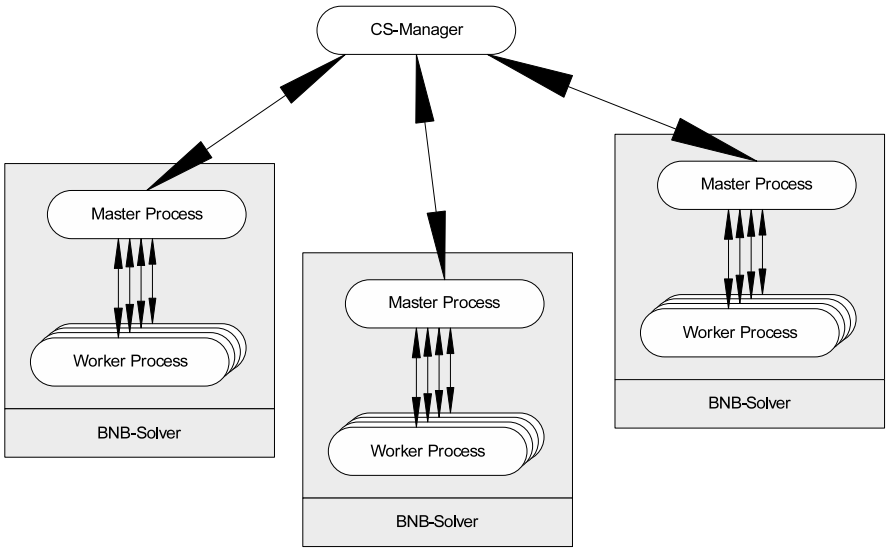

Рис. 6.14: Управление распределением вычислительной работы в BNB-Grid

в общий пул.

6.5.2 Применение программного комплекса BNB-Grid для нахождения минимума энергии молекулярного кластера

Кластерами принято называть группы близко расположенных, тесно связанных друг с другом атомов, молекул, ионов. Изучение кластеров имеет важнейшее значение для понимания процессов конденсации, расчета электронных и динамических характеристик наноматериалов, создания новых источников света и многих других областей [210]. Одной из фундаментальных задач данного направления является определение геометрической структуры, или, как иногда говорят, конформации кластера, соответствующей минимальной энергии взаимодействия входящих в него частиц. Такие конформации наиболее часто наблюдаются в веществе при определенных условиях, например во время перехода от молекулярного к конденсированному состоянию.

Широко используется модель, при учитываются только парное взаимодействия частиц, входящих в кластер. Энергия взаимодействия двух частиц задается парным потенциалом $v(r)$: функцией одного аргумента, которая определяет зависимость энергии взаимодействия от расстояния между частицами. При

этом энергия взаимодействия всех частиц кластера определяется как сумма энергии парных взаимодействий входящих в него n частиц:

$$E(x) = \sum_{i=1}^{n-1} \sum_{j=i+1}^{n} v(r_{ij}) \tag{6.2}$$

где r_{ij} — расстояние между частицами i и j, а вектор x содержит $3n$ декартовых координат этих частиц. Так как эта модель обычно применяется для моделирования химических веществ, состоящих из одинаковых атомов, то частицы кластера будем также называть атомами. При исследовании различных веществ и процессов применяются различные потенциалы. В данной работе ограничимся рассмотрением трех распространенных потенциалов: Леннарда-Джонса $v_{LJ}(r) = r^{-12} - 2r^{-6}$, Морзе $v_M(r) = e^{\rho(1-r)}(e^{\rho(1-r)} - 2)$ и Дзугутова $v_D(r) = A(r^{-m} - B)e^{c/(r-a)}\Theta(a-r) + Be^{d/(b-r)}\Theta(b-r)$. Эти потенциалы применяются для моделирования структуры различных химических веществ. Кластеры Леннарда-Джонса встречаются в инертных газах, Дзугутова — в структурах металлов и стекол. Параметр ρ в потенциале Морзе позволяет моделировать различные вещества. Обычно рассматривают вариации этого параметра в диапазоне 3-14.

Нахождению глобального минимума функции (6.2) посвящено значительное число работ. Существует база данных [211], в которой перечислены наименьшие найденные значения энергии и соответствующие конформации для различного числа атомов и разных потенциалов взаимодействия. Для нахождения минимума функции энергии применялись различные методы, которые можно условно разделить на две категории. К первой относятся подходы, не использующие специфичные для данной задачи свойства целевой функции, т.е. неспециализированные методы оптимизации. Во вторую категорию входят методы, использующие специфику задачи. Начнем с рассмотрения второй категории. Эти методы основаны на общих геометрических закономерностях, наблюдае-

мых для конформаций с минимальной энергией. Например, было замечено, что подавляющее большинство кластеров в модели Леннарда-Джонса имеют икосаэдральную структуру [212]. Изучение геометрических свойств кластеров Морзе показали, что минимумы могут соответствовать различным типам решеток, но, при этом, тяготеют к симметричной сферической форме [213]. Подобные наблюдения привели к созданию геометрически-обоснованных методов минимизации функции (6.2), которые на данный момент признаны наиболее эффективными для данной задачи.

Несмотря на высокую эффективность геометрически-обоснованных методов, они имеют ряд существенных недостатков. Во-первых, данные методы ориентированы на достаточно узкий класс задач, поэтому для каждой рассматриваемой модели взаимодействия требуется разрабатывать свой метод. В частности, методы, основанные на икосаэдральных решетках, показавшие высокую эффективность при отыскании энергетически-оптимальных конформаций кластеров Леннарда-Джонса, не позволяют столь же эффективно находить минимумы для потенциала Морзе. Во-вторых, априорные предположения о геометрической природе кластера могут стать причиной ошибок в нахождении минимумов, что было продемонстрировано в работе [214]. Поэтому, большое внимание уделяется также решению данной задачи неспециализированными оптимизационными методами. Считается [215], что такие методы обязательно должны применяться для верификации результатов проблемно-ориентированной оптимизации, основанной на геометрических соображениях.

К сожалению, попытки решения рассматриваемой задачи методами, гарантирующими глобальную оптимальность, не увенчались успехом для кластеров из 8 и более атомов [216]. Поэтому в качестве наиболее перспективных подходов к решению данной задачи рассматриваются различные эвристические алгоритмы. Одним из наиболее известных алгоритмов, хорошо зарекомендовав-

ших себя при решении задачи поиска конформации с минимальной энергией взаимодействия, является локально-стохастический метод Monotonic Sequence Basin-Hoping (MSBH), предложенный в [214]. Суть подхода состоит в комбинации незначительных сдвигов в пространстве допустимых решения и локального поиска. Более подробно этот алгоритм рассматривается далее. MSBH позволил найти все известные энергетически-оптимальные конформации кластеров Леннарда-Джонса в диапазоне 3-150 атомов. При этом удалось улучшить некоторые ранее найденные минимумы [108]. Несомненным достоинством данного алгоритма является его простота и применимость для широкого класса задач. Основным недостатком является высокая ресурсоемкость. В частности, до ближайшего времени непосредственное применение этого алгоритма не позволяло получить известные минимумы для кластеров Морзе при в диапазоне 40-80 атомов [109].

Для решения данной задачи был применен программный комплекс BNB-Grid, рассмотренный в предыдущем пункте. В вычислительном эксперименте были задействованы суперкомпьютеры различных организаций (МСЦ РАН, ВЦ РАН, ИСП РАН, ТГУ). Результаты расчетов показали, что объединение ресурсов распределенной вычислительной среды позволяет существенно повысить эффективность вычислений и расширить область применимости данного метода. В частности, с использованием разработанного программного комплекса были получены известные энергетически-оптимальные конформации кластеров Морзе для 70, 75, 80, 85, 90, 100 атомов, что не удавалось сделать ранее на однопроцессорных системах с использованием данного метода. Полученный результат показывает, что консолидация большой вычислительной мощности нескольких суперкомпьютеров позволяет успешно решать задачи оптимизации структуры молекулярных кластеров для размерностей и потенциалов, которые ранее удавалось решить только с применением геометрически-обоснованных

подходах. Этот факт представляется важным, так как применение неспециализированных методов требует меньше усилий по созданию новых алгоритмов, основанных на априорных предположениях о геометрической структуре химического соединения. Кроме того, неспециализированные методы могут рассматриваться как проверочные для проблемно-ориентированным подходов.

Для решения задачи был применен алгоритм MSBH (Monotonic Sequence Basin Hopping). Рассмотрим последовательную версию алгоритма.

Алгоритм MSBH

Параметры:

x_0 — начальное приближение,

N_{max} — макс. число сдвигов,

r — окрестность сдвига.

1. $x := x_0$, $I := 0$
2. Пока $I < N_{max}$
   - Сгенерировать $x' = x + \zeta$, $\|\zeta\| \leq r$ — локальный случайный сдвиг.
   - Применить алгоритм локальной оптимизации $x'' = L(x)$.
   - Если $f(x'') < f(x)$, то $x := x'', I := 0$, иначе $I := I + 1$.
3. Вернуть x.

На практике алгоритм MSBH чаще всего применяется вместе с каким-либо алгоритмом, генерирующим начальные приближения — например, методом Монте-Карло. Мы применяли вариант, предложенный в работе [214], в котором некоторое количество начальных приближений генерируется случайно в кубе $[-1, 1]^{3n}$. Далее сгенерированные точки используются в качестве начальных для MSBH. Такой алгоритм идеально подходит для реализации в среде параллельных и распределенных вычислений, так как различные начальные приближения мо-

Таблица 6.9: Вычислительные узлы

	Название	Архитектура	Число ядер	Местоположение
1	MVS100K	Intel Xeon 5365, 3 GHz	7856	МСЦ РАН(Москва)
2	MVS6K	Intel Itanium II, 2.2 GHz	256 ВЦ РАН (Москва)	
3	OMEGA	Intel Xeon 5355, 2.66 GHz	88	ИСП РАН (Москва)
4	TSTU	Pentium IV, 3.2 GHz	8	ТГУ (Тамбов)
5	DCS	Pentium IV, 3.2GHz	1	ИСА РАН (Москва)

гут обрабатываться независимым образом. Параллельная версия алгоритма построена по схеме "управляющий-рабочие":

Управляющий процесс:

1. Пока $I < M_s$
   - Сгенерировать $x \in X$ — случайная точка.
   - Принять от освободившегося рабочего процесса x_0, $x_r := \min(x_r, x_0)$.
   - Послать свободному рабочему процессу точку x, $I := I + 1$.
2. Вернуть x_r.

Рабочий процесс:

1. Пока не получено условие останова:
   - Принять от управляющего процесса точку x.
   - Применить MSBH к x, получить x_0.
   - Послать x_0 управляющему процессу.

Вычислительные эксперименты проводились для потенциалов Леннарда-Джонса, Дзюгутова и Морзе. В вычислениях были задействованы вычислительные узлы, приведенные в таблице 6.9.На начальной стадии экспериментов узлы 4, 5 были востребованы, так как из-за загруженности суперкомпьютеров коллективного доступа 1,3 не удавалось оперативно получить большое количество вычислительных ядер. В дальнейшем, в связи с появлением

Таблица 6.10: Наилучшие значения r

Потенциал	значение r
Морзе $\rho = 6$	0.4
Морзе $\rho = 14$	0.35
Леннард-Джонс	0.4
Дзюгутов	0.4

доступа на кластер 3 и увеличении числа доступных ядер на МВС 100К, необходимость использовать в расчетах маломощные узлы 4 и 5 практически отпала. Приводимые ниже результаты были получены на распределенной системе, состоящей из узлов 1-3. При этом общее число задействованных процессоров достигало 512. Среднее число колебалось в интервале приблизительно 200-512 и зависело от текущей загруженности используемых кластеров.

Алгоритм MSBH управляется следующими параметрами: количеством начальных приближений, максимальным числом сдвигов N_{max} в алгоритме MSBH до момента улучшения, и максимальным радиусом возмущения r. Сначала для каждого из рассматриваемых потенциалов проводилась небольшая серия предварительных экспериментов для установления наилучшего значения параметра r. Как отмечается в работе [214], это значение практически не зависит от числа атомов, а определяется только видом потенциала. В таблице 6.10 приведены наилучшие значения параметра r для различных потенциалов.

В первом вычислительном эксперименте отыскивался минимум энергии кластера Леннарда-Джонса из 98 атомов. Известно, что данная конформация является одной из наиболее сложных для отыскания [108, 109]. Обнаружение нового глобального минимума для 98 атомов стало заметным событием в свое время, послужившим основанием для публикации [108]. Результат вычислительного эксперимента приведен в строке 1 таблицы 6.11. Параметр N_{max} полагался равным 8192. Для обработки 512 начальных приближений потребовалось 43 минуты, на протяжении которых наименьшее известное на настоящее время значение целевой функции было найдено 8 раз. На основании этого можно сделать вы-

Таблица 6.11: Результаты вычислительного эксперимента

	Потенциал	Число атомов	Число нач. прибл.	N_{max}	Общее время расчетов (мин)	Количество “попаданий”	Найденный минимум	Наилучший известный минимум
1	Леннард-Джонс	98	512	8192	43	8	-543.665361	-543.665361
2	Морзе $\rho = 6$	85	512	8192	63	3	-405.246158	-405.246158
3	Морзе $\rho = 6$	90	512	8192	91	74	-433.355380	-433.355380
4	Морзе $\rho = 6$	100	512	8192	96	98	-488.675685	-488.675685
5	Морзе $\rho = 14$	70	1024	8192	174	9	-292.462856	-292.462856
6	Морзе $\rho = 14$	75	1024	8192	205	2	-318.407330	-318.407330
7	Морзе $\rho = 14$	80	1024	8192	244	3	-340.811371	-340.811371
8	Морзе $\rho = 14$	85	1024	8192	188	5	-363.891261	-363.893075
9	Морзе $\rho = 14$	85	512	32768	372	2	-363.893075	-363.893075
10	Морзе $\rho = 14$	90	1024	8192	266	2	-388.401652	-388.401652
11	Морзе $\rho = 14$	100	1024	8192	232	8	-439.070547	-439.070547
12	Дзюгутов	50	1024	8192	175	2	-104.366189	-104.366189
13	Дзюгутов	100	1024	8192	175	1	-218.678229	-219.523265
14	Дзюгутов	100	1024	32768	371	1	-218.744395	-219.523265

вод о том, что применяемый метод позволяет уверенно находить оптимальную конфигурацию.

Вторая серия экспериментов проводилась для потенциала Морзе при $\rho = 6$. Было выбрано также 512 начальных приближений, Nmax полагалось равным 8192. Результаты расчетов для 85, 90 и 100 атомов приведены в строках 2-4 таблицы 6.11. Удалось улучшить минимум для 85 атомов, ранее найденный с применением параллельных вычислений[17]. Интересно отметить очень большое число попаданий в точку минимума для $N = 90, 100$, и то, что нахождение минимума для $N = 85$ атомов оказалось более сложной задачей по сравнению с $N = 90$ и $N = 100$.

Третья серия экспериментов проводилась для потенциала Морзе при $\rho = 14$, который рассматривается как наиболее сложный с точки зрения отыскания минимума по сравнению с другими значениями параметра. Были взяты 1024 начальных приближения, параметр N_{max} был выбран равным 8192. Результаты экспериментов для 70, 75, 80, 85, 90, 100 атомов приведены в строках 5-10 таблицы 6.11. Для 70, 75, 80, 90 и 100 атомов найденные минимумы совпали с най-

денными с помощью геометрически-обоснованного алгоритма Dynamic Lattice Search [212]. Удалось улучшить минимумы для 85 и 90 атомов, ранее найденные с применением параллельной реализации MSBH [217].

Следующая серия экспериментов проводилась для потенциала Дзюгутова (таблица 3, строки 11-13). Известные значения минимумов были получены с использованием алгоритма [215], предложенного Дойе, который отличается от MSBH тем, что модификация решений проводится на основе геометрических соображений (сдвигаются атомы), а также для получения конфигурации из $N+1$ атома используется конфигурация из N атомов, полученная при предыдущих расчетах. Были проведены расчеты для 50 и 100 атомов. Для 50 атомов те же параметры, что и для кластеров Морзе дали хорошие результаты Ц минимум был найден. Для 100 атомов при первоначальном расчете с $N_{max} = 8192$ было получено значение, отличающееся от известного минимума. При пересчете с $N_{max} = 32768$ было получено несколько лучшее значение, но все равно, превосходящее известный минимум. При этом время расчетов приблизительно составило 12 часов. Полученные результаты позволяют сделать вывод о том, что кластеры Дзюгутова являются более трудными для обнаружения по сравнению с кластерами Морзе и Леннарда-Джонса и для успешного отыскания минимумов в диапазоне $50 - 100$ атомов требуется консолидация значительных вычислительных ресурсов. Результаты экспериментов показали, что суммарная вычислительная мощность нескольких суперкомпьютеров позволяет существенно раздвинуть границы применимости этого метода и получать оптимальные либо близкие к оптимальным конформации за приемлемое время для различных потенциалов межатомного взаимодействия. Были получены минимальные значения энергии взаимодействия и соответствующие им пространственные конфигурации для размерностей $N > 70$ для потенциала Морзе, что ранее не удавалось сделать с помощью данного метода.

Полученный результат показывает, что консолидация большой вычислительной мощности нескольких суперкомпьютеров позволяет успешно решать задачи оптимизации структуры молекулярных кластеров для размерностей и потенциалов, которые ранее удавалось решить только с применением геометрически-обоснованных подходах. Этот факт представляется важным, так как применение неспециализированных методов требует меньше усилий по созданию новых алгоритмов, основанных на априорных предположениях о геометрической структуре химического соединения. Кроме того, неспециализированные методы могут рассматриваться как проверочные для проблемно-ориентированным подходов.

6.6 Основные результаты главы

В главе 6 получены следующие результаты:

1. Предложен подход к реализации детерминированных и гибридных методов глобальной оптимизации на многопроцессорных вычислительных комплексах, основанный на разделении вычислительной, коммуникационной и управляющей функциональности в программе. Разработан язык описания управления параллельным процессом поиска глобального экстремума, основанный на формализме конечных автоматов. Этот язык позволяет описывать как управление процессом оптимизации, так и балансировку вычислительной нагрузки между процессорами параллельной системы. Такой подход позволяет независимым образом реализовывать и отлаживать соответствующие компоненты, а также, облегчать процесс расширения программного комплекса за счет повторного использования ранее разработанных компонент.

2. На основе разработанного подхода реализован программный комплекс BNB-

Solver для решения задач глобальной оптимизации на последовательных и параллельных вычислительных системах. В рамках этого программного комплекса были реализованы различные алгоритмы для решения задач конечномерной оптимизации с одним и несколькими критериями, проведены многочисленные эксперименты на различных вычислительных платформах. В частности, разработана и выполнена эффективная параллельная реализация метода ветвей и границ для задачи о булевом ранце с одним ограничением, превосходящая по скорости работы известные аналоги.

3. Разработан подход к эффективному использованию распределенных систем при решении задач глобальной оптимизации. В состав распределенной системы могут входить суперкомпьютеры, высокопроизводительные серверы, сегменты сервисных грид-систем. На основе разработанного подхода создан программный комплекс BNB-Grid, который позволяет использовать в процессе решения сразу несколько вычислительных узлов.

4. Разработанный программный комплекс BNB-Grid был успешно применен автором для решения задачи поиска энергетически-минимальных молекулярных кластеров, а также совместно с другой исследовательской группой для решения булевых уравнений большой размерности. Задача решения булевых уравнений рассмотрена в ПРИЛОЖЕНИИ 1.

Заключение

Основные результаты диссертационной работы

В диссертационной работе всесторонне рассмотрены методы решения задач глобальной оптимизации с одним и несколькими критериями. Предложены новые методы решения таких задач, получены оценки вычислительной сложности различных методов решения, разработаны эффективные реализации таких методов на современных вычислительных платформах, в том числе на параллельных и распределенных системах. Получены следующие результаты:

1. Предложены новые алгоритмы решения задач оптимизации задач с одним критерием, основанные на идее неравномерных покрытий. В частности, получены новые миноранты и нижние оценки для минимизации скалярных функций. Предложены новые миноранты, основанные на оценке спектра матрицы Гессе, разработан метод вычисления таких оценок. Для задач оптимизации с параллелепипедными ограничениями предложены новые миноранты, учитывающие необходимые условия оптимальности. Предложены способы сокращения области поиска, основанные на построении дополнения покрывающего множества до параллелепипеда, позволяющие существенно повысить быстродействие метода.

2. Метод неравномерных покрытий для задач многокритериальной оптимизации обобщен на случай произвольных минорант, доказана конечность числа шагов метода при определенных ограничениях на задачу. Получены новые

теоретические свойства приближенных решений для задач с параллелепипедными ограничениями: доказаны новые свойства ε-Парето множества, устанавливающие связь между этим множеством, границей Парето и оболочкой Эджворта-Парето. Для задач многокритериальной оптимизации с функциональными ограничениями введено новое понятие приближенного решения — множество ε, δ-Парето и предложен алгоритм для его построения, доказана сходимость метода.

3. Введено понятие эффективной оболочки множества, обобщающее понятие оболочки Эджворта-Парето для многокритериальных задач. Показано, что эта оболочка содержится в его выпуклой оболочке. Введены понятия ε-эффективной границы и ε-оболочки множества, предложены методы их численного построения. Разработанные понятия и методы применены для задачи построения внешней границы рабочей области многосекционного робота-манипулятора с заданной точностью.

4. Разработаны методы получения оценок сложности различных вариантов метода ветвей и границ. Разработан подход к оценке сложности метода неравномерных покрытий, с помощью которого получены верхние оценки сложности различных вариантов этого метода для различных классов задач. Получены оценки сложности метода ветвей и границ для задачи о сумме подмножеств, зависящие от коэффициентов задачи. Проведено экспериментальное исследование и сопоставление их точности. Исследован вопрос параллельной вычислительной сложности одной из возможных реализаций метода ветвей и границ. Получена общая оценка вычислительной сложности, не зависящая от конкретной задачи. Для задачи о сумме подмножеств получены асимптотические формулы для минимальной параллельной сложности, максимального параллельного ускорения и эффективности.

5. Предложен подход к реализации детерминированных и гибридных методов глобальной оптимизации на многопроцессорных вычислительных комплексах, основанный на разделении вычислительной, коммуникационной и управляющей функциональности в программе. На основе разработанного подхода реализован программный комплекс для решения задач глобальной оптимизации на последовательных, параллельных и распределенных системах. В рамках этого программного комплекса были реализованы различные алгоритмы для решения задач конечномерной оптимизации с одним и несколькими критериями, проведены многочисленные эксперименты на различных вычислительных платформах. В частности, разработана и выполнена эффективная параллельная реализация метода ветвей и границ для задачи о булевом ранце с одним ограничением, превосходящая по скорости работы известные аналоги.

Перспективы дальнейших разработок

Результаты диссертации могут быть развиты по нескольким направлениям. Во-первых, это дальнейшее совершенствование методов, предложенных в работе. Перспективным представляется использование техники отсечений для ускорения сходимости методов. Также представляют интерес комбинированные настраиваемые алгоритмы, в которых сочетаются глобальные и локальные методы. Такие алгоритмы удобно реализовывать в рамках программного комплекса BNB-Solver, в котором модуль управления вычислениями вынесен в отдельную часть.

Также большой интерес представляет перенос алгоритмов решения задач, рассматриваемых в работе на гибридные вычислительные системы, сочетающие различные виды параллелизма. В частности, графические процессоры и обычные вычислительные узлы. Перенос алгоритмов на подобные платформы

ставит ряд новых интересных задач, связанных с распределением вычислений по узлам неоднородной системы.

Безусловно в дальнейшем развитии нуждаются и оценки сложности, полученные в диссертационной работе. В частности, целесообразно расширить класс рассматриваемых алгоритмов, включив в него методы динамического программирования, методы отсечений. Также интересно дальнейшее развитие теории сложности для параллельных вариантов методов глобальной оптимизации.

Благодарности

В заключении автор хотел бы выразить глубокую благодарность своему научному консультанту академику РАН Ю.Г. Евтушенко, поставившему перед автором ряд задач, решение которых составило наиболее важную часть диссертации и под руководством которого автор решал эти задачи. Также диссертант благодарен профессору И.Х. Сигалу, открывшему для автора мир дискретной оптимизации, под руководством которого также были получены ряд важных результатов, вошедших в диссертацию. Неоценимую помощь в работе над диссертацией оказал профессор А.П. Афанасьев, который предложил использовать распределенные вычислительные системы для решения задач глобальной оптимизации и курировал эту работу. Очень плодотворным было сотрудничество с профессор Р.М. Колпаковым, результатом которого стали оценки сложности метода ветвей и границ для задачи о ранце, вошедшие в данную работу. Также автор благодарен профессорам А.В. Лотову, А.П. Карпенко, А.А. Лазареву, Я.Д. Сергееву сотрудникам ВЦ РАН А.И. Голикову, К.К. Абгарян, сотрудникам ИДСТУ РАН Семенову А.А., Заикину О.С. и другим коллегам по работе за поддержку и полезные советы в процессе работы над диссертацией.

Литература

[1] Anikin A., Gornov A. An implementation of Newton's method for Keating's potential optimization problems //Stud. Inform. Univ. - 2011. - T. 9. - №3. - C. 11-20.

[2] Романов А.Н. и др. Компьютерный дизайн лекарственных средств: программа докинга SOL //Вычислительные методы и программирование. - 2008. - Т. 9. - №1.

[3] Гаврилов С.В., Глебов А.Л., Стемпковский А.Л. Структурная оптимизация цифровых КМОП схем // Информационные технологии и вычислительные системы. 2002. №4. С. 40-47.

[4] Кулешов А.П., Бернштейн А.В., Бурнаев Е.В. Модели знаний в предсказательном метамоделировании. Труды Международной научно-технической конференции кИнформационные технологии и математическое моделирование системь (ИТММ, 19-26 сентября 2010 г., Франция). М.: Учреждение российской академии наук Центр информационных технологий в проектировании РАН, 2010. С. 209-210.

[5] Parkinson A.R., Balling R., and Hedengren J.D. Optimization Methods for Engineering Design, Brigham Young University, 2013.

[6] Афанасьев А.П., Гринберг Я.Р., Курочкин И.И. Алгоритм, реализующий критерий уменьшения неравномерности при заполнении сети многопродуктовым потоком. В кн.: Декомпозиционные методы в математическом

моделировании и информатике. Тезисы докладов 2-й московской конференции // М.: ВЦ РАН, 2004. С.12-14.

[7] Гимади Э.Х., Истомин А.М., Рыков И.А. О задаче нескольких коммивояжчров с ограничениями на пропускные способности рчбер графа // Дискретн. анализ и исслед. опер. Т. 20 №5. 2013. С. 13-30.

[8] Лазарев А.А., Мусатова Е.Г. Целочисленные постановки задачи формирования железнодорожных составов и расписания их движения // УБС. №38. 2012. С. 161-169.

[9] Колоколов А.А., Куряченко А.В. Разработка алгоритмов решения одной задачи размещения предприятий с интервальными данными // Динамика систем, механизмов и машин: материалы VII Международной научно-технической конференции (Омск, 10 ноября–12 ноября 2009), 3, ред. А. В. Косых, О. И. Бабенко, Издательство ОмГТУ, Омск. 2009. С. 47-51.

[10] Девятерикова М.В., Колоколов А.А., Колосов А.П. Об одном подходе к решению дискретной задачи планирования производства с интервальными данными // Тр. ИММ УрО РАН, 14, №2. 2008. С. 48–57.

[11] Лазарев А.А., Кварацхелия А.Г. Свойства оптимальных расписаний задачи теории расписаний минимизации суммарного взвешенного момента окончания для одного прибора // Автомат. и телемех., 2010, №10, 80-89

[12] Гончар Д., Фуругян М. Алгоритмы составления многопроцессорного расписания для неоднородного множества работ с директивными интервалами и произвольными процессорами // Системы управления и информационные технологии. 2013. Т. 3, №1. С. 204-208.

[13] Kukushkin A.B., Neverov V.S., Marusov N.L., Semenov I.B., Kolbasov B.N., Voloshinov V.V., Afanasiev A.P., Tarasov A.S., Stankevich V.G., Svechnikov

N.Yu., Veligzhanin A.A., Zubavichus Ya.V., Chernozatonskii L.A. Few-nanometer-wide carbon toroids in the hydrocarbon lms deposited in tokamak T-10 // Chemical Physics Letters, №506. 2011. С. 265-268.

[14] Воронцов К.В. Оптимизационные методы линейной и монотонной коррекции в алгебраическом подходе к проблеме распознавания //Журнал вычислительной математики и математической физики. 2000. Т. 40. №1. С. 166-176.

[15] Левитин Е.С., Поляк Б.Т. Методы минимизации при наличии ограничений // Журнал вычислительной математики и математической физики. 1966. Т. 6. №5. С. 787-823.

[16] Поляк Б.Т. Градиентные методы минимизации функционалов // Журнал вычислительной математики и математической физики. 1963. Т. 3. №4. С. 643-653.

[17] Nesterov Y. Efficiency of coordinate descent methods on huge-scale optimization problems // SIAM Journal on Optimization. 2012. Т. 22. №2. С. 341-362.

[18] Nesterov Y. Gradient methods for minimizing composite functions //Mathematical Programming. 2013. Т. 140. №. 1. С. 125-161.

[19] Кочетов Ю.А. Вычислительные возможности локального поиска в комбинаторной оптимизации // Ж. вычисл. матем. и матем. физ. Т. 48 №5. 2008. С. 788-807.

[20] Кочетов Ю.А., Плясунов А.В. Генетический локальный поиск для задачи о разбиении графа на доли ограниченной мощности // Ж. вычисл. матем. и матем. физ. Т. 52 №1. 2012. С. 164-176.

[21] Журавлев Ю.И. Локальные алгоритмы вычисления информации // Кибернетика, 1965. №1. С. 12-19.

[22] Дюкова Е.В., Журавлев Ю.И., Рудаков К.В. Об алгебро-логическом синтезе корректных процедур распознавания на базе элементарных алгоритмов // Ж. вычисл. матем. и матем. физ. Т. 36 №8. 1996. С. 215-223

[23] Рудаков К.В. Полнота и универсальные ограничения в проблеме коррекции эвристических алгоритмов классификации. Кибернетика. 1987.

[24] Финкельштейн Ю.Ю. Приближенные методы и прикладные задачи дискретного программирования. М.: Наука, 1976.

[25] Меламед И.И., Сергеев С.И., Сигал И.Х. Задача коммивояжера. Приближенные алгоритмы // Автомат. и телемех. 1989. №11. С. 3-26.

[26] Дюбин Г.Н., Корбут А.А. Поведение в среднем жадных алгоритмов для минимизационной задачи о ранце - общие распределения коэффициентов // Ж. вычисл. матем. и матем. физ. Т. 48 №9. 2008. С. 1556-1570.

[27] Граничин О. Н., Поляк Б. Т. Рандомизированные алгоритмы оптимизации и оценивания при почти произвольных помехах. М. : Наука, 2003.

[28] Карпенко А. П., Митина Е. В., Семенихин А. С. Когенетический алгоритм Парето-аппроксимации в задаче многокритериальной оптимизации / Карпенко А. П., Митина Е. В., Семенихин А. С. // Информационные технологии. 2013. №1. С. 22-32.

[29] Гладков Л.А., Курейчик В.В., Курейчик В.М. Генетические алгоритмы. Учебное пособие М. Физматлит. 2004.

[30] Glover F., Kochenberger G. A. (ed.). Handbook of metaheuristics. Springer, 2003.

[31] Kochenberger G.A., McCarl B.A., Paul Wyman F. A heuristic for general integer programming //Decision Sciences. 1974. Т. 5. №. 1. С. 36-44.

[32] Neri F., Cotta C., Moscato P. (ed.). Handbook of memetic algorithms. Springer, 2011. Т. 379.

[33] Glover F., Laguna M., Marti R. Fundamentals of scatter search and path relinking //Control and cybernetics. 2000. Т. 39. №. 3. С. 653-684.

[34] Поляк Б.Т., Цыпкин Я.З. Робастные псевдоградиентные алгоритмы адаптации //Автоматика и телемеханика. - 1980. №10. С. 91-97.

[35] Меламед И.И., Сигал И.Х., Исследование параметров алгоритмов ветвей и границ решения симметричной задачи коммивояжера // Автомат. и телемех., 1997, №10, С. 186-192.

[36] Финкельштейн Ю.Ю. О полиномиальном алгоритме ε-оптимизации в многомерной задаче о ранце // Ж. вычисл. матем. и матем. физ. Т. 20 №3. 1980. С. 800–802.

[37] Черенин В. П., Хачатуров В. Р. Решение методом последовательных расчетов одного класса задач о размещении производства //Экономико–математические методы. М.: Наука. 1965. №2. С. 279-290.

[38] Хачатуров В.Р. Аппроксимационно-комбинаторный метод и некоторые его приложения // Ж. вычисл. матем. и матем. физ. Т. 14 №6. 1974. С. 1464-1487.

[39] Лазарев А.А. Графический подход к решению задач комбинаторной оптимизации. Автомат. и телемех. 2007 №4. С. 13-23.

[40] Колоколов А.А. Регулярные разбиения и отсечения в целочисленном программировании. Сиб. журн. исслед. опер. Т. 1 №2. 1994, С. 18–39

[41] Dantzig G. B. et al. The generalized simplex method for minimizing a linear form under linear inequality restraints //Pacific Journal of Mathematics. 1955. Т. 5. №. 2. С. 183-195.

[42] Balas E. An additive algorithm for solving linear programs with zero-one variables //Operations Research. 1965. Т. 13. №. 4. - С. 517-546.

[43] Pisinger D., Ropke S. A general heuristic for vehicle routing problems //Computers & operations research. 2007. Т. 34. №. 8. С. 2403-2435.

[44] Lodi A., Martello S., Monaci M. Two-dimensional packing problems: A survey //European Journal of Operational Research. 2002. Т. 141. №. 2. С. 241-252.

[45] Christofides N., Mingozzi A., Toth P. Exact algorithms for the vehicle routing problem, based on spanning tree and shortest path relaxations //Mathematical programming. 1981. Т. 20. №. 1. С. 255-282.

[46] Евтушенко Ю.Г. Численный метод поиска глобального экстремума функций (перебор на неравномерной сетке) // ЖВМ и МФ. 1971. Т. 11. №6, С. 1390-1403.

[47] Евтушенко Ю.Г., Ратькин В.А. Метод половинных делений для глобальной оптимизации функции многих переменных // Техническая кибернетика. 1987. №1. С. 119-127.

[48] Евтушенко Ю.Г., Малкова В.У., Станевичюс А.А. Параллельный поиск глобального экстремума функций многих переменных // ЖВМ и МФ. 2009. Т. 49. №2. С. 255-269.

[49] Пиявский С. А. Один алгоритм отыскания абсолютного экстремума функций //Журнал вычислительной математики и математической физики. 1972. Т. 12. №. 4. С. 888-896.

[50] Стронгин Р. Г. Численные методы в многоэкстремальных задачах:(Информационно-статистические алгоритмы). Наука. Гл. ред. физ.-мат. лит. 1978.

[51] Гергель В. П., Гришагин В. А., Городецкий С. Ю. Современные методы принятия оптимальных решений. Нижний Новгород. 2001.

[52] Сергеев Я. Д., Квасов Д. Е., Диагональные методы глобальной оптимизации, Физматлит, М.:, 2008, 352 стр.

[53] Стрекаловский А.С. Элементы невыпуклой оптимизации. Новосибирск: Наука, 2003. 356 с.

[54] Волошинов В. В., Левитин Е. С. Приближенная глобальная минимизация невыпуклых функций, близких к выпуклым // Ж. вычисл. матем. и матем. физ. Т. 37 №7. 1997. С. 771-784.

[55] Булатов В. П., Касинская Л. И. Некоторые методы минимизации вогнутой функции на выпуклом многограннике и их приложения/Методы оптимизации и их приложения //Методы оптимизации и их приложения. Новосибирск: Наука. Сиб. отд-ние. 1982. С. 32-35.

[56] О. В. Хамисов, Невыпуклая оптимизация с нелинейными опорными функциями, Тр. ИММ УрО РАН, 19, №2, 2013, 295-306

[57] Hansen E., Sengupta S. Bounding solutions of systems of equations using interval analysis //BIT Numerical Mathematics. 1981. Т. 21. №. 2. С. 203-211.

[58] Kearfott R. B. Rigorous global search continuous problems. Springer, 1996.

[59] Pinter J. Branch-and bound algorithms for solving global optimization problems with Lipschitzian structure //Optimization. – 1988. – Т. 19. – №. 1. – С. 101-110.

[60] Horst R., Tuy H. Global optimization: Deterministic approaches. – Springer, 1996.

[61] Березкин В. Е., Каменев Г. К., Лотов А. В. Гибридные адаптивные методы аппроксимации невыпуклой многомерной границы Парето //Журнал вычислительной математики и математической физики. 2006. Т. 46. №. 11. С. 2009-2023.

[62] Bushenkov V. A., Lotov A. V. Methods and algorithms for analysis of linear systems of the construction of generalized attainability sets //Zhurnal Vychislitel'noi Matematiki i Matematicheskoi Fiziki. 1980. Т. 20. №. 5. С. 1130-1141.

[63] Podinovskiy, Vladislav V. Sensitivity analysis for choice problems with partial preference relations (английский). // European Journal of Operational Research, 2012. Т. 221. №1. С. 198-204

[64] Garcia-Gonzalo J. et al. A decision support system for a multi stakeholderЎs decision process in a Portuguese National Forest //Forest Systems. 2013. Т. 22. №. 2. С. 359-373.

[65] Подиновский В.В., Ногин В.Д. Парето-оптимальные решения многокритериальных задач. - М.: Физматлит, 2007, 256с.

[66] Соболь И. М., Статников Р. Б. Выбор оптимальных параметров в задачах со многими критериями. - Наука, 1981.

[67] Statnikov R. B. and Matusov J. Multicriteria optimization and engineering. NJ: Chapman and Hall, 1995

[68] Statnikov R. B. Multicriteria design: Optimization and identification. – Springer, 1999. – Т. 26.

[69] Deb K. Multi-objective optimization using evolutionary algorithms. Wiley. Chichester. 2001.

[70] Ehrgott M., Lohne A., Shao L.. A dual variant of BensonЎs outer approximation algorithm. Report 654. The University of Auckland. 2007.

[71] Zitzler E., Knowles J., Thiele L. Quality Assessment of Pareto Set Approximations // In Multi-objective Optimization - Interactive and Evolutionary Approaches, Springer LNCS 5252. P.373-404. 2008.

[72] Benson H.P. An outer approximation algorithm for generating all efficient extreme points in the outcome set of a multiple objective linear programming problem // J. Global Optim. Vol. 13 (1998). P. 1–24.

[73] Pascoletti A., Zanolin F. Example of a suspension bridge ODE model exhibiting chaotic dynamics: a topological approach //Journal of Mathematical Analysis and Applications. 2008. Т. 339. №. 2. С. 1179-1198.

[74] Pascoletti A., Serafini P. Scalarizing vector optimization problems. J. Optim. Theory Appl. Vol. 42, No. 4(1984), pp. 499-524.

[75] Miettinen K. Nonlinear Multiobjective Optimization. Kluwer Academic Publishers, Boston, 1999.

[76] Евтушенко Ю. Г., Потапов М. А. Методы численного решения многокритериальных задач //ДАН СССР. 1986. Т. 291. №. 1. С. 25-29.

[77] Меламед И. И., Сигал И. Х. Вычислительное исследование трехкритериальных задач о деревьях и назначениях //Журнал вычислительной математики и математической физики. 1998. Т. 38. №. 10. С. 1780-1787.

[78] Меламед И. И., Сигал И. Х. Вычислительное исследование линейной параметризации критериев в многокритериальном дискретном программиро-

вании // Журнал вычислительной математики и математической физики. 1996. Т. 36. №. 10. С. 23-25.

[79] Батищев Д. И., Коган Д. И., Лейкин М. В. Алгоритмы синтеза решений для многокритериальной многомерной задачи о ранце //Информационные технологии. 2004. №. 1. С. 18-27.

[80] Агеев А. А., Кельманов А. В., Пяткин А. В. Сложность задачи о разрезе максимального веса в евклидовом пространстве //Дискретный анализ и исследование операций. 2014. Т. 21. №4. С. 3-11.

[81] Кельманов А. В., Пяткин А. В. О сложности некоторых задач кластерного анализа векторных последовательностей //Дискретный анализ и исследование операций. 2013. Т. 20. №. 2. С. 47-57.

[82] Хачай М.Ю. Вопросы вычислительной сложности процедур обучения распознаванию в классе комитетных кусочно-линейных решающих правил //Автоматика и телемеханика. 2010. №. 3. С. 178-189.

[83] Хачай М.Ю., Поберий М.И. Вычислительная сложность и аппроксимируемость серии геометрических задач о покрытии //Тр. ИММ УрО РАН. Т. 18. №3. 2012. С. 247Ц260.

[84] Кочетов Ю.А., Пащенко М.Г., Плясунов А.В. О сложности локального поиска в задаче о p-медиане //Дискретный анализ и исследование операций. 2005. Т. 12. №2. С. 44-71.

[85] Плясунов А.В. Параллельная сложность. Теория и приложения // Методы оптимизации и их приложения. Труды XIV Байкальской международной школы-семинара. Иркутск, 2008, С. 147-158.

[86] Заозерская Л.А., Колоколов А.А. Оценки среднего числа итераций для некоторых алгоритмов решения задачи об упаковке множества //Журнал

вычислительной математики и математической физики. 2010. Т. 50. №. 2. С. 242-248.

[87] Еремеев А.В., Заозерская Л.А., Колоколов А.А. Задача о покрытии множества: сложность, алгоритмы, экспериментальные исследования // Дискретный анализ и исследование операций. 2000. Т. 7. №. 2. С. 22-46.

[88] Eiben A., Smith J. Introduction to Evolutionary Computing. Springer. 2008.

[89] Michalewicz Z. Genetic algorithms, numerical optimization, and constraints //Proceedings of the Sixth International Conference on Genetic Algorithms. Morgan Kaufmann, San Mateo, CA, 1995. Т. 195. С. 151-158.

[90] Карпенко А.П., Селиверстов Е.Ю. Глобальная оптимизация методом роя частиц. Обзор // Информационные технологии, 2010, №2, с. 25-34.

[91] Карпенко А. П., Селиверстов Е. Ю. Глобальная безусловная оптимизации роем частиц на графических процессорах архитектуры CUDA //Наука и образование: электронное научно-техническое издание. 2010. №04. - С. 5.

[92] Гришин А. А., Карпенко А. П. Исследование эффективности метода пчелиного роя в задаче глобальной оптимизации //Наука и образование: электронное научно-техническое издание. 2010. №08. С. 5.

[93] Kennedy J. Particle swarm optimization //Encyclopedia of Machine Learning. Springer US, 2010. С. 760-766.

[94] Clerc M. Particle swarm optimization. John Wiley & Sons, 2010. Т. 93.

[95] Hansen, E.: Global Optimization Using Interval Analysis Dekker, New York (1992)

[96] Shary S.P. A surprising approach in interval global optimization, Reliable Computing. 2001. Vol. 7, No. 6. P. 497-505

[97] Tuy H. D(C)-optimization and robust global optimization // J. Glob. Optimization. 2010, V. 47, pp. 485-501.

[98] Strongin R. G., Sergeyev Ya. D., Global optimization with non-convex constraints: Sequential and parallel algorithms, Kluwer Academic Publishers, Dordrecht, 2000, 728 pp.

[99] Гергель В.П. Об одном способе учета значений производных при минимизации многоэкстремальных функции // Ж. вычисл. матем. и матем. физ. 1996. Т. 36 N 6. С. 51-67.

[100] Sergeyev Y. D., Strongin R. G., Lera D. Introduction to global optimization exploiting space-filling curves. Springer, 2013.

[101] Bulatov V. P. Methods for solving multi-extremal problems (global search) //Annals of Operations Research. 1990. Т. 25. №. 1. С. 253-277.

[102] Bulatov V. P., Khamisov O. V. The branch and bound method with cuts in En+ 1 for solving concave programing problem //System Modelling and Optimization. Springer Berlin Heidelberg, 1992. С. 273-281.

[103] Булатов В. П., Хамисов О. В. Метод отсечения в En+1 для решения задач глобальной оптимизации на одном классе функций //Журнал вычислительной математики и математической физики. 2007. Т. 47. №11. С. 1830-1842.

[104] Хамисов О. В. Глобальная оптимизация функций с вогнутой опорной минорантой //Журнал вычислительной математики и математической физики. 2004. Т. 44. №. 9. С. 1552-1563.

[105] Ершов А. Р., Хамисов О. В. Автоматическая глобальная оптимизация //Дискретный анализ и исследование операций. 2004. Т. 11. №. 2. С. 45-68.

[106] Хамисов О. В. Глубокие отсечения в вогнутом и линейном 0-1 программировании //Труды Института математики и механики УрО РАН. 2014. Т. 20. №. 2. С. 294-304.

[107] Ratchek H., Rokne J. New Computer Methods for Global Optimization. Chichester, Ellis Horwood, 1988.

[108] Leary, Robert H. and Doye, Jonathan P. K., Tetrahedral global minimum for the 98-atom Lennard-Jones cluster // Phys. Rev. E. Vol. 60. №6. 1999. P. 6320-6322.

[109] Andrea Grosso, Marco Locatelli, Fabio Schoen, A population-based approach for hard global optimization problems based on dissimilarity measures // Mathematical Programming. Vol. 110. 2007. P. 373-404.

[110] Evtushenko Yu.G., Potapov M.A., Korotkikh V.V. Numerical methods for global optimization //Recent advances in global optimization (Edited by C. Floudas and P. Pardalos), Princeton University Press. 1992. pp. 274-297.

[111] Evtushenko Y., Posypkin M., Sigal I. A framework for parallel large-scale global optimization // Computer Science - Research and Development 23(3). 2009. pp. 211-215.

[112] Посыпкин М.А. Решение задач глобальной оптимизации в среде распределенных вычислений // Программные продукты и системы. №1. 2010. С. 23-29.

[113] Посыпкин М.А. Методы и распределенная программная инфраструктура для численного решения задачи поиска молекулярных кластеров с минимальной энергией //Вестник нижегородского университета им. Н.И. Лобачевского №1. 2010. С. 210-219.

[114] Tawarmalani M., Sahinidis N. V. A polyhedral branch-and-cut approach to global optimization, Mathematical Programming, 103(2), 225-249, (2005)

[115] http://www.lindo.com/

[116] http://www.gams.com/

[117] Evans J.P., Gould F.J. Stability in Nonlinear Programming // Opns. Res. V. 18. 1970. pp. 107-118.

[118] Голиков А.И., Коткин Г.Г. Характеристика множества оптимальных оценок задачи многокритериальной оптимизации // ЖВМ и МФ, Т. 28, 1988, №10, С. 1461-1474.

[119] Тыртышников Е.Е. Матричный анализ и линейная алгебра. М.: Физматлит, 2007.

[120] Евтушенко Ю. Г., Посыпкин М. А. Варианты метода неравномерных покрытий для глобальной оптимизации частично-целочисленных нелинейных задач. // Доклады Академии Наук, 2011, Т. 437, №2, С. 168-172.

[121] Евтушенко Ю. Г., Посыпкин М. А. Применение метода неравномерных покрытий для глобальной оптимизации частично целочисленных нелинейных задач. // ЖВМиМФ, 2011, Т. 51, №8, С. 1376-1389.

[122] Evtushenko Y., Posypkin M. A Deterministic Algorithm for Global Optimization // Lecture Notes in Economics and Mathematical Systems, Vol. 658, ISBN 978-3-642-22883-4, P. 205-218, 2012.

[123] Evtushenko Y., Posypkin M. A deterministic approach to global box-constrained optimization // Optimization Letters, 2013, Volume 7, Issue 4, pp 819-829

[124] Штойер Р. Многокритериальная оптимизация. М.: Радио и связь, 1992.

[125] Лотов А.В., Поспелова И.И. Многокритериальные задачи принятия решений, М.: МАКС-Пресс, 2008,200с.

[126] Multi-objective Optimization - Interactive and Evolutionary Approaches, Springer LNCS 5252, Branke, J.; Deb, K.; Miettinen, K.; Slowinski, R. (Eds.), 470p. 2008.

[127] Zadeh L. Optimality and non-scalar-valued performance criteria //Automatic Control, IEEE Transactions on. - 1963. - Т. 8. - №. 1. - С. 59-60.

[128] Ehrgott M. Multicriteria Optimization. Springer. Berlin. 2005.

[129] Краснощеков П.С., Морозов В.В., Федоров В.В. Декомпозиция в задачах проектирования // Изв. АН СССР. Техн кибернетика. 1979. №2. С. 7 - 18

[130] Fliege J. Gap-free computation of Pareto-points by quadratic scalarizations // Mathematical Methods of Operations Research. V. 59. 2004. P. 69–89.

[131] Kim I.Y., de Weck O.L., Adaptive weighted sum method for multiobjective optimization: a new method for Pareto front generation // Struct Multidisc Optim. V. 31. 2006. P. 105–116.

[132] Eichfelder G. Scalarizations for adaptively solving multi-objective optimization problems // Computational Optimization and Applications. V.44. 2009, P. 249–273.

[133] Tan K.C., Khor E.F., Lee T.H. Multiobjective Evolutionary Algorithms and Applications. Springer-Verlag. London. 2005.

[134] Zitzler E., Laumanns M., Thiele L. Spea2: Improving the strength Pareto evolutionary algorithm. Technical Report 103. ETH Zurich. 2001.

[135] Карпенко А. П., Митина Е. В., Семенихин А. С. Популяционные методы аппроксимации множества Парето в задаче многокритериальной оптими-

зации //Наука и образование: электронное научнотехническое издание. 2012. Т. 4.

[136] Гладков Л. А., Курейчик В. В., Курейчик В. М. Генетические алгоритмы. М. : Физматлит, 2006.

[137] Жадан В.Г. Метод параметризации целевых функций в условной многокритериальной оптимизации // ЖВМиМФ. Т. 26. №2. 1986. С. 177–189.

[138] Жадан В.Г. Метод модифицированной функции Лагранжа для задач многокритериальной оптимизации // Т. 28. №11. 1988. С. 1603–1618.

[139] Антипин А. С., Голиков А. И., Хорошилова Е. В., Функция чувствительности, ее свойства и приложения // Ж. вычисл. матем. и матем. физ. Т. 51 №12. 2011. С. 2126-2142.

[140] Shao L., Ehrgott M. Approximately solving multiobjective linear programmes in objective space and an application in radiotherapy treatment planning // Math. Methods Oper. Res. Vol. 68. 2008. P. 257-276.

[141] Ehrgott M., Shao L., Schobel A. An approximation algorithm for convex multiobjective programming problems // Journal of Global Optimization. V. 50. 2011. P. 397–416.

[142] Поспелов А. И. Аппроксимация выпуклой оболочки Эджворта-Парето в многокритериальных целочисленных задачах с монотонными критериями //Журнал вычислительной математики и математической физики. 2009. Т. 49. №. 10. С. 1765-1778.

[143] Лазарев А. А. Парето-оптимальное множество NP-трудной задачи минимизации максимального временного смещения //Известия РАН. Теория и системы управления. 2006. №. 6. С. 103-110.

[144] Канторович Л.В., Акилов Г.П. Функциональный анализ. БХВ-Петербург. 2004.

[145] Bleuler S., Laumanns M., Thiele L., Zitzler E. PISAЧА Platform and Programming Language Independent Interface for Search Algorithms // In C. M. Fonseca et al., editors, Conference on Evolutionary Multi-Criterion Optimization (EMO 2003). V. 2632 of LNCS, P. 494-508. Berlin. Springer. 2003.

[146] Chafekar D., Xuan J., Rasheed K. Constrained Multiobjective Optimization Using Steady State Genetic Algorithms // In Proceedings of Genetic and Evolutionary Computation Conference, Chicago. Illinois. P. 813-824. 2003.

[147] Квасов Д. Е., Сергеев Я. Д. Методы липшицевой глобальной оптимизации в задачах управления //Автоматика и телемеханика. 2013. №. 9. С. 3-19.

[148] Калмыков С.А., Шокин Ю.И., Юлдашев З.Х. Методы интервального анализа. Новосибирск: Наука. 1986.

[149] Шарый С.П. Рандомизированные алгоритмы в интервальной глобальной оптимизации // Сибирский журнал вычислительной математики. 2008. Т. 11. №4. С. 457-474.

[150] Карпенко А.П., Семенихин А.С., Червяцова М.Н. Метод приближенного построения границы области достижимости многосекционного манипулятора типа “хобот”// Наука и образование: электронное научно- техническое издание. 2011. http://technomag.edu.ru/doc/165078.html

[151] Alekseeva E., Kochetov Y., Plyasunov A. Complexity of local search for the p-median problem //European Journal of Operational Research. 2008. Т. 191. №. 3. С. 736-752.

[152] Гимади Э.Х., Кельманов А.В., Пяткин А.В., Хачай М.Ю. Эффективные алгоритмы с оценками точности для некоторых задач поиска нескольких клик в полном неориентированном взвешенном графе //Труды Института математики и механики УрО РАН. 2014. Т. 20. №2. С. 99-112.

[153] Хачай М. Ю. Вычислительная сложность комбинаторных задач, индуцированных коллективными процедурами обучения распознаванию образов //Тр. ИММ УрО РАН, 16:3 (2010). 2010. С. 276-284.

[154] Сигал И.Х. Параметризация и исследование некоторых задач дискретного программирования большой размерности // Известия РАН. Теория и системы управления, 2001, №2, С. 60-69.

[155] Сигал И.Х. Задачи дискретного программирования большой размерности: параметризация и исследование ε-приближенных алгоритмов // Доклады Академии наук. 2003. Т. 391. №6. С. 741-745.

[156] Гришухин В.П. Эффективность метода ветвей и границ в задачах с булевыми переменными // Исследования по дискретной оптимизации. 1976. С. 203-230.

[157] Kellerer H., Pfershy U., PisingerD. Knapsack Problems. Springer Verlag, 2004.

[158] Martello S., Toth P., Knapsack Problems. John Wiley & Sons Ltd., 1990.

[159] Сигал И.Х., Иванова А.П. Введение в прикладное дискретное программирование: модели и вычислительные алгоритмы. изд. 2-е, исп. и доп. М.: ФИЗМАТЛИТ, 2007.

[160] Kolesar P.J. A branch and bound algorithm for the knapsack problem // Management Science. 1967. V. 13, N 9. P. 723–735.

[161] Greenberg H., Hegerich R.L. A branch and bound algorithm for the knapsack problem // Management Science. 1970. V. 16, N 5. P. 327-332.

[162] Воеводин В.В., Воеводин Вл.В. Параллельные вычисления. БХВ-Петербург, 2002. 608 стр.

[163] Гергель В.П. Теория и практика параллельных вычислений, учебное пособие. Москва, 2007.

[164] Стронгин Р.Г., Гергель В.П., Баркалов К.А. Параллельные методы решения задач глобальной оптимизации // Известия высших учебных заведений. Приборостроение. 2009. Т. 52. №10. С. 25-33.

[165] Grishagin V.A., Sergeyev Y.D., Strongin R.G. Parallel characteristical algorithms for solving problems of global optimization // Journal of Global Optimization. 1997. T. 10. №2. C. 185-206.

[166] Sergeyev Y.D. Parallel information algorithm with local tuning for solving multidimensional go problems // Journal of Global Optimization. 1999. T. 15. №2. C. 157-167.

[167] Онищенко Б.О. Решение одной задачи стохастической глобальной оптимизации параллельным методом ветвей и границ на кластере // Компьютерная математика. Вып. 3. Киев: Ин-т кибернетики им. В.М. Глушкова НАН Украины, 2005. С. 153-160

[168] H. Trienekens, A. Bruin, Towards a taxonomy of parallel branch and bound algorithms. Report EUR-CS-92-01, Erasmus Universisty. Rotterdam, 1992.

[169] B. Gendron, T.G. Grainic, Parallel branch-and-bound survey and synthesis // Operations Research. V. 42. №6. P. 1042-1066, 1994.

[170] K. Aida, W. Natsume, Y. Futakata, Distributed Computing with Hierarchical Master-worker Paradigm for Parallel Branch and Bound Algorithm, Proc. 3rd IEEE/ACM International Symposium on Cluster Computing and the Grid (CCGrid 2003), pp.156-163, 2003.

[171] Wu I. C., Kung H. T. Communication complexity for parallel divide-and-conquer //Foundations of Computer Science, 1991. Proceedings., 32nd Annual Symposium on. IEEE, 1991. C. 151-162.

[172] Bhatt S. et al. Tight bounds for on-line tree embeddings //SIAM Journal on Computing. 1999. Т. 29. №2. С. 474-491

[173] Афанасьев А. П., Волошинов, В. В., Рогов, С. В., Сухорослов, О. В. Развитие концепции распределенных вычислительных сред //Проблемы вычислений в распределенной среде: организация вычислений в глобальных сетях. Труды ИСА РАН.-М.: РОХОС. - 2004. - С. 6-105.

[174] Кудрявцев Л.Д. Курс математического анализа. М.: Высш. шк. 1981.

[175] Яблонский С.В. Введение в дискретную математику. М.: Наука,1986.

[176] Anikin A.S., Gornov A.Yu. Software implementation of Newton's method for Keating's potential optimization problems with use of sparse matrix technology // J. Studia Informatica Universalis. 2012. Vol. 9. №3. P. 11-19.

[177] Волошинов В.В., Неверов В.С. Использование распределенной среды REST-сервисов для оптимизационной идентификации углеродистых структур по рентгеновским дифракционным характеристикам наноматериалов // Труды XIII Российской конференции с участием иностранных ученых "Распределенные информационные и вычислительные ресурсы"(DICR'2010) Новосибирск, 30 ноября - 3 декабря 2010 г. 15 стр.

[178] Lotov A. V. et al. Optimal control of cooling process in continuous casting of steel using a visualization-based multi-criteria approach //Applied mathematical modelling. 2005. Т. 29. №7. С. 653-672.

[179] Албу А.В., Албу А.Ф., Зубов В.И. Управление процессом кристаллизации вещества в литейной форме сложной геометрии // Ж. вычисл. матем. и матем. физ. Т. 52 №12. 2012. С. 2149-2162.

[180] Гаранжа В. А., Голиков А. И., Евтушенко Ю. Г., Нгуен М. Х. Параллельная реализация метода Ньютона для решения больших задач линейного программирования // Журнал вычислительной математики и математической физики. Т. 49. №8. С. 1369-1384.

[181] Various optimization codes by D. Pisinger. URL: http://www.diku.dk/ pisinger/codes.html

[182] Голиков А. И., Евтушенко Ю. Г., Моллаверди Н. Применение метода Ньютона к решению задач линейного программирования большой размерности //Журнал вычислительной математики и математической физики. 2004. Т. 44. №9. С. 1564-1573.

[183] Jünger M., Thienel S. The ABACUS system for branch-and-cut-and-price algorithms in integer programming and combinatorial optimization // Softw., Pract. Exper. 2000. Т. 30. №11. С. 1325-1352.

[184] Böhm M. Parallel ABACUS-Introduction and Tutorial. 1999.

[185] George L. Nemhauser, Martin W.P. Savelsbergh, Gabriele C. Sigismondi. MINTO, a Mixed INTeger Optimizer. Operations Research Letters. 15. P. 47-58.

[186] IBM ILOG CPLEX Optimizer home page, URL: http://www-01.ibm.com/software/commerce/optimization/cplex-optimizer/index.html

[187] Linderoth J.T., Ralphs T.K. Noncommercial Software for Mixed-Integer Linear Programming, Integer Programming: Theory and Practice, John Karlof (ed.) CRC Press Operations Research Series. 2005. P. 253-303.

[188] Веселов В. Н., Евтушенко Ю. Г., Мазурик В. П. Содержательные возможности диалоговой системы оптимизации (ДИСО) //Проблемы вычислительной техники. М.: Международный центр науч. и техн. информации. 1981. С. 110-112.

[189] Жадан В. Г. Нелинейное программирование в системе ДИСО/ПК // Сообщ. по прогр. обеспечению ЭВМ. 1988.

[190] Жадан В. Г., Кушнирчук В. И. Пакет методов многокритериальной оптимизации в системе ДИСО //Пакеты прикладных программ. Программное обеспечение оптимизационных задач.М.: Наука. 1987. С. 17.

[191] Giunta A. A. et al. DAKOTA, a multilevel parallel object-oriented framework for design optimization, parameter estimation, uncertainty quantification, and sensitivity analysis: Version 4.1 reference manual. - Albuquerque, NM : Sandia National Laboratories, 2007.

[192] Программный комплекс Acro. URL: https://software.sandia.gov/trac/acro

[193] Eckstein J., Phillips C. A., Hart W. E. PICO: An object-oriented framework for parallel branch and bound //Studies in Computational Mathematics. 2001. Т. 8. С. 219-265.

[194] Ralphs T. K. Parallel branch and cut //Parallel Combinatorial Optimization. 2006. С. 53-101.

[195] Пакеты BiCePs и BLIS. URL: https://projects.coin-or.org/CHiPPS

[196] Alba E. et al. Efficient parallel LAN/WAN algorithms for optimization. The MALLBA project //Parallel Computing. - 2006. Т. 32. №. 5. С. 415-440.

[197] Eckstein J., Phillips C. A., Hart W. E. PEBBL 1.0 user guide. - RUTCOR Research Report RRR 19-2006.

[198] Patricia Hough, Tamara G. Kolda, and Virginia Torczon, Asynchronous Parallel Pattern Search for Nonlinear Optimization, SIAM J. Scientific Computing, V. 23. №1. P.134-156. June 2001.

[199] Menouer T., Le Cun B., Vander-Swalmen P. Partitioning Methods to Parallelize Constraint Programming Solver Using the Parallel Framework Bobpp //Advanced Computational Methods for Knowledge Engineering. Springer International Publishing, 2013. C. 117-127.

[200] Chen Q., Ferris M. C. Fatcop: A fault tolerant condor-pvm mixed integer programming solver //SIAM Journal on Optimization. 2001. T. 11. №4. C. 1019-1036.

[201] Litzkow M. J., Livny M., Mutka M. W. Condor-a hunter of idle workstations // Distributed Computing Systems, 1988., 8th International Conference on. IEEE, 1988. C. 104-111.

[202] Goux J. P. et al. Master-worker: an enabling framework for applications on the computational grid // Cluster Computing. 2001. T. 4. №1. C. 63-70.

[203] Anstreicher K. et al. Solving large quadratic assignment problems on computational grids // Mathematical Programming. 2002. T. 91. №3. C. 563-588.

[204] Aida K., Futakata Y., Osumi T. Parallel Branch and Bound Algorithm with the Hierarchical Master-Worker Paradigm on the Grid //Information and Media Technologies. 2007. T. 2. №1. C. 17-30.

[205] Nakada H., Tanaka Y., Matsuoka S., Sekiguchi S., Ninf-G: a GridRPC system on the Globus toolkit , Grid Computing: Making the Global Infrastructure a Reality,John Wiley & Sons Ltd , P. 625-638, 2003.

[206] Сигал И.Х. Дискретные модели и методы решения задач типа коммивояжера большой размерности: исследование, комбинированные алгоритмы, вычислительный эксперимент, применения. Диссертация на соискание ученой степени доктора технических наук. М.: 1990.

[207] Межведомственный суперкомпьютерный центр. URL: http://www.jscc.ru/hard/mvs100k.shtml

[208] Henning M. A New Approach to Object-Oriented Middleware // IEEE Internet Computing, Jan 2004.

[209] Смирнов С.А. Распределенный программный комплекс для моделирования структуры молекулярных соединений // Труды Третьей Международной научно-практической конференции "Современные информационные технологии и ИТ-образование" Москва, 6-9 декабря 2008 г. М.: МАКС ПРЕСС, 2008, С. 521-528.

[210] Елецкий А.В. Экзотические объекты атомной физики // Соросовский образовательный журнал. №4. 1995. С. 86-95.

[211] Wales D. J., Doye J. P. K., et al. The Cambridge Cluster Database. URL: http://www-wales.ch.cam.ac.uk/CCD.html.

[212] Northby J.A. Structure and binding of Lennard-Jones clusters: 13<= N <= 147 // Journal of Chemical Physics, Vol. 87(1987), P. 6166–6178.

[213] Longjiu Cheng, Jinlong Yang, Global Minimum Structures of Morse Clusters as a Function of the Range of the Potential: 81 <= N <= 160 // Journal of Physical Chemistry. 2007. 111. P. 5287-5293.

[214] Leary, R. H. 1997. Global Optima of Lennard-Jones Clusters. // Journal of Global Optimization 11, 1 (Jul. 1997), P. 35-53.

[215] Doye J. P., Wales D. J., Simdyankin S. I. Global optimization and the energy landscapes of Dzugutov clusters // Faraday Discuss. 2001. Vol. 118. P. 159-170.

[216] Costas D. Maranas, Christodoulos A. Floudas. A Global Optimization Approach for Lennard-Jones Microclusters // Journal of Chemical Physics. Vol. 97. 1992. P. 7667-7677.

[217] Посыпкин М. А., Параллельный эвристический алгоритм глобальной оптимизации // Труды ИСА РАН. Т. 32. 2008. С. 166-179.

ПРИЛОЖЕНИЕ 1. Применение системы BNB-Grid для решения задачи криптоанализа поточных шифров

Задача поиска набора, выполняющего произвольную конъюнктивную нормальную форму (КНФ), либо констатация факта отсутствия такового набора известна как SAT-задача. Несмотря на то, что SAT-задача в такой общей постановке NP-трудна, многочисленные ее частные случаи, возникающие в практических приложениях, допускают весьма эффективные процедуры решения посредством разнообразных эвристических алгоритмов. Разработка специализированных программных решателей SAT-задач выделилась за последние 10 лет в самостоятельное активно развивающееся направление. Среди создаваемых SAT-решателей регулярно проводятся конкурсы в Internet. Основную массу проблем, в отношении которых оправдан SAT-подход, составляют задачи верификации дискретных автоматов и моделей программ (model checking).

В данном приложение рассматривает использование SAT-подхода для решения задач криптоанализа некоторых систем поточного шифрования в распределенных вычислительных средах (PBC). Применение SAT-подхода к задачам криптоанализа известно как логический криптоанализ. Первой публикацией по данному направлению следует считать, по-видимому, работу (Massacci F., Marraro L. Logical Cryptoanalysis as a SAT Problem: the Encoding of the Data Encryption Standard. Preprint. Dipartimento di Imformatica e Sistemistica,

Universita di Roma YLa SapienzaY, 1999). Базовые концепции криптоанализа были независимым образом обоснованы в работе (Семенов А.А., Буранов Е.В. Погружение задачи криптоанализа симметричных шифров в пропозициональную логику // Вычислительные технологии (совместный выпуск с журналом кРегиональный вестник Востокаь), т.8, 2003, С. 118-126) Для решения задачи криптоанализа одного из наиболее интересных с практической точки зрения генераторов поточного шифрования и генератора A5/1 (Biryukov A., Shamir A., Wagner D. Real Time Cryptanalysis of A5/1 on a PC // Fast Software Encryption Workshop. 2000. pp. 1-18) вычислительных мощностей обычных кластеров оказалось недостаточно. Основным результатом настоящей работы является полный криптоанализ генератора ключевого потока A5/1, осуществленный в распределенной среде. Особый акцент мы делаем на том факте, что при решении данной задачи, в отличие от некоторых недавно анонсированных исследований, не использовались специальные вычислители на ПЛИС и дополнительные накопители данных. Также подчеркнем, что главная мотивация предпринятых исследований состояла не столько в желании осуществить криптоанализ реально используемой системы шифрования, сколько в желании аргументировать эффективность предлагаемой методики решения SAT-задач в распределенных вычислительных средах.

Пусть $X = \{x_1, \ldots, x_n\}$ — множество, образованное булевыми переменными и пусть $L(x_1, \ldots, x_n)$ — произвольная формула исчисления высказываний, содержащая переменные из множества X. Подстановка в $L(x_1, \ldots, x_n)$ произвольного вектора значений переменных $(\alpha_1, \ldots, \alpha_n)$, $\alpha_i \in \{0, 1\}$, $i = 1, \ldots, n$, дает в результате либо 0, либо 1. Данный факт будем записывать как $L(\alpha_1, \ldots, \alpha_n) = \beta, \beta \in \{0, 1\}$. Выражения вида

$$L(x_1, \ldots, x_n) = 0, L(x_1, \ldots, x_n) = 1,$$

называются логическими уравнениями. Решением логического уравнения

$L(x_1, \ldots, x_n) = \beta$ называется такой вектор $(\alpha_1, \ldots, \alpha_n), \alpha_i \in \{0,1\}, i \in \{1, \ldots, n\}$, что $L(\alpha_1, \ldots, \alpha_n) = \beta$. называются логическими уравнениями. Важнейший класс логических уравнений образован уравнениями вида

$$C(x_1, \ldots, x_n) = 1, \tag{6.3}$$

где $C(x_1, \ldots, x_n)$ — конъюнктивная нормальная форма (КНФ). Если уравнение (6.3) имеет решения, то КНФ $C(x_1, \ldots, x_n)$ называется выполнимой, а решения (6.3) называются выполняющими данную КНФ наборами. В противном случае $C(x_1, \ldots, x_n)$ называется невыполнимой. К SAT-задачам относятся задачи определения выполнимости произвольных КНФ и поиска наборов, выполняющих выполнимые КНФ.

Обозначим через $f = \{f_n\}$, $n \in \mathbb{N}$ семейство дискретных функций вида $f_n : \{0,1\}^n\{0,1\}^*$, определенных всюду на $\{0,1\}^n$ и алгоритмически вычислимых за полиномиальное от n время. Проблемой обращения произвольной функции f_n из такого семейства называется следующая задача: по произвольному $y \in \{0,1\}^*$ и известному алгоритму вычисления $y \in \mathrm{range} f_n$ (программе для выбранной вычислительной модели) требуется найти такой $x \in \{0,1\}^n$, что $f_n(x) = y$. Данную проблему будем называть проблемой обращения функции f_n в точке y.

Перенося известную идею С.А. Кука о пропозициональном кодировании алгоритмов на проблему обращения функций из рассматриваемого класса, можно показать справедливость следующего факта.

Теорема 6.1 *Для любого семейства f дискретных функций из определенного выше класса существует алгоритм с полиномиально от n ограниченной сложностью, который, получая на входе n и $y \in range f_n$, преобразует проблему обращения f_n в точке y в проблему поиска решений логического уравнения вида $C(x_1, \ldots, x_{q(n)}) = 1$, где $q(n)$ — некоторый полином, а $C(x_1, \ldots, x_{q(n)}) = 1$*

— выполнимая КНФ над множеством булевых переменных $x_1, \ldots, x_{q(n)}$.

Детальное доказательство данного факта в контексте бинарных машин с неограниченными регистрами (МНР) приведено в (Семенов А.А. Трансляция алгоритмов вычисления дискретных функций в выражения пропозициональной логики // Прикладные алгоритмы в дискретном анализе. Серия: Дискретный анализ и информатика, Вып. 2. 2008. Иркутск: Изд-во ИГУ. С. 70-98.). Там же подробно описана концепция транслятора алгоритмов вычисления дискретных функций из рассматриваемого класса в SAT-задачи. Программная реализация данного транслятора дала возможность рассматривать задачи криптоанализа некоторых систем шифрования в форме SAT-задач. Наиболее успешным этот подход оказался в отношении генераторов ключевого потока, используемых в поточном шифровании.

Генератором ключевого потока (генератором) называется такое натуральное семейство функций вида $f_{k,m} : \{0,1\}^k \to \{0,1\}^*$, $m \in \mathbb{N}$, что для произвольного $x \in \{0,1\}^k$ и произвольного $m \in \mathbb{N}$ существует $y \in \{0,1\}^m \cap \mathrm{range} f_{k,m}(x)$, и данное значение может быть вычислено детерминированным алгоритмом $A(f_{k,m})$ с полиномиальной от m трудоемкостью. При этом вектор x называется инициализирующей последовательностью длины k, а вектор y — фрагментом ключевого потока длины m. Задача криптоанализа генератора заключается в восстановлении по некоторому вектору $y \in \mathrm{range} f_{k,m}$ (фрагменту ключевого потока) и известному алгоритму $A(f_{k,m})$ инициализирующей последовательности $x \in \{0,1\}^k : f_{k,m}(x) = y$. Обратим внимание на тот факт, что генераторы, строго говоря, не относятся к классу дискретных функций, определенному выше: параметр k — длина инициализирующей последовательности является константой. Однако идея трансляции алгоритмов вычисления функций в КНФ к генераторам полностью применима.

Технология логического криптоанализа основана она на следующем простом

факте. Пусть $C(x_1, \ldots, x_{q(n)})$ — КНФ, кодирующая (в смысле теоремы 6.1) задачу обращения функции f_n из определенного выше класса в точке $y \in \mathrm{range} f_n$. Переменные из множества $X = \{x_1, \ldots, x_n\}$ называем переменными входа функции f_n (если функцию реализовать схемой из функциональных элементов в произвольном полном базисе, то входные полюса данной схемы будут помечены переменными из множества X). Все переменные в $x_1, \ldots, x_{q(n)} \setminus X$ функционально зависят от переменных из X, и поэтому любая подстановка значений всех переменных из X индуцирует вывод остальных переменных из $C(x_1, \ldots, x_{q(n)})$ по правилу единичного дизъюнкта. Данный факт строго доказан в (Семенов А.А., Отпущенников И.В. Об алгоритмах обращения дискретных функций из одного класса // Прикладные алгоритмы в дискретном анализе. Серия: Дискретный анализ и информатика, Вып. 2. Иркутск: Изд-во ИГУ. 2008.С. 127-156)). Данный факт дает основу для построения декомпозиции исходной SAT-задачи на семейство, состоящее из 2^d, $1 \leq d \leq n$ SAT-задач для КНФ, построенных следующим образом: среди переменных входа выбираются некоторые d переменных $x_{i_1}, \ldots, x_{i_d}$, после чего рассматривается семейство, образованное КНФ вида

$$C(x_1, \ldots, x_{q(n)}) : x_{i_1} = \alpha_{i_1}, \ldots, x_{i_d} = \alpha_{i_d},$$

причем вектор $(\alpha_{i_1}, \ldots, \alpha_{i_d})$ пробегает все множество $\{0,1\}^d$. Полученное семейство КНФ обозначается через Δ_d и называется *декомпозиционным семейством*, а множество $x_{i_1}, \ldots, x_{i_d}$ — *декомпозиционным множеством*. Поскольку SAT-задачи, соответствующие декомпозиционному множеству можно решать независимо, для их решения можно использовать распределенную вычислительную систему (РВС).

Вопрос выбора декомпозиционного множества далеко не всегда является тривиальным: если величина d будет большой, то в декомпозиционном семействе окажутся простые КНФ, однако их количество будет слишком велико. При ма-

лых значениях параметра d КНФ в декомпозиционном семействе, как правило, слишком сложны для отдельных вычислительных узлов. Для решения проблемы наилучшего (с точки зрения общей трудоемкости вычислительного процесса) выбора значения параметра было предложено использовать специальные прогнозные функции. Прогнозные функции позволяют делать оценочные предположения относительно общей трудоемкости процесса решения SAT-задач в РВС на основе обработки случайных выборок КНФ из декомпозиционных семейств, построенных для различных значений параметра d.

Прогнозная функция имеет следующий вид:

$$T(\Theta_d) = \begin{cases} \frac{2^d}{q_d}\tau_s(\Theta_d), 2^d > R_0, \tau_s(\Theta_d) < g(C); \\ \tau_s(\Theta_d), 2^d \leq R_0, \tau_s(\Theta_d) < g(C); \\ \infty, \tau_s(\Theta_d) \geq g(C). \end{cases}$$

Здесь через Θ_d обозначена случайная выборка, состоящая из q_d КНФ декомпозиционного семейства Δ_d . Через $\tau_s(\Theta_d)$ обозначено суммарное время работы фиксированного SAT-решателя s на всех КНФ выборки Θ_d . Число R_0 — некоторая константа, разделяющая ситуации: когда требуется формировать случайную выборку, а когда нет (данная величина определяется числом вычислительных узлов распределенной вычислительной системы). Функция $g(C)$ — некоторый полином от длины кодировки исходной КНФ, определяющий границы кразумного временинь проведения процедуры прогнозирования. Глобальный минимум функции $T(\cdot)$ на множестве выборок, получаемом в результате изменения параметра d на множестве $\{1, \ldots, n\}$, можно рассматривать как прогноз наилучших (с позиции трудоемкости решения) параметров декомпозиции исходной SAT-задачи.

Рассмотрим приведенную схему на примере SAT-задачи, кодирующей криптоанализ генератора A5/1. Данный генератор интересен, прежде всего, тем, что

он используется в действующем протоколе шифрования трафика в сетях сотовой телефонии стандарта GSM. В литературе описано довольно много атак на данный генератор, однако нам не удалось обнаружить убедительные результаты его криптоанализа в форме программной реализации и серий тестов. Относительно недавно был анонсирован криптоанализ A5/1 с использованием специальных вычислителей на ПЛИС, однако и в этом случае публикаций, в которых были бы приведены результаты успешных численных экспериментов (в форме серий тестов), на текущий момент нет. В описании генератора A5/1 мы следуем статье (Biryukov A., Shamir A., Wagner D. Real Time Cryptanalysis of A5/1 on a PC // Fast Software Encryption Workshop. 2000. pp. 1-18).

Проблема построения оптимальной декомпозиции для решения задачи криптоанализа A5/1 в форме SAT-задачи оказалась нетривиальной. Были исследованы различные стратегии формирования декомпозиционных множеств. Первые схемы основывались на идеях, ранее успешно примененных в параллельном криптоанализе ряда генераторов (пороговый, суммирующий, Гиффорда). При этом, исходя из некоторых кразумныхнь предположений, сначала строилось базовое декомпозиционное множество, а затем предпринимались попытки его усечения на основе значений соответствующих прогнозных функций. В соответствии с данной схемой предлагается включить в переменные, кодирующие начальные состояния ячеек регистров, начиная с первых ячеек, до ячеек, содержащих серединные биты. Данный выбор продиктован следующими естественными соображениями. Произвольная фиксация битов из построенного таким образом множества индуцирует точные значения серединных битов для значительного числа последующих состояний всех трех регистров. Но серединные биты являются наиболее информативными битами, поскольку именно они определяют, какая ветвь соответствующего условия (значение функции большинства) имеет место.

Довольно неожиданным оказался тот факт, что построенное таким образом множество не удалось уменьшить за счет технологии прогнозных функций. В проводимых статистических экспериментах для каждого варианта декомпозиционного множества и начального фрагмента ключевого потока некоторой фиксированной длины (от 128 до 256 бит) строились случайные выборки объемом 1000 КНФ. Для каждой такой выборки считалось значение прогнозной функции Ц среднее значение времени решения SAT-задач по выборке, умноженное на мощность пространства перебора. Прогнозы по схемам формирования, отличным от схемы (2), оказались хуже. Во всех экспериментах использовался SAT-решатель dminisat, представляющий собой модификацию известного решателя minisat. Подробное описание dminisat приведено ниже.

На основании большого числа экспериментов по прогнозированию параметров наилучшей в смысле общей трудоемкости декомпозиции было принято решение остановиться на декомпозиции, определяемой естественным декомпозиционным множеством. Тем самым возникла проблема организации в РВС вычислительной процедуры, в ходе которой требуется решить (в худшем случае) $2^3 1$ SAT-задач.

Во всех вычислительных экспериментах по логическому криптоанализу в РВС генератора A5/1 использовался решатель dminisat. Данный решатель является модификацией SAT-решателя minisat (http://minisat.se/MiniSat.html) и ориентирован на решение SAT-задач из декомпозиционного семейства, полученного в результате применения к КНФ, кодирующей работу A5/1, описанной ранее декомпозиции.

Модифицировалась версия minisat-Cv1.14.1. Основной этап модификации заключался в незначительном изменении процедуры выбора уровней решения, реализованной в minisat. А именно, была добавлена процедура присвоения начальной активности (отличной от нулевой) для 64 переменных, кодирующих ис-

ходное заполнение регистров генератора. Данный факт позволил на начальной стадии процесса решения выделить 64 переменные, соответствующие инициализирующей последовательности, в качестве приоритетных при выборе очередной переменной уровня решения. Это простое изменение привело к значительному росту производительности SAT-решателя на рассматриваемых КНФ.

Кроме того, были изменены некоторые базовые константы решателя. Подобно большинству известных SAT-решателей, minisat периодически понижает активность всех переменных и дизъюнктов с целью увеличения приоритета в угадывании для переменных, повышающих свою активность на более поздних шагах поиска. Кроме этого, в 2% случаев minisat присваивает значение переменной, выбранной случайным образом, а не переменной, обладающей максимальной активностью. Эти эвристики в среднем показывают хорошие результаты на широком наборе тестовых примеров, используемых в конкурсах SAT-решателей. Однако для КНФ, кодирующих шифр A5/1, они оказываются неэффективными. Запрещение периодического понижения активности и случайного выбора переменных дает увеличение скорости работы модифицированного в описанном выше смысле решателя еще на 20-30%.

Еще один аспект модификации SAT-решателя обусловлен необходимостью решения в PBC большого числа однотипных задач (SAT-задач из декомпозиционного семейства). Передача по сети отдельных SAT-задач из декомпозиционного семейства снижает общую эффективность вычислительного процесса в PBC. Для решения данной проблемы была организована передача на вычислительные узлы PBC кпакетов заданийь. Если для простоты предположить, что PBC состоит из 2^k узлов ($k < 31$), то на каждом узле можно организовать решение 2^{31-k} SAT-задач для КНФ из декомпозиционного семейства. Тем самым пакет заданий для каждого узла может быть кописанъ двоичным вектором длины (всего имеем различных пакетов заданий). Получая конкретный двоич-

Таблица 6.12: Сравнительные характеристики процессоров Intel E8400 и Intel E5472

Название процессора	Intel E8400	Intel E5472
Число ядер	2	4
Тактовая частота	3.0 ГГц	3.0 ГГц
Частота шины	1333 МГц	1600 МГц
Кэш L2	6 Мб	12 Мб

ный вектор длины , решатель на произвольном узле самостоятельно порождает векторов длины 31. Первые компонент этих векторов образуют вектор , оставшиеся компоненты изменяются произвольным образом. Описанная процедура генерации пакетов заданий на рабочих узлах РВС была встроена в dminisat.

Далее мы приводим прогноз времени решения задачи криптоанализа генератора A5/1 в рамках описанной технологии (с использованием решателя dminisat) на вычислительном кластере СКИФ-МГУ кЧебышевь (http://parallel.ru/cluster/). Данный кластер состоит из 1250 четырехядерных процессоров E5472. Все численные эксперименты по прогнозированию проводились на процессоре Intel E8400. В Табл. 6.6 приведены сравнительные характеристики данного процессора и процессора Intel E5472.

Из таблицы видно, что мощность одного ядра процессора Intel E8400 сопоставима с мощностью одного ядра процессора Intel E5472 (незначительное отличие лишь в частоте шины). Исходя из этого факта, мы сочли возможным напрямую экстраполировать результаты проведенных численных экспериментов применительно к процессу решения задачи логического криптоанализа генератора A5/1 на кластере СКИФ-МГУ “Чебышев”. Итоговый прогноз времени логического криптоанализа A5/1 на данном кластере составил 19-21 час.

Прогноз времени решения данной задачи на суперкомпьютере СКИФ-МГУ показывает, что даже при полной загрузке этого кластера, решение задачи криптоанализа требует значительного времени. Монопольное использование многопроцессорных вычислительных комплексов, находящихся в коллективном доступе, невозможно. Вместе с тем, в распоряжении исследователей зачастую име-

ются ресурсы различных кластеров, грид-систем, высокопроизводительных серверов. Программный комплекс BNB-Grid позволяет эффективно задействовать подобные разнородные распределенные вычислительные ресурсы для решения сложных вычислительных задач.

Структура приведенного выше вычислительного алгоритма для решения задачи криптоанализа шифра A5/1 допускает эффективную реализацию для параллельных и распределенных систем, так как подзадачи из декомпозиционного семейства могут обрабатываться независимым образом. Поэтому, учитывая высокую вычислительную сложность решения рассматриваемой задачи, было принято решение применить систему BNB-Grid для решения данной задачи, задействовав вычислительные ресурсы нескольких многопроцессорных вычислительных комплексов.

В библиотеку BNB-Solver был добавлен модуль, решающий набор подзадач SAT на параллельной вычислительной системе с распределенной памятью. Этот модуль работает в соответствии с известной парадигмой куправляющий-рабочиень: один из процессов (управляющий) распределяет подзадачи по рабочим процессам. Этот модуль используется на многопроцессорных вычислительных узлах, входящих в состав распределенной системы.

Система BNB-Grid загружает на управляющий модуль описание задачи в формате XML, в состав которого входит набор КНФ из декомпозиционного семейства. Управляющий модуль в дальнейшем распределяет эти КНФ по различным вычислительным узлам. Параллельному приложению передается порция задач, существенно превосходящая количество рабочих процессов, это нужно для оптимизации загрузки процессоров и максимального снижения простоя. Эффективность работы параллельного приложения тем выше, чем больше число переданных подзадач, так как в этом случае нивелируются негативные эффекты, связанные с разбалансировкой нагрузки. Однако, при слишком большом

Таблица 6.13: Характеристики многопроцессорных вычислительных комплексов, участвующих в вычислениях

Название	Ведомственная принадлежность	Архитектура процессора	Число вычислительных ядер
MVS-100k	МСЦ РАН	Xeon E5450 3 GHz	7920
СКИФ-МГУ	МГУ	Xeon E5472 3 GHz	5000
Кластер РНЦ	РНЦ кКурчатовский институтњ	Xeon 5345 2.333 GHz	3456
BlueGene/P	МГУ	Power PC 850 MHz	8192

числе подзадач требуемое время работы параллельного приложения сильно увеличивается. Это является нежелательным из-за ограничений систем пакетной обработки, установленных на кластерах: параллельное приложение, запрашиваемое время работы которого велико, может быть не запущено или будет простаивать в очереди очень длительное время. Экспериментально было определено компромиссное число КНФ в порции задач, обеспечивающее приемлемые показатели эффективности и, вместе с тем, не приводящее к чрезмерно длительному ожиданию в очереди. Это число составляет $4p$, где p — число процессов параллельного приложения. Расчеты останавливаются, когда хотя бы на одном из вычислителей найден выполняющий набор. Это набор передается на управляющий модуль BNB-Grid и сохраняется пользователем с помощью графического интерфейса.

Для расчетов были выбраны три тестовых примера для генератора A5/1 с длиной инициализирующей последовательности 64 бита. Для каждой из полученных SAT-задач было сгенерировано декомпозиционное семейство, содержащее 2^18 наборов. Расчеты осуществлялись на четырех многопроцессорных вычислительных комплексах, характеристики которых представлены в Табл. 6.13, взятых из 11-й редакции списка самых мощных суперкомпьютеров стран СНГ.

В результате расчетов, на проведение которых потребовалось несколько дней, были решены все тестовые задачи криптоанализа A5/1. На каждом кластере на выполнение отправлялось по четыре приложения BNB-Solver. При этом из-за

загруженности кластеров одновременно выполнялась только часть приложений. Количество одновременно работающих вычислительных ядер изменялось в процессе расчетов от 0 до 5568, составляя в среднем приблизительно 2-3 тысячи. Выполняющий набор (инициализирующая последовательность генератора) для первого теста был найден за 56 часов, для второго Ц за 25 часов, для третьего теста потребовалось 122 часа непрерывных расчетов. Время нахождения выполняющего набора в основном определяется следующими факторами: в каком сегменте декомпозиционного множества он расположен и загруженностью вычислителей.

Результаты проведенных вычислительных экспериментов показывают, что при условии вовлечения достаточной вычислительной мощности, удается выполнить криптоанализ генератора A5/1 в течение нескольких дней. Данный результат показывает, с одной стороны, потенциальную применимость предложенного подхода для решения задач криптоанализа, с другой, аргументирует использование параллельных вычислительных технологий в решении SAT-задач, возникающих в практических приложениях. Высокая степень параллелизма использованного алгоритма позволяет надеяться на существенное снижение времени его работы с увеличением числа вычислительных ядер.