Федеральное государственное бюджетное образовательное учреждение
высшего образования
«Сибирский государственный университет науки и технологий
имени академика М.Ф. Решетнева»

На правах рукописи

Масич Игорь Сергеевич

МЕТОД ОПТИМАЛЬНЫХ ЛОГИЧЕСКИХ РЕШАЮЩИХ ПРАВИЛ ДЛЯ КЛАССИФИКАЦИИ ОБЪЕКТОВ

05.13.01 – системный анализ, управление и обработка информации
(космические и информационные технологии)

ДИССЕРТАЦИЯ
на соискание ученой степени доктора технических наук

Научные консультанты:

доктор технических наук, профессор
Антамошкин Александр Николаевич

доктор технических наук, доцент
Казаковцев Лев Александрович

Красноярск – 2019

Оглавление

ВВЕДЕНИЕ

Актуальность работы. К настоящему времени в области классификации, кластеризации и распознавания сформировался ряд довольно эффективных методов, в том смысле, что они могут решать задачи распознавания с высокой точностью. Но при решении реальных задач распознавания и прогнозирования часто возникают вопросы, связанные с интерпретируемостью получаемых результатов и обоснованностью предлагаемых решений. Это, прежде всего, задачи, в которых могут быть велики негативные последствия принятия неверного решения (прогноза). И системы поддержки принятия решений, используемые для таких задач, должны обеспечивать возможность обосновывать решения и интерпретировать результат.

Такими задачами являются, в частности, задачи медицинской диагностики и прогнозирования. К примеру, при прогнозировании осложнений инфаркта миокарда есть необходимость в логических решающих правилах, имеющих подтверждение на опытных данных и способных обосновать прогнозируемое решение для новых пациентов.

Логические закономерности могут иметь широкое применение в задачах анализа и моделирования, в частности, при использовании лингвистических шаблонов для извлечения информации о событиях из открытых источников (Дмитриев М.Г.).

Требования доказательности и интерпретируемости предъявляются также к средствам, предназначенным для поддержки принятия решений при проведении испытаний, классификации и отборе электрорадиоизделий (ЭРИ) для комплектации радиоэлектронной аппаратуры космического применения. Одной из ключевых задач, возникающих при отборе изделий микроэлектроники, является задача классификации ЭРИ по однородным партиям.

Испытанием и отбором ЭРИ для комплектации РЭА КА занимаются специализированные предприятия. Для предотвращения попадания в аппаратуру потенциально ненадежных элементов, ЭРИ подвергаются дополнительным

отбраковочным испытаниям (ДОИ), а часть из них – разрушающему физическому анализу (РФА). В то же время, как указано в работах Орлова В.И. и Федосова В.В., задача отбора изделий, относимых к наивысшему классу пригодности, не может быть решена исключительно ужесточением требований по отдельным параметрам.

Поступающие от производителя партии ЭРИ часто являются неоднородными, вследствие изменений в производственном процессе, в частности из-за того, что могут быть изготовлены из разных полупроводниковых пластин. Такие изменения влияют на характеристики изделий, на их сопротивляемость воздействиям факторов космического пространства, прежде всего, воздействию радиации. Распространять результаты РФА на всю группу изделий можно только при условии, что эти изделия изготовлены из одной партии платин, являются однородными (Орлов В.И. и др.).

Выявление однородных партий ЭРИ, применяемых в узлах космической электроники, является одной из важных задач на пути повышения качества этих узлов и, как следствие, срока активного существования и надежности космической техники. Повышение качества достигается как за счет более согласованной работы радиоэлементов с идентичными характеристиками, так и за счет повышения качества и достоверности результатов разрушающих тестовых испытаний, для которых появляется возможность гарантированно отбирать элементы из каждой производственной партии.

Задачу выявления однородных партий можно рассматривать как задачу автоматической группировки объектов или задачу размещения (Казаковцев Л.А.). Решение такого типа задач исследовано в работах Береснева В.Л., Гимади Э.Х., Колоколова А.А., Кельманова А.В., Кочетова Ю.А., Забудского Г.Г. и др. Применение алгоритмов автоматической группировки и размещения на практических задачах рассмотрено в работах Дулесова А.С., Прутовых М.А. и др.

Но внедрение в процесс испытания алгоритмов группировки, использующих всё множество признаков и представляющих собой «черный ящик», сопряжено с трудностями. Для включения этапа классификации ЭРИ в

программу испытаний требуется способ проверки соответствия изделия классу с помощью логических решающих правил, опирающихся на четкие значения допустимых интервалов признаков. В условиях космического производства от метода классификации требуется не только точность решения задачи, но одновременно с этим доказательность применяемого подхода и интерпретируемость формулируемого им решения, что в данном случае означает возможность разработки ужесточенных норм параметров изделий.

Интерпретируемость означает возможность записать правило классификации в виде инструкции, понятной человеку. Под доказательностью понимается наличие веских объективных аргументов, подтверждающих предлагаемое системой решение. Более формально, E. Boros, Y. Crama, P.L. Hammer и др. называют классификатор доказательным по отношению к некоторой обучающей выборке, если каждое правило подтверждено (не менее чем одним) наблюдением и не противоречит ни одному наблюдению, и, кроме того, никакое правило не может быть заменено другим обоснованным правилом, которое имеет большую поддержку в данной выборке.

Направленность на доказательность смещает акцент в развитии подходов к обучению: основная цель уже не в том, чтобы получить высокое отношение правильно распознанных наблюдений, а в том, чтобы предоставить убедительные обоснования для каждой отдельной классификации. Другими словами, нас интересует априорное обоснование правил, а не их апостериорная эффективность. Один из подходов к априорному обоснованию состоит в применении алгоритмов определения информационной энтропии (Дулесов А.С.).

Наиболее интересными с этой точки зрения являются алгоритмы построения решающих правил с использованием аппарата алгебры логики. Известными подходами к распознаванию такого типа являются алгоритмы КОРА (Бонгард М.М., Вайнцвайг М.Н.), ТЭМП (Лбов Г.С.), предназначенные для определения логических закономерностей в виде конъюнкций значений признаков, метод «тупиковых тестов» (Чегис И.А., Яблонский С.В.), а также алгоритмы вычисления оценок, совмещающие метрические и логические

принципы классификации (Журавлёв Ю.И.). Решающее правило в этих случаях задаётся в виде алгоритмической процедуры – для распознавания наблюдений используется голосование по конъюнкциям или по «тупиковым тестам». Для выявления закономерностей алгоритм КОРА использует перебор всевозможных конъюнкций с числом литералов (степенью) не более некоторого заданного числа. Эти подходы получили развитие в работах Загоруйко Н.Г., Журавлева Ю.И., Дюковой Е.В., Рудакова К.В., Воронцова К.В.

Рабочей альтернативой являются алгоритмы стратегии «отделяй и властвуй» («separate-and-conquer» или по-другому «стратегии покрытия»), которая берёт начало в семействе алгоритмов AQ (Algorithm for synthesis of quasi-minimal covers, Michalski) и заключается в последовательном исключении покрытых найденной закономерностью обучающих наблюдений и поиске следующей закономерности на оставшихся наблюдениях. Основной упор в этих алгоритмах делается на быстрое построение классификатора (с помощью, как правило, жадной эвристической процедуры), но не на выявление множества наиболее информативных закономерностей, так как все закономерности, за исключением первой, строятся лишь на оставшейся части обучающих наблюдений.

Одним из наиболее основательных подходов к выявлению и использованию логических закономерностей представляется логический анализ данных (Hammer P.L.). Первоначально в данном подходе для выявления закономерностей использовались две техники перебора: снизу вверх (поиск допустимых закономерностей путём добавления литералов) и сверху вниз (исключение литералов для повышения покрытия), которые дополняли друг друга, что является очень трудоёмкой процедурой. В дальнейшем задача выявления закономерностей была сформулирована как задача условной псевдобулевой оптимизации: поиск закономерности с наибольшим покрытием при условии недопустимости (или ограниченной допустимости) покрытия обучающих наблюдений другого класса. Для решения этой задачи использовался жадный алгоритм (для нахождения приближенного решения) либо линейная аппроксимация целевой псевдобулевой

функции со сведением к задаче целочисленного линейного программирования, решение которой является приближенным решением исходной задачи.

В данном исследовании показано, что такой подход не обеспечивает нахождения оптимальных закономерностей, а именно закономерностей, удовлетворяющих критерию доказательности («сильных» закономерностей), которые являются наиболее привлекательными для повышения точности и интерпретируемости результата распознавания. Но, кроме этого, к получаемым закономерностям могут предъявляться дополнительные требования – удовлетворение условиям простоты или избирательности – в результате получаемые закономерности будут обладать своими особенностями. Этим обусловлена необходимость усовершенствования методов решения задачи выявления и использования логических решающих правил для распознавания.

Идея настоящей диссертации состоит в разработке метода поиска оптимальных закономерностей различных типов – сильных первичных и сильных охватывающих закономерностей – и их совместном использовании для построения логических решающих правил.

Степень проработанности проблемы. Методология анализа данных, состоящая в комбинации идей из оптимизации, комбинаторики и булевых функций, была впервые предложена Питером Хаммером (P. Hammer) в 1986 г. и названа логическим анализом данных. Затем этот подход был развит в работах Crama Y., Ibaraki T., Bores E., Kogan A., Alexe G., Alexe S., Bonates T., Anthony M. и др. Обзор результатов приведен в работе Чикалова И.В. и др. (2013). Подход получил широкое практическое применение, описанное в работах Caserta, Reiners (2016), Lejeune, Lozin, Lozina и др. (2019), Bruni, Bianchi, Dolente, Leporelli (2019), Bain, Avila-Herrera, Subasi (2018), Kim, Choi (2015), Yan, Ryoo (2017), Shaban, Meshreki, Yacout и др. (2017), Ragab, Koujok, Ghezzaz и др. (2019). Тем не менее, оставались открытыми ключевые вопросы, касающиеся оптимального назначения порогов при бинаризации количественных признаков, поиска оптимальных закономерностей различных типов, выбора типа закономерностей для решения

практических задач, анализа большого количества данных, повышения компактности классификатора.

Объектом диссертационного исследования являются задачи классификации, результат решения которых требует обоснования и доказательности, **предмет исследования** – методы и алгоритмы их решения.

Цель диссертационного исследования состоит в повышении точности решения задач классификации с требованием обоснования результатов распознавания и интерпретации в виде логических правил.

Поставленная цель достигается путем решения следующих **задач:**

- сравнительный анализ известных алгоритмов выявления закономерностей в данных и их использования для решения задач классификации;

- разработка новых способов выявления закономерностей на основе моделей оптимизации, построение моделей для нахождения закономерностей, удовлетворяющих различным критериям (простоты, доказательности, избирательности);

- сравнительный анализ применения разных типов закономерностей и разработка способов совместного использования закономерностей различных типов для повышения качества распознавания;

- построение единой модели оптимизации на основе метода логического анализа данных для поиска пары закономерностей (первичной и охватывающей);

- сведение задачи выявления закономерностей, оптимальных по критериям простоты, доказательности и избирательности, к задачам комбинаторной оптимизации и разработка алгоритмов их решения;

- разработка алгоритма поиска пары закономерностей (сильной первичной и сильной охватывающей) с использованием алгоритмов оптимизации;

- анализ существующих способов бинаризации вещественных признаков для применения метода оптимальных логических решающих правил и разработка способа оптимального назначения порогов вещественных признаков для их дискретизации, при котором наблюдения разных классов оказываются наиболее различимы, а число самих порогов ограничено;

- разработка процедуры ускорения поиска закономерностей, позволяющей применять метод оптимальных логических решающих правил для случаев большого объема данных;

- разработка способа отбора закономерностей в методе оптимальных логических решающих правил для повышения интерпретируемости классификатора без потерь в точности распознавания;

- апробация метода оптимальных логических решающих правил на решении практических задач из разных областей.

Методы исследования. Основные теоретические и прикладные результаты получены с применением методов системного анализа, исследования операций, теории оптимизации, теории множеств.

Новые научные результаты и положения, выносимые на защиту:

1. Впервые предложен метод нахождения сильных первичных и сильных охватывающих закономерностей на основе решения задачи комбинаторной оптимизации – метод оптимальных логических решающих правил. Метод состоит в использовании новых моделей оптимизации и алгоритмов поиска оптимальных решений, основанных на использовании свойств классов задач псевдобулевой оптимизации и принципах схемы ветвей и границ. Разработанный метод позволяет строить классификаторы, обеспечивающие высокую точность (в сравнении с известными классификаторами, основанными на правилах) и при этом способные интерпретировать и обосновывать результаты распознавания.

2. Разработан новый алгоритм условной оптимизации монотонных псевдобулевых функций на основе схемы метода ветвей и границ и поиска среди граничных точек допустимой области. Показано, что применение предлагаемого алгоритма к решению задачи поиска оптимальных закономерностей обеспечивает нахождение сильных первичных закономерностей за время, допускающее решение задач в интерактивном режиме (в то время как известные алгоритмы находят лишь первичные закономерности), тем самым повышая покрытия отдельных закономерностей и точность классификатора в целом.

3. Построена новая оптимизационная модель для поиска закономерностей, максимальных по отношениям доказательности и избирательности. Показано, что для выявления сильной охватывающей закономерности на основе предлагаемой модели достаточно нахождение любой крайней точки допустимой области.

4. Впервые предложен подход к решению задачи распознавания, заключающийся в использовании одновременно различных видов закономерностей – сильных первичных и сильных охватывающих. Предлагаемый подход приводит к снижению числа нераспознанных наблюдений (по сравнению с использованием одних охватывающих закономерностей), к повышению точности распознавания (по сравнению с использованием одних первичных закономерностей) и даёт возможность более детально оценить уровень надёжности результата распознавания за счёт уточнённой схемы принятия решения, использующей информацию о числе и виде закономерностей, покрывающих наблюдение.

5. Впервые разработан алгоритм поиска пары закономерностей (сильной первичной и сильной охватывающей) с использованием новой модели оптимизации и нового алгоритма псевдобулевой оптимизации. Предлагаемый алгоритм обеспечивает нахождение пар сильных закономерностей с одинаковым покрытием обучающей выборки, имеющих наименьшее (первичная закономерность) и наибольшее (охватывающая) число условий.

6. Разработана новая модель оптимизации для назначения порогов при бинаризации вещественных признаков. Предлагаемая модель обеспечивает нахождение оптимального размещения порогов вещественных признаков, при котором наблюдения разных классов являются наиболее различимыми при заданном максимальном числе порогов, что приводит к повышению покрытий отдельных закономерностей и повышению точности классификатора.

7. Впервые предложена комплексная процедура ускорения поиска закономерностей, состоящая в выборе базовых наблюдений для формирования закономерностей, отборе признаков путём решения задачи псевдобулевой

оптимизации и применении приближенного варианта нового алгоритма псевдобулевой оптимизации для выявления закономерностей. Предлагаемая процедура делает возможным применение метода оптимальных логических решающих правил для случаев большого объёма данных без существенного снижения точности классификатора.

8. Предложен новый способ повышения интерпретируемости классификатора, основанного на правилах, состоящий в применении нового алгоритма псевдобулевой оптимизации, новой схемы использования одновременно двух видов закономерностей (сильной первичной и сильной охватывающей) и в отборе достаточного числа закономерностей с ограниченным числом условий на основе решения задачи условной псевдобулевой оптимизации. Предлагаемый способ позволяет повысить обобщающую способность отдельных закономерностей и уменьшить общее число закономерностей, необходимых для распознавания, что позволяет получать более компактные решающие правила.

9. Предложен новый метод классификации электрорадиоизделий космического применения по однородным партиям, состоящий в формировании логических решающих правил, использующих результаты тестовых воздействий. Предложенный метод позволяет эффективно решать задачу формирования однородных партий при проведении отбраковочных испытаний ЭРИ.

Значение для теории: Семейство алгоритмов классификации, основанных на правилах, дополнено новым подходом построения логических алгоритмов классификации, имеющим более высокую точность. Повышение точности обеспечивается новыми моделями и алгоритмами оптимизации для поиска логических решающих правил, максимальных по отношениям простоты, доказательности и избирательности, новым способом назначения порогов при бинаризации вещественных признаков, новым способом классификации объектов на основе совместного использования сильных первичных и сильных охватывающих закономерностей. Результаты диссертации уже получили дальнейшее развитие в работах последователей. На основе метода оптимальных

логических решающих правил разрабатываются новые алгоритмы и модели, например: Кузьмич и др. (2018).

Практическая ценность: Результаты исследования могут быть востребованы в различных областях при решении задач распознавания и прогнозирования, особенно там, где решение, принимаемое по результатам работы системы распознавания, должно быть обосновано, а цена ошибки может быть велика.

Предлагаемый метод оптимальных логических решающих правил создаёт основу для синтеза новых систем поддержки принятия решений при распознавании и прогнозировании. Наиболее важным преимуществом таких систем является способность интерпретировать получаемые решения и обосновывать даваемые рекомендации. Зачастую наличие таких возможностей является первостепенным для работы пользователя при решении практических задач распознавания и прогнозирования.

Применение предлагаемого метода для решения задачи прогнозирования осложнений инфаркта миокарда дает значимые преимущества (по сравнению, например, с ранее использованным способом решения задачи с помощью искусственных нейронных сетей), заключающиеся в возможности объяснения и обоснования результатов прогнозирования, с точностью распознавания, превосходящей известные логические алгоритмы классификации.

Предлагаемый метод был применен для решения задачи классификации электрорадиоизделий (ЭРИ) по однородным партиям, что позволило получить явные правила классификации, и выполнять классификацию на основе части признаков (результатов тестовых воздействий), которые являются значимыми (значимыми комплексно или комбинаторно, а не только индивидуально).

Практическая реализация результатов. Результаты исследования использованы для решения задачи классификации ЭРИ по производственным партиям для комплектации радиоэлектронной аппаратуры космических аппаратов. Построенные модели оптимизации для нахождения закономерностей в данных и разработанные алгоритмы псевдобулевой оптимизации легли в основу

программы для ЭВМ, позволяющей выявлять закономерности в данных тестовых испытаний ЭРИ и использовать их для классификации ЭРИ. Программа классификации ЭРИ по производственным партиям легла в основу СППР в АО «ИТЦ – НПО ПМ» (г. Железногорск).

Апробация. Основные положения и результаты работы докладывались и прошли всестороннюю апробацию на международных и всероссийских научных и научно-практических конференциях и семинарах. В их числе: VII Международная конференция «проблемы оптимизации и их приложения» Optimization Problems and Their Applications (ОРТА-2018, г. Омск), Седьмая Международная конференция «Системный анализ и информационные технологии» (САИТ-2017, г. Светлогорск Калининградской области), 30th International Business Information Management Association Conference (IBIMA 2017, Мадрид, Испания), Международная научно-практическая конференция «Решетневские чтения» (2017, 2018 гг., Красноярск-Железногорск), VI Международная конференция «Проблемы оптимизации и экономические приложения» (Омск, 2015), Всероссийская научно-практическая конференция «Информационно-телекоммуникационные системы и технологии» (ИТСиТ-2014, Кемерово) и др.

Научные и научно-технические результаты были получены в рамках тематических планов НИР СибГУ им. М.Ф. Решетнёва (2008-2019 гг.), хозяйственного договора между СибГУ им. М.Ф. Решетнёва и ОАО «ИТЦ-НПО ПМ» (2014-2019 гг.).

Основные результаты исследований были отмечены Правительством и Законодательным собранием Красноярского края Государственной премией Красноярского края в области профессионального образования в 2016 году.

Публикации. По материалам диссертации опубликовано 84 работы, в том числе 25 статей в ведущих российских рецензируемых периодических изданиях, рекомендуемых ВАК РФ для опубликования основных научных результатов диссертационных исследований, 12 статей в изданиях, включенных в международные базы цитирования Web of Science, Scopus и Mathematical Reviews, 5 монографий. Зарегистрировано 6 программ для ЭВМ.

Структура работы. Диссертация состоит из введения, шести глав, заключения и списка литературы из 250 наименований.

ГЛАВА 1. АНАЛИЗ ПОДХОДОВ К ПОСТРОЕНИЮ КЛАССИФИКАТОРОВ, ОСНОВАННЫХ НА ПРАВИЛАХ

1.1 Принципы автоматического обучения

Автоматическое обучение связано с компьютерными системами, которые используют набор обучающих материалов для разработки своего рода внутреннего представления, позволяющего им делать выводы, выходящие за рамки источника обучения. Чему могут научиться компьютерные системы? Например, их можно обучить прогнозировать, страдает ли пациент от определенного заболевания, наблюдая за предыдущими пациентами, или решать, заслуживает ли кредитодатель доверия на основании предыдущих записей о кредитах. Их также можно научить распознавать рукописные или печатные символы, обнаруживать наличие лиц или объектов на изображениях и даже распознавать человека по изображению его лица. При определенных условиях некоторые системы могут даже научиться понимать человеческую речь. Это иллюстрации областей, в которых компьютерные системы могут учиться на примерах и обобщать приобретенные возможности для новых случаев, которые не использовались на этапе обучения. Автоматизированные системы обучения приобретают все большую приемлемость и применимость, и помимо своей способности давать приблизительно правильные ответы на рассматриваемые проблемы, они могут быть полезными инструментами для лучшего понимания человеком самой проблемы. Например, если медицинский персонал может неохотно доверять изученной системе для выдачи важных диагностических данных, он может воспользоваться преимуществами знания соответствующего процесса принятия решения. То есть, если система способна объяснить свои решения, она может дать доктору новое ценное понимание его собственной области. Все эти методы обычно рассматриваются в области *машинного обучения* [1].

С появлением информационных технологий и их постоянно растущей способности обрабатывать и хранить большие объемы данных возникает необходимость в инструментах, которые могут извлекать значимую информацию из, возможно, в значительной степени зашумленных данных, в которых могут присутствовать данные, не относящиеся к делу, и позволять делать полезные выводы из наиболее представительных из них. Инструменты машинного обучения играют важную роль и в этом процессе. Обнаружение знаний в базах данных [2] включает в себя процессы сбора, подготовки и обработки данных для извлечения знаний, в основе которого лежат методы машинного обучения.

Подходы к автоматическому обучению можно разделить по крайней мере на два типа: с учителем и без учителя. В первом случае система получает обучающие случаи, каждый из которых представлен набором входных значений вместе с правильным ответом для него. Цель состоит в том, чтобы приобрести способность правильно реагировать на новые случаи. *Классификация* - это техника обучения с учителем, в которой результат прогноза содержится в наборе предопределенных *классов*. Это относится ко всем вышеупомянутым примерам автоматических систем обучения. Набор классов может быть набором слов или фонем (при автоматическом распознавании речи), набором символов и цифр (при распознавании символов), но он также может содержать только два класса, например «здоровый / больной пациент» или «платежеспособный / не платежеспособный соискатель кредита». Эта диссертация посвящена проблемам классификации. Другим видом обучения с учителем является *регрессия*, где результат является оценкой численного значения.

При обучении без учителя машине не предоставляются выходные данные, только входные значения, поэтому она должна попытаться самостоятельно найти значимую информацию. Примеры методов обучения без учителя включают кластеризацию, моделирование распределения вероятности данных и нахождение в них других видов структур [3-6]. Например, методы кластеризации пытаются найти естественное распределение данных в кластеры без какой-либо предварительной информации о членстве в классе, что можно рассматривать как

попытку определить естественные классы данных. Название «без учителя» не означает, что процесс обучения, осуществляемый с использованием этих методов, является полностью автономным. На самом деле он руководствуется некоторой мерой оптимальности, как в случае с методами обучения с учителем.

Как видно, выход в задачах классификации представляет собой дискретный набор классов. Что касается ввода, то он состоит из вектора значений, которые описывают случаи, которые должны быть классифицированы. Эти значения называются *атрибутами* или *признаками*, и они должны позволять машине создавать структуры, различающие случаи, которые принадлежат к разным классам. Например, мы можем разумно согласиться с тем, что естественный цвет глаз или волос пациента не помогает диагностировать его как здорового или страдающего каким-либо заболеванием сердца. Эти два признака, безусловно, не помогут решить, имеет ли кто-то право на получение кредита либо нет. И напротив, возраст и кровяное давление для проблемы с сердцем, или годовой доход для проблемы с назначением кредита, вероятно, будут считаться важными отличительными признаками.

В любой практической задаче классификации мы предполагаем, что между входными данными и выходными данными имеется базовое, неизвестное стохастическое отображение, которое представлено обучающими данными. Предположим, что это отображение могло быть изучено точно, если бы машина имела доступ к исчерпывающему набору входных данных, то есть если бы она могла узнать правильный ответ для каждого возможного ввода. Можно легко сделать вывод, что в такой ситуации ни одна система обучения не будет полезной, поскольку ответы на каждый возможный классификационный запрос были бы известны. Кроме того, ни одна практическая ситуация не предоставляет возможности доступа ко всему набору входных данных, что соответствует рассмотрению каждой возможной комбинации значений входных признаков.

Выше мы использовали термин «случаи» для обозначения единиц данных, которые используются для обучения, а также для тех, которые должны быть классифицированы. Эти объекты будут называться *наблюдениями* или *примерами*,

и они соответствуют пациентам, соискателям кредита, изображениям или интервалам речевых аудиосигналов, чтобы соответствовать примерам, упомянутым ранее. Предполагаем, что для данной задачи классификации все наблюдения набора данных описываются одним и тем же набором признаков, что позволяет хранить их в таблице данных, где каждая строка содержит один пример и каждому столбцу соответствует один признак. Кроме того, примеры, используемые для обучения, содержат дополнительный столбец с *целевым* признаком, который является выходным значением наблюдения. Этот специальный признак также называется *меткой класса*.

Чтобы построить систему классификации, используется *алгоритм обучения*, цель которого состоит в том, чтобы создать модель отношения между входом и выходом для данной задачи классификации. Это скрытое отношение называется *целевым концептом* [7], функцией, которую нужно выявить. Модель, созданная алгоритмом обучения, называется *моделью классификации*. *Классификатор* - это система, которая использует сгенерированную модель для предоставления решений классификации. Для построения модели классификации алгоритму обучения дается конечный набор данных, представляющих отображение, которое должно быть выявлено. Требуется, чтобы созданный классификатор правильно классифицировал случаи, которые не участвовали в обучении, но которые также представляют целевой концепт. *Обобщающая способность* классификатора оценивается тем, насколько он может обобщать то, чему он обучился, на новые ситуации, то есть его способностью правильно классифицировать примеры из той же совокупности, что и данные обучения, но которые не были представлены ранее.

Способность к обобщению является важной мерой качества классификатора. Во всех практических ситуациях его необходимо оценивать с использованием некоторого конечного объема данных, прежде чем система будет внедрена на практике. Общий подход заключается в разделении данных, доступных для создания классификатора, на два разных набора: *обучающая выборка* и *контрольная выборка* (тестовая), первый используется для обучения

классификатора, а второй - для проверки его эффективности. Обучающая выборка предоставляется алгоритму обучения с метками классов, а контрольная выборка - без меток. *Точность* классификатора выражается в пропорции контрольных наблюдений, которые он правильно классифицирует. Эквивалентной мерой является *коэффициент ошибок* - доля ошибочно классифицированных контрольных наблюдений. Обучающая выборка в некоторых случаях может быть разделена на два подмножества: фактический обучающая выборка и проверочная выборка.

При решении задачи классификации могут возникать определенные трудности. В зависимости от конкретной рассматриваемой проблемы, а также от количества и качества доступных данных, редко бывает, когда точность классификатора приближается к 100%, на самом деле, обычно она значительно ниже этого уровня, особенно если условия обучения «жесткие». Это объясняется несколькими факторами: объем обучающей выборки данных ограничен и иногда недостаточен; выбор входных признаков может быть неоптимальным; возможности моделирования самих алгоритмов обучения имеют ограничения; или данные могут быть искажены. Искажение данных может происходить как минимум в двух разных формах: (а) они могут содержать *шум*, что означает, что определенные метки класса назначены неправильно или что некоторые значения признаков были неправильно измерены; (б) некоторые наблюдения могут содержать *пропущенные значения признаков*. Это нормальные ситуации, с которыми должен справляться алгоритм обучения.

1.2 Формальное описание задачи классификации

Опишем основные обозначения для задач классификации, которые будут использоваться на протяжении всей диссертации.

Алгоритм обучения строит оценку $\hat{F}$ неизвестного целевого концепта F, которая определяется как отображение *входного пространства* $\Omega \equiv R^A$, где A -

количество признаков, в *выходное пространство* $\Sigma \equiv \{c_1,...,c_K\}$ дискретных классов, которые могут быть упорядочены или нет, где K - количество классов. Этап обучения осуществляется с использованием множества X обучающих данных вида $\langle x_n, F(x_n)\rangle, n \in \{1,...,N\}$, где x_n - это пример, $F(x_n)$ - это метка класса, а N - количество примеров в X.

Можно сказать, что функция $F: \Omega \to \{c_1,...,c_K\}$ определяет K-разбиение входного пространства на подмножества, каждое из которых относится к некоторому классу. Следовательно, набор обучающих данных X также разбивается на подмножества классов $X = \bigcup_{k=1}^{K} X_k$. Функция $\hat{F}: \Omega \to \{c_1,...,c_K\}$ является *решающим правилом*, и она принимает различную форму в зависимости от конкретного типа модели классификации.

Набор обучающих данных X является подмножеством полного пространства признаков $X \subseteq \Omega$, и созданная модель $\hat{F}$ должна обобщаться на примеры пространства признаков, которые не содержатся в X. Точность классификатора, то есть качество полученного приближения $\hat{F}$, можно измерить эмпирически, используя контрольную выборку $X_{контр}$ примеров, согласующихся с F, но не пересекающихся с X ($X_{контр} \cap X = \varnothing$).

1.3 Основные элементы теории обучения

Аппроксимация $\hat{F}$ целевого концепта F может быть реализована с помощью большого разнообразия различных моделей. В следующем разделе кратко представлены некоторые основные типы алгоритмов обучения и модели, которые они создают. Каждый тип модели классификации реализуется с помощью некоторого репрезентативного языка (языка гипотез), который может варьироваться от очень простого до очень сложного. Этот язык определяет, как информация, полученная в результате обучения, хранится и используется для принятия новых решений классификации.

Некоторые типы репрезентативных языков являются очень мощными и способны моделировать очень сложные информационные структуры, в то время как другие более ограничены. Существует связь между репрезентативной силой моделей классификации и способностью соответствующих алгоритмов правильно решать задачи классификации.

Можно считать, что каждый алгоритм обучения имеет доступ к пространству функций, также называемому пространством *гипотез*. Это пространство определяется репрезентативным языком конкретного типа моделей. Другими словами, каждый тип модели реализует функции определенного класса или семейства. Это может быть класс всех нейронных сетей, класс всех деревьев решений и т.д. Цель алгоритма обучения - найти в пространстве гипотез (классе функций) наиболее подходящую для конкретной задачи.

Выбор наилучшей гипотезы зависит от конечного набора обучающих данных. Этот факт имеет два основных следствия: (а) если пространство гипотез слишком широкое, это позволяет алгоритму выбрать гипотезу, которая слишком точно соответствует обучающим данным, что может привести к плохому обобщению полученной модели; и (б) чем больше обучающих примеров доступно, тем выше вероятность того, что выбранная гипотеза будет правильно аппроксимировать фактический целевой концепт *F*.

Репрезентативная сила класса функций *C* определяется *емкостью* класса *C*. Вапник [8] разработал важные основы теории статистического обучения, в которых он формально определяет емкость класса функций *C* как максимальное число *N* обучающих наблюдений, которые могут быть правильно классифицированы функцией в *C* независимо от меток классов, назначенных для этих *N* наблюдений.

Любой успешный алгоритм обучения должен иметь некоторый контроль над емкостью ради способности к обобщению. Одна из возможностей для ограничения емкости модели состоит в рассмотрении класса функций с ограниченной репрезентативной силой, что предотвращает чрезмерную адаптацию модели к обучающим данным. Другой, обычно более удачной

альтернативой является интеграция в алгоритм обучения механизмов, позволяющих избегать слишком сильных гипотез. Это может быть сделано с использованием либо техники проверки, либо статистических показателей достоверности. Первая состоит в том, чтобы отложить часть обучающих данных, чтобы измерить обобщение различных гипотез во время выполнения алгоритма. Эти данные обычно называют *проверочной выборкой* (не путать с контрольной или экзаменационной выборкой). С другой стороны, статистические методы могут использоваться для расчета оценок эффективности каждой гипотезы на будущих примерах, основываясь только на поведении на обучающих данных. Причина, по которой эти два подхода лучше, чем использование гипотез с ограниченной емкостью, заключается в том, что они могут адаптироваться к особенностям каждой конкретной задачи обучения.

Рисунок 1.1, который приведен в [9], показывает простой пример задачи классификации, когда обучающие данные представлены в двухмерном пространстве (двумя признаками, a_1 и a_2) и разделены на два класса. На каждом графике показана функция, реализующая границу решения. Можно заметить, что функция в (а) имеет низкую емкость. Это приводит к нескольким ошибкам обучения, и, вероятно, будет приводить к ошибкам также на контрольных данных. Представляется, что в (б) функция имеет необходимый уровень емкости. Несмотря на несколько ошибок на обучающих данных, она, вероятно, будет обобщать правильно. Наконец, функция, представленная в (с), обладает высокой емкостью и может очень точно моделировать обучающие данные. Однако результаты по контрольным данным, вероятно, будут разочаровывающими. В этом случае говорят, что модель *переобучена* на данных. Отметим, что эти выводы основаны на предположении, что обучающие данные могут быть зашумлены, что объясняет утверждение, что функция в (b) является наиболее подходящей для этой задачи.

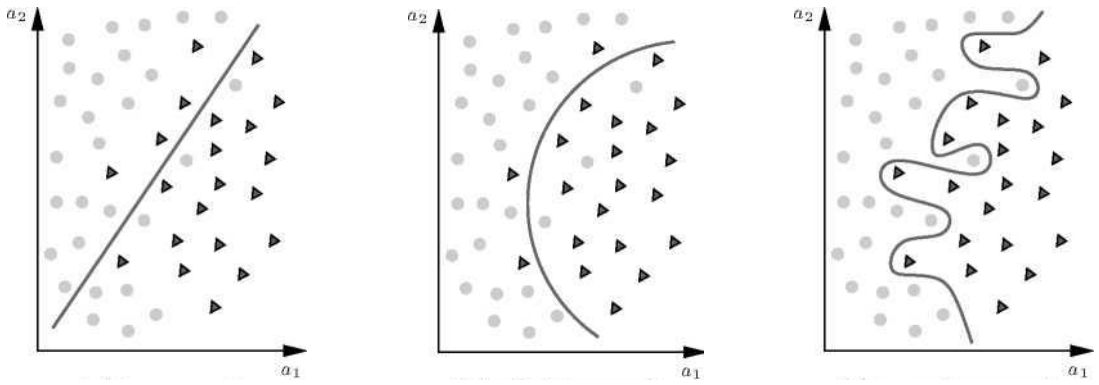


Рисунок 1.1 - Граница решения, реализованная тремя функциями, каждая с разным уровнем емкости, для одной и той же задачи классификации [9]. (a) низкая емкость, (b) подходящая емкость, (c) избыточная емкость.

На практике ошибка обучения не имеет значения с точки зрения классификации, именно ошибка обобщения (рассчитанная на контрольных данных) показывает меру качества классификатора. Чем меньше обучающих наблюдений доступно, тем легче создать модель, которая их правильно классифицирует. Однако это приводит к высокой ошибке на контроле. С другой стороны, использование многих обучающих примеров позволяет приближению $\hat{F}$ целевого концепта F быть лучше, возможно, за счет более высокой ошибки обучения.

1.4 Классификаторы, основанные на правилах

Методы, основанные на правилах, являются популярным классом методов машинного обучения и интеллектуального анализа данных [10]. Они нацелены на поиск закономерностей в данных, которые могут быть выражены в форме правил ЕСЛИ-ТО. В зависимости от типа правила, которое должно быть найдено, мы можем различать *обнаружение описательного правила*, которое нацелено на описание значимых закономерностей в наборе данных в терминах правил, и *выявление предсказательных правил*. В последнем случае часто также интересно узнать набор правил, которые в совокупности охватывают пространство

наблюдений в том смысле, что они могут сделать прогноз для каждого возможного наблюдения. Далее кратко представим обе задачи и укажем на некоторые ключевые работы в этой области.

При обнаружении описательных правил основной упор делается на поиск правил, описывающих закономерности, которые можно наблюдать в предоставленном наборе данных. В отличие от выявления предсказательных правил, основное внимание уделяется поиску индивидуальных правил. Следовательно, оценка, как правило, фокусируется не на прогнозной эффективности, а на статистической достоверности найденных правил. В литературе преобладают две основные задачи, а именно: обнаружение подгруппы, где наблюдается представляющее интерес свойство (обучение с учителем), и обнаружение ассоциативных правил, где могут рассматриваться произвольные зависимости между признаками (обучение без учителя).

Задача обнаружения подгруппы была определена [11] и Вробелем [12] следующим образом: Для заданной популяции индивидуумов и свойств индивидуумов, которые нас интересуют, найти подгруппы популяции, являющиеся статистически «наиболее интересными», например: настолько велики, насколько это возможно, и обладают наиболее необычными статистическими характеристиками по отношению к нужному свойству.

Таким образом, подгруппа может рассматриваться как правило ЕСЛИ-ТО, которое связывает набор независимых переменных с целевой переменной, представляющей интерес. Условие правила (*тело правила* или *антецедент*) обычно состоит из конъюнкции булевых термов, так называемых *характеристик*, каждая из которых представляет собой ограничение, которое должно быть удовлетворено наблюдением. Если все ограничения выполняются, то говорят, что правило срабатывает, и пример считается *покрытым* правилом. *Заголовок правила* (также называемый *последствием* или *выводом*) состоит из единственного значения класса, которое прогнозируется в случае срабатывания правила. В простейшем случае это бинарный целевой класс *c*, и мы хотим найти

одно или несколько правил, которые являются предсказательными для этого класса.

В литературе также можно найти несколько близких задач, где заголовок правила состоит не только из одного бинарного признака. Примеры включают в себя разработки по обнаружению подгрупп, множеств контрастов [13], коррелированных закономерностей [14], появляющихся закономерностей [15] и другие. Дополнительную информацию можно найти в работах Kralj Novak [16] и

Zimmermann и De Raedt [17], которые представляют обзорные исследования этих подходов.

Тела правил обычно состоят из характеристик, которые проверяют наличие определенного значения признака или, в случае числовых признаков, выполнение неравенства, которое требует, чтобы наблюдаемое значение было выше или ниже порога. Более сложные ограничения включают в себя *множественно-значимые признаки* (несколько значений одного и того же признака, которое можно наблюдать в обучающих примерах), *внутренние дизъюнкции* (должно присутствовать только одно из нескольких значений одного и того же признака), *иерархические признаки* (определенные значения признаков включаются в другие значения) и т.д. Конъюнктивную комбинацию характеристик можно рассматривать как утверждения в логике высказываний (правила высказываний). Если могут быть рассмотрены отношения между характеристиками (то есть если высказывания могут быть сформулированы в логике первого порядка), то говорят о правилах первого порядка.

В то время обнаружение описательных правил направлено на поиск отдельных правил, которые фиксируют некоторые закономерности во входных данных, задача обучения правилам прогнозирования заключается в обобщении обучающих данных, чтобы можно было делать предсказания для новых примеров. Поскольку отдельные правила обычно охватывают только часть обучающих данных, необходимо обеспечить полноту, построив неупорядоченный набор правил или список решений.

Неупорядоченный набор правил - это набор отдельных правил, которые в совокупности образуют классификатор. В отличие от списка решений, правила в наборе не имеют собственного порядка, и все правила в наборе должны быть задействованы для получения прогноза для примера. Это может вызвать два типа проблем, которые должны быть решены с помощью дополнительных алгоритмов:

Срабатывание нескольких правил: в одном примере может срабатывать несколько правил, и эти правила могут давать противоречивые прогнозы. Этот тип конфликта, как правило, разрешается путем предпочтения правил, которые охватывают более высокую долю обучающих примеров своего класса. Это эквивалентно преобразованию набора правил в список решений, упорядоченный в соответствии с этой эвристикой оценки. Также известны более сложные схемы, такие как использование наивного байесовского алгоритма или создание отдельного набора правил для обработки этих конфликтов (двойная индукция [18]).

Нет сработавших правил: может также случиться, что ни одно правило не срабатывает для данного примера. Такие случаи обычно обрабатываются с помощью так называемого правила по умолчанию, которое обычно предсказывает класс большинства. Опять же, были предложены более сложные алгоритмы, такие как FURIA [19], пытающиеся найти наиболее близкое правило (правило растяжения [20]).

Набор правил, в котором все правила предсказывают один и тот же класс, должен быть дополнен (неявным) правилом по умолчанию, которое предсказывает другой класс на случай, если не сработает ни одно из предыдущих правил. Если все правила являются конъюнкциями, то такие наборы правил могут интерпретироваться как *дизъюнктивная нормальная форма* для этого класса.

В отличие от неупорядоченного набора правил, *список решений* имеет свойственный порядок, что делает классификацию довольно простой. Для классификации нового наблюдения правила пробуются по порядку, и прогнозируется класс первого правила, которое покрывает наблюдение. Если никакие правила не срабатывают, то вызывается *правило по умолчанию*, которое

обычно прогнозирует класс большинства непокрытых обучающих наблюдений. Списки решений особенно популярны в *программировании индуктивной логики* [21, 22], поскольку программы PROLOG можно рассматривать как простые списки решений, где все правила предсказывают один и тот же концепт.

1.5 Покрывающие алгоритмы

Наиболее популярными алгоритмами построения логических правил для распознавания являются алгоритмы «отделяй и властвуй» или покрывающие алгоритмы [23]. Все они имеют один и тот же цикл верхнего уровня: в начале алгоритм «отделяй и властвуй» ищет правило, которое объясняет часть его обучающих наблюдений, «отделяет» эти наблюдения и рекурсивно «властвует» над оставшимися наблюдениями, формируя правила до тех пор, пока непокрытых наблюдений не останется. Это гарантирует, что каждый экземпляр исходного набора обучающей выборки покрывается, по крайней мере, одним правилом.

Хорошо известно, что каждый алгоритм обучения основан на некоторой идее или предубеждении (в англоязычных источниках используется термин «bias» или «inductive bias»), которое необходимо для получения обобщения. Митчелл [24] определил bias как любая основа для выбора одного обобщения над другим.

Таким образом, алгоритм обучения может характеризоваться предубеждениями, которые он использует. В то время как основной цикл верхнего уровня является одинаковым для всех алгоритмов семейства «отделяй и властвуй», методы формирования отдельных правил могут значительно варьироваться для разных алгоритмов. Можно охарактеризовать алгоритмы «отделяй и властвуй» с помощью трех измерений [23]:

Язык: Пространство поиска для алгоритма обучения определяется его языком гипотез. Некоторые концепты не могут быть выражены или могут иметь неудобное представление на определенных языках гипотез.

Поиск: Обучающие алгоритмы различаются способом поиска в пространстве гипотез. Он включает в себя алгоритм поиска (например, жадный алгоритм), стратегию поиска (сверху вниз или снизу вверх) и эвристические критерии поиска, которые используются для оценки найденных гипотез.

Предотвращение переобучения: Многие алгоритмы используют эвристики для предотвращения переобучения на зашумленных данных. Они могут предпочесть простые правила более сложным, даже если точность более простых правил на обучающих данных ниже, в надежде, что их точность на контрольных данных будет выше.

Все эти три измерения являются разновидностями inductive bias.

Опишем в общем виде алгоритм «отделяй и властвуй», на основе которого строятся различные существующие (и новые) алгоритмы обучения, определенные разными предубеждениями [23].

Стратегия «отделяй и властвуй» берет свое начало в семействе алгоритмов AQ [25] под названием «стратегия покрытия». Термин «отделяй и властвуй» был придуман Пагальо и Хаусслером [26] для описания стратегии обучения: сформировать правило, которое покрывает часть заданного обучающего множества, и рекурсивно формировать другие правила, которые покрывают часть из оставшихся примеров до тех пор, пока примеров не останется.

AQ можно рассматривать как оригинальный покрывающий алгоритм. Его оригинальная версия была предложена Ришардом Михальски в 60-х годах [25], многочисленные версии и варианты алгоритма появились впоследствии в литературе. AQ использует поиск «сверху вниз» для нахождения лучшего правила. Он не ищет все возможные варианты правила, а рассматривает только уточнения, которые охватывают конкретный пример, так называемый начальный пример.

CN2 [27, 28] применяет поиск на основе оценок Лапласа, и использует критерий значимости отношения правдоподобия для борьбы с переобучением. Он может работать в двух режимах: один для обучения множества правил (путем

моделирования каждого класса независимо) и один для обучения списков решений.

FOIL [29] был первым алгоритмом реляционного обучения, который привлек внимание за пределами программирования индуктивной логики. Он строит концепт с помощью цикла покрытий и обучает отдельные концепты с помощью оператора уточнения «сверху вниз», опираясь на прирост информации. Основным отличием от предыдущих систем является то, что FOIL позволял использовать знания первого порядка. Вместо того, чтобы использовать только тесты для отдельных признаков, FOIL может использовать тесты, которые вычисляют отношения между несколькими признаками, а также может вводить новые переменные в тело правила.

RIPPER была первой системой обучения правилам, которая эффективно противодействовала проблеме переобучения путем постепенного сокращения ошибок [30]. Также добавлена фаза пост-обработки для оптимизации набора правил. Основная идея состоит в том, чтобы удалить одно правило из ранее полученного набора правил и попытаться повторно индуцировать его не только в контексте предыдущих правил (как это было бы в случае обычного правила покрытия), но также в контексте последующего правила. RIPPER по-прежнему считается эффективным алгоритмом среди алгоритмов индуктивного обучения правил. Свободно доступная реализация может быть найдена в библиотеке машинного обучения WEKA [31] под названием JRIP.

OPUS [32] был первым алгоритмом обучения правил, который продемонстрировал возможность полного переборного поиска по всем возможным телам правил для нахождения правила, которое максимизирует заданный критерий качества (или эвристическую функцию). Ключевой идеей является использование упорядоченного поиска, который предотвращает создание правила несколько раз. Кроме того, OPUS использует несколько методов, которые сокращают значительную часть пространства поиска, поэтому этот метод поиска становится возможным.

Алгоритмы «отделяй и властвуй» были разработаны для различных задач обучения. Например (по типам формируемых правил):

Множества правил в виде ДНФ: AQ (Michalski, 1969 [25]), PRISM (Cendrowska, 1987 [33]), PFOIL (Mooney, 1995 [34]), GROW (Cohen, 1993 [35]), RIPPER (Cohen, 1995 [36]), CORAL (Лбов Г.С., Котюков В.И., Манохин А.Н., 1973 [37]), GARIPPER (Yang, Tiyyagura, Chen, Honavar, 1999 [38]), RipMC (Asadi, Shahrabi, 2016 [39]), DiRUC (Govada, Thomas, Samal, Sahay, 2016 [40]).

Множества правил в виде КНФ: PFOIL-CNF (Mooney, 1995 [34]), ICL (De Raedt, Van Laer, 1995 [41]).

Списки решений: CN2 (Clark, Niblett, 1989 [42]; Clark, Boswell, 1991 [28]), GREEDY3 (Pagallo, Haussler, 1990 [26]), FOIDL (Mooney, Califf, 1995 [43]), ДРЭТ (Лбов Г.С., 1981 [44]), ACORI (Asadi, Shahrabi, 2016 [45]).

Логические программы: INDUCE (Michalski, 1980 [46]), FOIL (Quinlan, 1990 [29]; Quinlan, Cameron-Jones, 1995 [47]), I-REP (Furnkranz, Widmer, 1994 [48]), PROGOL (Muggleton, 1995 [49]), ProbFOIL (Raedt, Thon, 2011 [50]), ProbFOIL+ (Raedt, 2015 [51]).

Правила регрессии: IBL-SMART (Widmer, 1993 [52]), RULE (Weiss, Indurkhya, 1993, 1995 [53, 54]), FORS (Karalic, Bratko, 1997 [55]), PCR (Zenko, 2005 [56]), GuideR (Sikoraa, Wróbel, Gudyś, 2019 [57]).

Задача обучения с помощью правил заключается в следующем [23]. Даны положительные и отрицательные примеры целевого концепта, описанные фиксированным числом признаков. Цель алгоритма - найти описание целевого концепта в форме явных правил, сформулированных в терминах тестов для определенных значений признаков. Результирующий набор правил должен быть способен правильно распознавать экземпляры целевого концепта и отличать их от объектов, которые не относятся к целевому концепту.

Существуют различные подходы к решению этой задачи. Наиболее часто используемой альтернативой является обучение дерева решений через стратегию «разделяй и властвуй» («divide-and-conquer» [58]). Высокая популярность деревьев решений связана с их эффективностью в обучении и классификации

[59]. Кроме того, деревья решений можно легко превратить в правило, создав одно правило для каждого пути от корня до листа [23].

Тем не менее, есть несколько аспектов, которые делают формирование правил по стратегии «отделяй и властвуй» порой более привлекательными:

1. Деревья решений часто оказываются довольно большими и трудны для понимания. Quinlan [60] отметил, что даже усеченные деревья решений могут быть слишком громоздкими и сложными для понимания рассматриваемой ситуации, и разработал процедуры упрощения деревьев решений в наборы правил. Дополнительные подтверждения этого исходят от Rivest [61], который показал, что списки решений (упорядоченные наборы правил) с не более чем k условиями на правило являются более понятными, чем деревья решений с глубиной k. Аналогичные выводы были сделаны у Bostrom [59].

2. Обучающие алгоритмы в деревьях решений создают неперекрывающиеся правила, что накладывает серьезные ограничения на формируемый набор правил. Одной из проблем, возникающих из этого ограничения, является проблема репликации поддерева [26]: наличие одинаковых поддеревьев в разных местах дерева решений, что вызвано ограничением на неперекрываемость правил в стратегии разделения. Алгоритмы типа «отделяй и властвуй» не создают такого ограничения и, следовательно, не восприимчивы к этой проблеме.

Ниже показан простой алгоритм «отделяй и властвуй» [33].

Вход: D — обучающая выборка,

Выход: R — множество правил.

1: $D_U \leftarrow D$ множество непокрытых примеров

2: $R \leftarrow \emptyset$ старт с пустого множества правил

3: **повторять**

4: $r \leftarrow \emptyset$ старт с пустого правила

5: $r \leftarrow \text{Grow}(r, D, D_U)$ процедура наращивания

6: $r \leftarrow \text{Prune}(r, D)$ процедура обрезки

7: $R \leftarrow R \cup \{r\}$

8: $D_U \leftarrow D_U \setminus \mathrm{Cov}(r, D_U)$ удалить из D_U примеры, покрытые r

9: **до тех пор, пока не выполнится** $D_U = \varnothing$

Алгоритм начинает работу с пустой теории (решающей функции) и последовательно добавляет к ней правила, пока не будут покрыты все положительные наблюдения [23]. Формирование отдельных правил начинается с правила, которое всегда истинно (покрывает все наблюдения, т.е., в данном контексте, верно распознает положительные наблюдения и неверно распознает отрицательные). Текущее правило специализируется путем добавления в него условий до тех пор, пока оно ещё покрывает (неправильно распознает) отрицательные наблюдения. Возможными условиями являются тесты на наличие определенных значений различных признаков. Чтобы достигнуть цели поиска – правила, которое не покрывает отрицательных наблюдений, алгоритм выбирает тест, который оптимизирует чистоту правила, т. е. тест, который максимизирует долю положительных наблюдений среди всех покрываемых наблюдений. Когда найдено правило, которое покрывает только положительные наблюдения, все покрытые наблюдения удаляются, а следующее правило формируется на оставшихся наблюдениях. Это повторяется до тех пор, пока не останется наблюдений. Таким образом, обеспечивается, чтобы полученные правила вместе охватывали все данные положительные наблюдения (условие полноты), но ни одно из отрицательных наблюдений (условие согласованности).

Все алгоритмы типа «отделяй и властвуй» расширяют основную структуру этого простого алгоритма. Однако для многих задач обучения требуются модификации этой процедуры. Например, если данные являются зашумленными, использование полных и согласованных наборов правил может привести к переобучению. Таким образом, многие алгоритмы ослабляют это ограничение и используют критерии остановки или методы пост-обработки, чтобы иметь возможность получать более простые теории (наборы правил), которые не являются полными и согласованными, но часто более предсказательны на новых данных [23]. Другие алгоритмы заменяют поиск сверху вниз внутреннего цикла

на поиск снизу вверх, где правила, наоборот, последовательно обобщаются, начиная с самого конкретного правила (например, покрывающего лишь одно положительное наблюдение). Одни алгоритмы используют жадную эвристику, другие – менее «близорукие» алгоритмы поиска.

Отметим, что мы рассматриваем бинарную задачу классификации: цель индуцированных правил состоит в том, чтобы различать положительные и отрицательные наблюдения целевого концепта. На основе этого предположения основаны многие алгоритмы обучения «отделяй и властвуй», в частности алгоритмы, используемые в индуктивном логическом программировании [23]. В этом случае порядок, в котором правила используются для классификации, не имеет значения, поскольку правила описывают только один класс – положительный. Отрицательные наблюдения классифицируются с помощью невыполнения правил, т.е. когда для данного наблюдения не срабатывает ни одно правило, оно будет классифицироваться как отрицательное. Это эквивалентно принятию правила по умолчанию для отрицательного класса в конце списка упорядоченных правил.

Однако многие практические задачи связаны с многозначными или даже непрерывными переменными класса. В таких задачах с несколькими классами или задачах регрессии порядок правил очень важен, поскольку каждый пример может быть покрыт несколькими правилами, которые делают разные прогнозы. Таким образом, другой порядок правил может сделать другое предсказание для одного и того же наблюдения. Эта проблема известна как проблема перекрытия [23].

Теории (решающие функции), которые имеют определенный порядок оценки по полученным правилам, обычно называются списками решений [61]. К примеру, алгоритм CN2 [27] предназначен для решения задач с помощью списка решений. Для этой цели он использует функцию оценки, отдающую предпочтение такому распределению классов, при котором доминируют наблюдения одного класса. Каждое сформированное правило указывает на класс, доминирующий среди наблюдений, которые оно покрывает. Обучение

прекращается, когда все наблюдения покрываются хотя бы одним правилом [23]. Для обработки противоречий (когда срабатывают несколько правил) CN2 располагает правила в том порядке, в котором они были сформированы. Это считается естественной стратегией при формировании списка решений и объясняется тем, что большинство эвристик поиска, как правило, сначала формируют более общие правила.

Алгоритмы типа «отделяй и властвуй» быстро работают, и в результате получается небольшой упорядоченный список правил, который, к тому же, легко интерпретируется. Но такие алгоритмы имеют значимые изъяны. Во-первых, эти алгоритмы являются жадными алгоритмами локального поиска и поэтому не обеспечивают оптимального решения. Во-вторых, может быть недостаточно информации для принятия решения о классификации, так как решение принимается только на основе выполнения лишь одного правила (или невыполнения всех), а не на основе «голосования» правил. В-третьих, если список правил длинный, то решение трудно интерпретировать, так как нужно учитывать предшествующие правила, которые относятся к различным классам, и объяснение получается запутанным.

Использование в логических процедурах распознавания ряда улучшений, таких как бустинг, голосование правил, выделение признаков и пограничных объектов [62], позволяет существенно повысить эффективность классификатора.

Логический анализ данных (Logical Analysis of Data – LAD [63]) обладает рядом преимуществ перед алгоритмами типа «отделяй и властвуй». Во-первых, все получаемые правила могут быть оптимальными по используемому критерию (простота, избирательность или доказательность, а также их возможные совмещения). Во-вторых, классификатор не просто разделяет области классов, а строит аппроксимацию областей набором правил. Для оценки силы разделения этих областей может быть использовано понятие «зазора» [64]. В-третьих, голосование правил является средством оценки достоверности принадлежности к каждому классу.

Несмотря на относительно высокую вычислительную сложность логического анализа данных, этот подход представляется перспективным, а разработка и использование алгоритмов псевдобулевой оптимизации, реализующих свойства решаемого класса задач по формированию решающих правил, делают возможным применение этого подхода за время, допускающее решение задач в интерактивном режиме, при этом позволяя извлечь из данных максимум полезной информации.

Логический анализ данных (ЛАД) [63] представляет собой методологию для автоматического обучения, которая строит модели классификации на основе закономерностей. Закономерности - это конъюнкции определенных значений признаков, и они используются в ЛАД для описания целевых концептов, включающих только два класса. Использование закономерностей предполагает, что можно найти комбинации значений признаков, которые являются типичными для примеров одного из классов и не встречаются в другом. Классификационная модель ЛАД строится с помощью процесса, который исследует пространство поиска, чтобы найти соответствующие наборы закономерностей, обобщающие обучающие примеры и различающие данные двух классов. Как и для любого другого метода обучения, это делается на этапе обучения для того, чтобы построить модель, которая должна обобщать свои выводы для ранее неиспользованных наблюдений из той же области.

Этот подход по своей природе является логическим, так как он был разработан в рамках теории булевых функций. Его логическая природа делает его особенно приспособленным для изучения концептов, включающих два класса, и где данные представлены булевыми признаками. ЛАД использует техники, которые позволяют преобразовывать данные числового или символьного формата в булевые признаки.

Помимо предоставления решений классификации, логические закономерности являются мощным и интерпретируемым механизмом, позволяющим человеку понять проблемную область. Более подробное описание этого подхода сделано во второй главе.

Многие исследования показали, что не существует такого понятия, как лучший алгоритм классификации, так как это зависит от рассматриваемой задачи и применяемого критерия качества (например, точность классификации, сложность модели, интерпретируемость, время обучения). Одни алгоритмы лучше в некоторых задачах, но менее эффективны в других задачах. Тем не менее, есть методы, известные своей точностью классификации, например, деревья решений, искусственные нейронные сети, метод опорных векторов, которые, как известно, очень точны в самых разных задачах.

Среди всех типов алгоритмов, которые были упомянуты в этом разделе, только логический анализ данных, классификаторы, основанные на правилах и, в некотором смысле, деревья решений, генерируют модели, которые могут быть восприняты человеком и которые могут использоваться для целей, отличных от высокой точности классификации. Что касается логического анализа данных, то сравнение с другими методами на известных задачах классификации [65] показало его эффективность и конкурентоспособность. Более убедительные результаты представлены в главе 6.

Выводы к главе 1

Первоначальное направление для этого исследования было в значительной степени мотивировано подходом логического анализа данных. В начале исследования общая цель состояла в обобщении применимости подхода к более широкому кругу задач, чем те, которые были возможны в то время. Основными выявленными препятствиями были следующие:

подход имеет слабую процедуру бинаризации количественных данных, затрудняющих применение подхода для задач с разнотипными данными; этот вопрос подробно рассмотрен во *второй главе*;

существующие модели оптимизации для поиска закономерностей не обеспечивают нахождения закономерностей различных требуемых типов; исследования по этому вопросу изложены в *третьей главе*;

принятие решения при распознавании основано на использовании набора закономерностей одного типа, при том, что каждый тип закономерностей обладает своими преимуществами и недостатками; техника совместного использования закономерностей различных типов изложена в *четвертой главе*;

применяемые алгоритмы поиска закономерностей являются, по сути, жадными алгоритмами и не обеспечивают нахождения оптимальных закономерностей; алгоритмы оптимизации для поиска оптимальных закономерностей, основанные на свойствах класса задач оптимизации, изложены в *пятой главе*;

при обработке данных достаточно большого объема рассматриваемый подход может требовать чрезмерно больших вычислительных затрат; число получаемых закономерностей может быть велико, что усложняет интерпретацию классификатора; эти вопросы рассмотрены в *шестой главе*.

Следовательно, работа началась с попытки найти решение каждой из этих проблем. Этот документ показывает, что эти цели были достигнуты. Проведенное исследование затрагивает различные области, из которых наиболее важными являются: автоматическое обучение, обнаружение знаний, булева логика и алгоритмы комбинаторной оптимизации. Результаты исследования привели к созданию нового метода – метода оптимальных логических решающих правил.

ГЛАВА 2. ЛОГИЧЕСКИЙ АНАЛИЗ ДАННЫХ С РАЗНОТИПНЫМИ ПРИЗНАКАМИ

В настоящей главе исследуется подход к выявлению и использованию логических решающих правил, называемый логическим анализом данных.

Предлагаемый метод к поддержке принятия решений при распознавании основан на подходе к выявлению и использованию закономерностей, называемом логическим анализом данных (P. Hammer). Логический анализ данных — это методология анализа данных, которая объединяет идеи и концепции из оптимизации, комбинаторики и булевых функций. Согласно [66] идея логического анализа данных была впервые описана Питером Л. Хаммером в лекции, прочитанной в 1986 году на Международной конференции по принятию многофакторных решений с помощью экспертных систем на основе исследования операций [67], а затем была расширена и разработана в [68]. За этой первой публикацией последовал поток исследований, многие из которых можно найти в списке литературы. В ранних публикациях основное внимание уделялось теоретическим разработкам и вычислительной реализации. В последние годы внимание было сосредоточено на практических применениях от медицины до рейтингов кредитного риска. Цель настоящей главы - дать обзор теоретических основ этой методологии, обсудить различные аспекты ее реализации и рассмотреть некоторые из ее многочисленных приложений.

Начнем с вводного примера, предложенного в [66].

Врач хотел бы выяснить комбинацию продуктов питания, которые вызывают головную боль у одного из его пациентов, и просит своего пациента вести учет своей диеты. Через неделю пациент возвращается к врачу и вносит запись, показанную в таблице 2.1.

После краткого осмотра врач приходит к выводу, что в дни, когда у пациента не было головной боли, он никогда не принимал пищу № 2 без еды № 1, но он делал это в некоторых случаях, когда у него болела голова. Точно так же доктор приходит к выводу, что пациент никогда не принимал пищу № 4 без еды

№ 6 в те дни, когда у него не было головной боли. Но он сделал это однажды, и у него болела голова. В заключении он приходит к выводу, что две «закономерности», отмеченные выше, объясняют каждую головную боль, и выдвигает «теорию» о том, что головные боли у этого пациента всегда можно объяснить с помощью этих двух закономерностей.

Таблица 2.1 - Вводный пример - запись диеты

	Номер продукта								Головная
День	1	2	3	4	5	6	7	8	боль
1		x		x		x	x		Да
2	x		x		x		x		Нет
3				x	x	x			Нет
4	x	x		x		x		x	Нет
5	x	x		x	x			x	Да
6			x		x		x		Нет
7		x	x		x			x	Да

Очевидно, что врач должен был ответить на три вопроса.

а) Каков краткий список продуктов, достаточных для объяснения наличия или отсутствия головной боли? В нашем примере, продуктов № 1, 2, 4, 6 уже было достаточно для этой цели.

(б) Как выявлять закономерности (то есть комбинации пищевых продуктов), вызывающие головные боли? В нашем случае врач обнаружил две таких закономерности.

(в) Как построить теории (то есть набор закономерностей), объясняющих каждую наблюдаемую головную боль?

Эти три вопроса отражают суть методологии логического анализа данных.

2.1 Краткая историческая справка

Основы логического анализа данных (ЛАД) были положены в 1980-х годах, когда коллектив профессора Хаммера из Центра исследований операций при

Ратгерском университете (RUTCOR) в Нью-Джерси обнаружил возможность использования специальных видов булевых функций для извлечения знаний из данных. Первая публичная презентация этих оригинальных идей была сделана на конференции в Пассау, Германия [67], содержание которой было опубликовано позже [68].

Исследовательская группа RUTCOR продолжала регулярно работать над этой темой, в основном на теоретическом уровне. Между тем, методология развивалась с практической стороны: была создана процедура для преобразования необработанных данных в логический формат, чтобы позволить LAD обрабатывать произвольные данные, а не только логические входные данные, и были добавлены методы для обработки шума и пропущенных данных. С 1994 г. создание программной реализации ЛАД позволило провести первые эмпирические эксперименты, о которых сообщалось в [69]. Эти результаты дали толчок, чтобы стимулировать проведение дальнейших исследований в этой области. Технический отчет 1996 года [70] предоставил подробное и всестороннее описание реализации LAD как конкретного метода для анализа и классификации данных, а также результаты эмпирических экспериментов на известных наборах данных [63].

LAD был создан в контексте булевых функций, и именно эта точка зрения используется здесь для описания его основных характеристик. Однако использование логических понятий для задач классификации не является исключительной особенностью ЛАД, их можно найти в других подходах.

Прежде чем описать ЛАД, полезно определить подходящие обозначения и терминологию. Ниже приводятся основные булевы понятия, которые используются в этой диссертации.

2.2 Обозначения и терминология

Булева переменная b - это переменная, которая может принимать два значения, 0 или 1, и эти два значения образуют множество $B = \{0,1\}$.

Дополнением к булевой переменной $\bar{b}$ является ее отрицание ($\bar{b} = 1 - b$), а символы b и $\bar{b}$ называются *литералами*.

Если x является вектором булевых значений B, то функция $f : B^n \to B$, такая, что выходное значение $f(x)$ равно 0 или 1, является *булевой функцией* n переменных. Так как f имеет выход в B для каждого возможного входного вектора x, то он определяет разбиение множества B^n на два непересекающихся подмножества T и F, соответственно подмножество истинных (положительных) векторов и подмножество ложных (отрицательных) векторов в B^n.

Определяем *множество булевых данных* как набор векторов булевых переменных одинаковой длины n. Это количество признаков, описывающих данные. Для заданного набора Y булевых данных ($Y \subseteq B^n$), разделенных на два непересекающихся подмножества положительных и отрицательных данных, соответственно, Y^+ и Y^-, функция, определенной как:

$$h : Y \to B, \qquad h(x) = \begin{cases} 1, \text{ если } x \in Y^+, \\ 0, \text{ если } x \in Y^-, \end{cases}$$

является *частично определенной булевой функцией* на Y. Для заданных двух булевых функций h и h', h' считается согласованной с h, если:

$$h(x) = 1 \Rightarrow h'(x) = 1,$$
$$h(x) = 0 \Rightarrow h'(x) = 0,$$

то есть все истинные (соответственно ложные) векторы h также являются истинными (соответственно ложными) векторами h'. Каждая *полностью определенная булева функция* h' (т.е. определенная на всем множестве B^n), которая согласуется с частично определенной логической функцией h, является *расширением* h.

Терм является конъюнкцией литералов, а *степень* G_t терма t является количеством литералов, которые он содержит. Говорят, что терм t *покрывает* булев вектор x, если каждому литералу в t удовлетворяет соответствующее значение в x, и это обозначается как $t(x) = 1$. *Характеристический терм* булева вектора x - терм степени $G = |x|$, покрывающий x. Например, $\bar{b}_1 b_2 b_3 \bar{b}_4$ является характеристическим термом вектора (0, 1, 1, 0). *Покрытие* терма соответствует количеству векторов, которые он покрывает. Естественно следует, что покрытие характеристического терма некоторого вектора x равно 1, поскольку x является единственным вектором, который он покрывает. Фактически, все термы степени, равные количеству булевых переменных, могут иметь покрытие не более 1. Когда степень терма уменьшается, он становится более общим, и вероятность того, что он покроет больше векторов, возрастает.

2.3 Анализ разнотипных признаков

Исследуемый подход к распознаванию, основанный на выявлении в данных правил, изначально разработан для распознавания объектов, признаки которых могут принимать лишь два допустимых значения, то есть являются бинарными. Для задач распознавания, в которых объекты описываются вещественными или какими-либо другими признаками, необходимо применить процедуру бинаризации, которая должна быть неотъемлемым этапом при построении классификатора.

В этой главе рассматривается задача преобразования набора данных, представленных в произвольном числовом формате, в булевый вид. Такая подготовка входных данных необходима для дальнейшего выявления в них логических закономерностей, что предполагает, что все примеры представлены векторами булевых значений.

Применимость результатов, представленных здесь, не ограничивается рассматриваемым подходом, поскольку некоторые другие методы классификации

также используют булевы концепты. Это относится к подходам на основе правил и деревьев решений. Большинство существующих методов либо бинаризуют исходные признаки во время построения модели по мере необходимости, либо определяют расширенную логическую версию исходного набора на основе всех возможных комбинаций входных признаков. Очевидно, что последний подход может применяться только к признакам, принимающим дискретный набор значений, и плохо масштабируется при увеличении размера задачи, а именно количества признаков и количества различных значений, которые они могут принимать.

Логический анализ данных изначально был разработан для анализа наборов данных, признаки которых принимают только двоичные значения. Однако во многих реальных задачах используются данные количественные (например, температура, вес и т.д.), а также номинальные (цвет, форма и т.д.). Чтобы сделать такие наборы данных пригодными для выявления закономерностей предлагаемым методом, данные должны быть преобразованы в двоичный формат. Процедура реализации этого преобразования была предложена в [71] и получила название бинаризации.

Задача, рассматриваемая здесь, может быть сформулирована как определение отображения $M\colon \Omega \to B^n$ из произвольного входного пространства $\Omega \subset R^t$ в бинарное выходное пространство B^n на основе набора данных $X \subset \Omega$, разделенных на классы. Кодирование данных таким способом обычно подразумевает потерю информации. Поэтому отображение должно быть определено таким образом, чтобы определенные фундаментальные свойства данных сохранялись во время преобразования. В то же время размер n логических отображений должен быть как можно меньше по причинам, связанным как с вычислительной сложностью процедур, использующих выходные данные, так и с интерпретируемостью модели, которая будет сгенерирована из этих данных. Это создает проблему оптимизации, рассматриваемую далее.

Признаки входного пространства могут быть любого типа: бинарные, дискретные упорядоченные или неупорядоченные и непрерывные. Одним из

факторов, определяющих интерпретируемость модели классификации, является то, что каждый литерал конъюнкции (терм) связан с одним исходным признаком. Если представить отображение M как состоящее из n булевых функций $\Omega \to B$, то каждая из этих функций должна включать только один исходный признак входного пространства Ω. Такие функции называются *дискриминантами*. Каждое булево отображение M соответствует некоторому множеству дискриминантов. Также существует взаимно-однозначное соответствие между каждым дискриминантом и одной из булевых переменных, которые будут использоваться в качестве описательных данных для классификации.

На рисунке 2.1 показан пример [1] преобразования данных в логический формат с использованием отображения M. Как видно, одному исходному количественному признаку может соответствовать несколько дискриминантов (булевых признаков).

gender	marital_ status	education	age
F	married	high_school	56
F	single	bachelor	26
M	married	bachelor	40

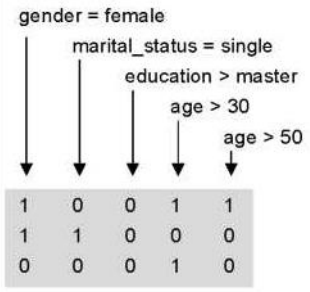


Рисунок 2.1 - Преобразование данных в булевый формат.

2.3.1 Основные типы признаков

Ниже приводится описание основных типов признаков, а также кодирование их в соответствующие бинарные (дискриминанты). Эти типы покрывают большинство признаков, встречающихся в обычных практических приложениях.

Бинарные (двоичные) признаки. Эти признаки могут принимать только два возможных значения и поэтому естественно соответствуют булевым переменным. Примеры:

• пол: женский, мужской;

• курящий: да, нет;

• водительские права: да, нет.

Бинарные признаки могут быть закодированы непосредственно как логические значения, даже если они не являются логическими исходно, как, например, пол. Дискриминанты имеют тип [признак = значение], и обычно один отдельный дискриминант создается из каждого бинарного признака.

Дискретные неупорядоченные признаки. Признаки этого типа могут принимать значения в предопределенном неупорядоченном наборе. Примеры:

• цвет: красный, зеленый, синий, желтый;

• материал: дерево, камень, стекло, железо;

• национальность: ...

Дискриминанты имеют тип [признак = значение], и их число равно количеству различных значений признака.

Дискретные упорядоченные признаки. Этот вид признака может принимать значения в предопределенном упорядоченном наборе. Заметим, что, несмотря на существование порядка между различными значениями, не обязательно существует мера расстояния. Примеры:

• уровень образования: средняя школа, бакалавр, магистр, аспирант;

• воинское звание: рядовой, сержант, лейтенант, капитан, майор, полковник, генерал;

• частота: никогда, редко, иногда, часто, всегда;

• качество: плохое, нормальное, хорошее, отличное.

Дискриминанты имеют тип [признак > значение], и число дискриминантов равно числу возможных различных значений признака (за вычетом одного).

Количественные признаки. Значения этих признаков имеют как порядок, так и меру расстояния. Большинство признаков, встречающихся в практических приложениях, являются количественными.

• температура, возраст, высота над уровнем моря, стоимость, количество, дата. и т.п.

Дискриминанты имеют тип [признак > значение] для любого возможного значения, которое может принимать признак.

Таким образом, дискриминанты, связанные с упорядоченными признаками, являются пороговыми значениями, применяемыми к диапазону возможных значений, тогда как дискриминанты неупорядоченных признаков являются селективными, поскольку они выбирают конкретное значение признака. В итоге: неупорядоченные признаки – бинарные и дискретные, неупорядоченные признаки – дискретные и количественные.

2.3.2 Геометрическая интерпретация дискриминантов

Рассматривая только количественные признаки, входное пространство Ω можно представить как евклидово гиперпространство с количеством измерений равным количеству исходных признаков. Так, дискриминант d количественного признака a представляет гиперплоскость, пересекающую ось a на соответствующем пороге, перпендикулярном оси. Таким образом, d разделяет все входное пространство на два подпространства, где примеры, расположенные в «нижнем» подпространстве или на самой гиперплоскости, имеют значение 0 (ложь) для дискриминанта d, а те, которые расположены в «верхнем» подпространстве, имеют значение 1 (истина). Подобная интерпретация действительна и для дискретных упорядоченных признаков, хотя в этом случае входное пространство разрезается на «кусочки», каждый из которых соответствует одному значению признака. Однако, те же рассуждения не применимы к неупорядоченным признакам. Фактически, из-за отсутствия

порядка, они не так хорошо интерпретируются как измерения евклидова пространства.

2.3.3 Согласованность булева отображения

Простая последовательность. Отображение M создается с использованием набора обучающих данных X. Одним из возможных решений задачи является создание максимального отображения, то есть составленного из максимального числа дискриминантов, которые могут быть получены из признаков, описывающих данные. Это решение может содержать слишком много дискриминантов не только с точки зрения сложности процедуры генерации модели, но также потому, что часть из них не обязательно может быть полезной. Тем не менее, это может быть отправной точкой для процедуры сокращения, направленной на поиск минимального набора дискриминантов, который по-прежнему сохраняет полезную информацию, содержащуюся в данных.

Поскольку рассматривается отображение для целей классификации, фундаментальным свойством, которое должно быть сохранено отображением M, является отделимость классов. Таким образом, определим согласованность отображения следующим образом: для каждых двух экземпляров данных x и x', таких что $F(x) \neq F(x')$, должно выполняться условие $M(x) \neq M(x')$, где F представляет собой неизвестную классификационную функцию. Это означает, что экземпляры, принадлежащие разным классам в исходном пространстве, также должны иметь разные логические изображения.

С точки зрения согласованности, может быть определено максимальное отображение, включающее для каждого признака a следующие дискриминанты:

• если a – бинарный, то создается дискриминант [a = значение], если есть два примера $x,x' \in X$, таких что $F(x) \neq F(x')$, x_a = значение и $x'_a \neq$ значение;

• если a – дискретный неупорядоченный, дискриминант [a = значение] создается для каждого значения, для которого существуют два примера $x,x' \in X$, такие что $F(x) \neq F(x')$, x_a = значение и $x'_a \neq$ значение;

• если a –дискретный упорядоченный, то создается дискриминант [$a>$ значение] для каждого значения при тех же условиях, что и для неупорядоченных признаков, за исключением самого высокого из этих значений;

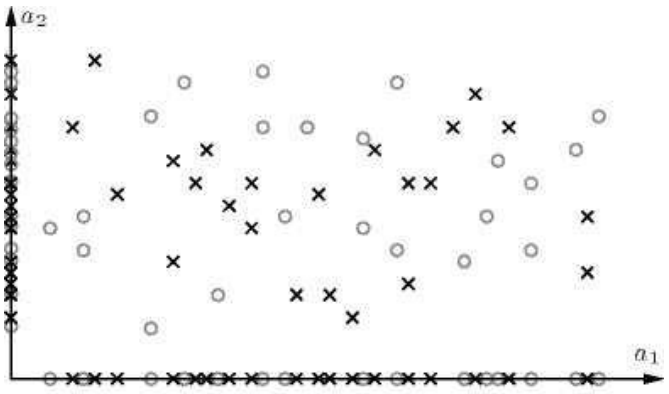
• если a – количественный, то дискриминант [$a>$ значение] создается для каждого значения $v = (v_i + v_j)/2$ при выполнении следующего условия: существует два примера $x,x' \in X$, таких что $F(x) \neq F(x')$, $x_a = v_i$ и $x'_a = v_j$, и нет примера $x'' \in X$, такого что $x_a < x'' < x'_a$, то есть x и x'являются последовательными по признаку a.

Этот подход естественным образом может быть применен к задачам с любым количеством классов.

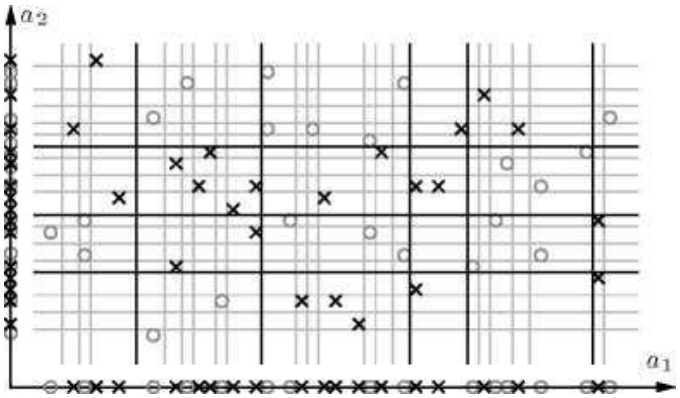
На рисунке 2.2 приведен пример [1]: (а) показывает набор данных, состоящий из наблюдений двух классов, описываемых двумя количественными признаками; (б) содержит набор дискриминантов максимального отображения; (в) содержит набор дискриминантов возможного минимального согласованного отображения. Понятие согласованности выражается в том факте, каждый «прямоугольник» содержит экземпляры только одного класса.

Повышенная согласованность. Описанное выше понятие относится к простой согласованности, которая означает, что каждая пара примеров из разных классов в обучающих данных отличается, по крайней мере, одним дискриминантом. Это означает, что их логические изображения должны отличаться как минимум одним значением. На практике может быть полезно использовать повышенную согласованность. Для этой цели ограничение согласованности можно естественным образом расширить, установив, что отображение M является m-согласованным с набором обучающих данных X тогда и только тогда, когда для любых двух примеров $x,x' \in X$, таких что $F(x) \neq F(x')$ расстояние Хэмминга между $M(x)$ и $M(x')$ составляет не менее m. Из этого

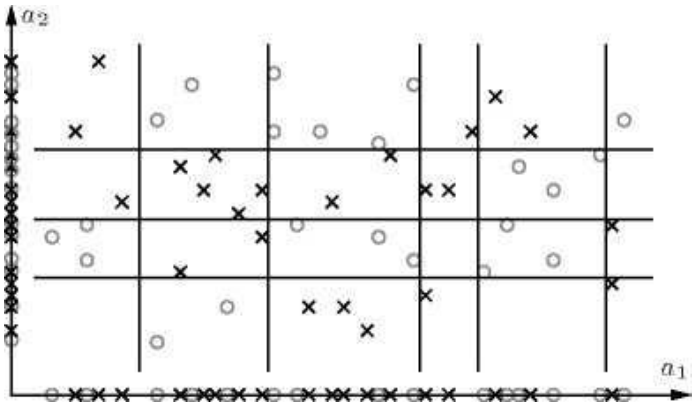
следует, что 1-согласованность идентична простой согласованности. Целью использования согласованности более высокого порядка является повышение устойчивости, хотя при этом размер отображений также возрастает.



(а) набор данных



(б) дискриминанты максимального отображения



(в) дискриминанты минимального согласованного отображения

Рисунок 2.2 – Пример дискриминантов

Следует уточнить, что используемое здесь понятие *m*-согласованности является распределенным. Это означает, что каждая пара экземпляров из разных классов должна быть разделена как минимум по *m* разным исходным признакам. Это подразумевает, что уровень согласованности не может быть выше, чем число исходных признаков.

Другая проблема, которую необходимо учитывать, заключается в том, что уровни согласованности выше единицы не могут быть гарантированы с самого начала для всех пар примеров из разных классов в наборе данных, по меньшей мере, используя процедуру кодирования дискриминанта, описанную выше. Рассмотрим, например, случай, когда два экземпляра *x* и *x*' из разных классов имеют одинаковые значения признаков, за исключением некоторого определенного дискретного признака. Учитывая, что дискретные признаки кодируются с помощью одного бита на значение в начальном отображении, булевы отображения *M* (*x*) и *M* (*x*') не могут различаться более чем на два бита. Таким образом, требуемый уровень согласованности может применяться только к парам примеров, удовлетворяющим ему уже в начальном максимальном отображении.

2.3.4 Минимизация числа дискриминантов

Целью настоящего исследования является определение процедуры для нахождения минимального набора *D* дискриминантов, согласующихся с данным набором данных *X*, и позволяющего преобразовать данные *X* в соответствующий набор *Y* в логическом формате. В дальнейшем рассматривается *m*-согласованность как для $m = 1$ (простая), так и для $m > 1$.

В большинстве практических задач классификации данные описываются разнотипными признаками, а не только бинарными. Рассмотрим процедуру бинаризации для преобразования количественного признака в один или несколько бинарных признаков, пытаясь сохранить информацию, доступную в исходном

признаке. Описанный алгоритм бинаризации основан на алгоритме, введенном в [71].

Простейшие небинарные признаки - это так называемые «номинальные» (или описательные) признаки. Типичным номинальным признаком является «форма», значения которого могут быть «круглая», «треугольная», «прямоугольная» и т.д. Бинаризация номинального признака x может быть выполнена следующим образом. Пусть $\{v1, ..., v_k\}$ будет множеством всех возможных значений x, которые появляются в наборе данных. Очевидно, это множество конечно, так как число наблюдений в наборе данных конечно. Каждому значению v_i из x мы сопоставляем булеву переменную $\alpha(x, v_i)$ такую, что

$$\alpha(x, v_i) = \begin{cases} 1, \text{ если } x = v_i, \\ 0, \text{ в противном случае.} \end{cases}$$

Бинаризация количественных признаков отличается и основана на понятии критических значений или порогов. При заданном наборе порогов для количественного признака x бинаризация состоит в использовании для каждого порога t булевой переменной x_t такой, что

$$x_t = \begin{cases} 1, \text{ если } x \geq t, \\ 0, \text{ если } x < t. \end{cases}$$

Новая переменная x_t называется индикаторной переменной.

Пример [72]. Рассмотрим таблицу 2.1, содержащую набор T положительных наблюдений и набор F отрицательных наблюдений и имеющие три числовых атрибута x, y, z.

Мы вводим порог 3.0 для признака x, порог 2.0 для признака y и порог 3.0 для признака z. Это преобразует числовые признаки x, y, z в переменные

логического индикатора $x_{3.0}$, $y_{2.0}$, $z_{3.0}$. Результат такой бинаризации приведен в таблице 2.2.

Таблица 2.1 - Количественные данные

Признаки	*x*	*y*	*z*
T: Положительные наблюдения	3.5	3.8	2.8
	2.6	1.6	5.2
	1.0	2.1	3.8
F: Отрицательные наблюдения	3.5	1.6	3.8
	2.3	2.1	1.0

Хотя в приведенном выше примере для каждого признака был введен ровно один порог, нет никаких оснований запрещать введение для некоторых признаков более одного порога или наоборот ни одного. В качестве другого примера, введем порог 2.0 для признака *y*, два порога 5.0 и 1.5 для признака *z*, и ни одного порога для признака *x*. Результат этой бинаризации показан в таблице 2.3 (следует отметить, что в этом случае первый и третий векторы *T* совпадают).

Таблица 2.2 - Бинаризация данных

Бинарные переменные	$x_{3.0}$	$y_{2.0}$	$z_{3.0}$
T: Положительные наблюдения	1	1	0
	0	0	1
	0	1	1
F: Отрицательные наблюдения	1	0	1
	0	1	0

В некоторых случаях выбор порогов определяется характером признаков. Например, многие медицинские параметры, такие как температура тела,

артериальное давление, частота пульса, уровень холестерина, называются «нормальными» или «ненормальными» в зависимости от того, находятся ли они внутри или вне определенного интервала, выше или ниже определенного порога. В тех случаях, когда «критические» значения признака неизвестны, применяется следующая процедура назначения порогов.

Таблица 2.3 - Второй вариант бинаризации данных

Бинарные переменные	$y_{2.0}$	$z_{5.0}$	$z_{1.5}$
T: Положительные наблюдения	1	0	1
	0	1	1
	1	0	1
F: Отрицательные наблюдения	0	0	1
	1	0	0

Пусть x будет количественным признаком. Поскольку число наблюдений в наборе данных конечно, x может принимать только конечное число различных значений в наборе. Пусть $v_1 < v_2 < ... < v_k$ - эти значения. Ясно, что если мы введем два порога между последовательными значениями x, то соответствующие булевы переменные будут идентичны. Следовательно, достаточно использовать не более одного порога между любыми двумя последовательными значениями x. Кроме того, пороги следует выбирать таким образом, чтобы можно было различать положительные и отрицательные наблюдения. Поэтому в порогах ниже v_1 или выше v_k нет надобности. Следовательно, можно ограничиться порогами вида $1/2\ (v_{i-1} + v_i)$. Порог $1/2\ (v_{i-1} + v_i)$ является существенным, если существуют положительное и отрицательное наблюдения, для одного из которых значение x равно v_{i-1}, а для другого $x = v_i$. Очевидно, что в процедуре бинаризации достаточно использовать только существенные пороги.

Применяя эти обозначения, теперь мы имеем двоичное описание каждой точки Ω в терминах индикаторных переменных. Логическое обоснование процесса бинаризации состоит в том, что точки в Ω, которые значительно отличаются по исходным количественным признакам, должны также существенно отличаться в значениях индикаторных переменных.

В целом, набор порогов, генерируемых описанным выше способом, может быть очень большим. Однако часто оказывается, что только небольшого подмножества порогов достаточно, чтобы отобразить Ω в двоичное представление, где различия между существенно различными точками сохраняются. Результаты последующих глав показывают, что общая вычислительная производительность алгоритмов выявления закономерностей напрямую связана с количеством фактически используемых индикаторных переменных. В связи с этим возникает задача поиска минимального набора порогов, который все еще сохраняет исходную информацию о наблюдениях различных классов.

2.4 Выбор порогов при бинаризации количественных признаков

Наибольшую сложность представляет кодирование количественных признаков. Так как число различных значений некоторого количественного признака всех объектов выборки может быть велико, наиболее популярным и обоснованным подходом является дискретизация количественного признака, при которой весь диапазон значений разбивается на интервалы, и каждое значение может относиться только к одному интервалу. Граничные значения этих интервалов называются порогами.

Очевидно, что от выбора числа интервалов и расположений порогов зависит работа всего метода распознавания и качество распознавания.

Исходным критерием при дискретизации является качество распознавания объектов, а именно точность и интерпретируемость. Но оценить точность распознавания можно только после прохождения всех этапов построения

классификатора, поэтому применить этот критерий при дискретизации не представляется возможным.

Наиболее подходящими критериями, которые можно вычислить непосредственно в процессе бинаризации (а не в результате распознавания контрольных наблюдений), являются следующие.

1. Различимость объектов разных классов. Если два объекта разных классов в пространстве исходных признаков различались друг от друга значениями метрических признаков, то после дискретизации этих признаков может оказаться, что эти объекты совпадут, то есть будут иметь одни и те же значения бинаризованных признаков, что может негативно отразиться на качестве распознавания.

2. Минимум интервалов дискретизации. Чем меньше будет использовано порогов при дискретизации, тем меньше будет число бинарных признаков, что снизит размерность задачи и, возможно, позволит её решить более успешно. Кроме того, меньшее число интервалов положительно сказывается на интерпретируемости результатов распознавания.

3. Робастность дискретизации. В случае наличия ошибок в измерениях значений признаков может представлять проблему то, что эти значения находятся близко к порогам. Поэтому более робастными являются пороги, расстояние от которых до ближайших значений признака является наибольшим.

2.4.1 Способы кодирования действительной переменной

Рассмотрим основные применяемые способы кодирования [73]. В таблице 2.4 представлены «единичный» код, двоичный, а также модифицированный двоичный код.

Наиболее простым является «единичный» код. Считается, что он имеет два значимых недостатка:

1) Большая размерность. Для кодирования m разнозначных значений требуется использование m-1 бинарных переменных.

2) Большое число недопустимых комбинаций бинарных переменных, то есть которым не соответствуют точки в исходном пространстве. Число таких точек равно $2^{m-1} - m$.

Таблица 2.4 - Способы кодирования

Число	Единичный	Двоичный	Модифицированный
0	0	0	0
1	1	1	1
2	11	10	11
3	111	11	10
4	1111	100	110
5	11111	101	111
6	111111	110	101
7	1111111	111	100

Для двоичного кода требуется использование лишь $\lceil \log_2 m \rceil$ бинарных переменных. Недопустимые комбинации бинраных переменных также могут присутствовать, но их число значительно меньше. Недостатком двоичного кода является несоответствие расстояний в исходном и бинарном пространствах. Точки, близкие в исходном пространстве, могут быть далеки в бинаризованном (по Хеммингу).

В модифицированном коде этот недостаток частично устранен - близкие точки в исходном пространстве остаются близкими и в бинарном. Для этого кода расстояние по Хеммингу между соседними кодовыми комбинациями всегда равно 1. Но обратное утверждение не верно, так как близкие точки в бинарном пространстве часто бывают дальними в исходном.

Чтобы выбрать наиболее подходящий способ кодирования, нужно учитывать, в каких условиях будет использоваться этот код. Рассмотрим, к примеру, алгоритмы оптимизации, изначально предназначенные для бинарных переменных. Такими являются, в частности, генетические алгоритмы. Бинарные переменные в алгоритмах оптимизации являются «управляемыми» переменными, «входами», их различные комбинации назначаются в процессе работы алгоритма, и для каждой назначаемой комбинации требуется вычисление целевой функции (и ограничения). При этом число переменных может быть велико и зависит от желаемой погрешности в определении решения. Поэтому наилучшим выбором является использование (модифицированного) двоичного кода. Это позволяет использовать значительно меньшее число бинарных переменных (по сравнению с «единичным» способом); число недопустимых комбинаций (для которых отсутствуют точки в исходном пространстве) относительно невелико.

Теперь рассмотрим, какой способ кодирования следует выбрать для бинаризации признаков объектов при решении задачи распознавания. В этой задаче не происходит назначения значений переменных, они уже заданы и являются величинами, характеризующими объекты распознавания. Поэтому наличие большого числа недопустимых комбинаций не представляет никакой проблемы.

Предположим, что для некоторого признака b_j задано определенное число порогов $\beta_1^j, \beta_2^j, ..., \beta_{k_j}^j$. Простейшим способом кодирования представляется назначение каждому порогу β_i^j бинарной переменной x_i^j, такой что

$$x_i^j = \begin{cases} 1, & \text{если } b_j \geq \beta_i^j, \\ 0, & \text{в противном случае.} \end{cases}$$

Тогда каждая бинарная переменная будет иметь вполне конкретный смысл: если её значение равно единице, то значение соответствующего признака превышает некоторый заданный порог.

Например, $x_2^j = 0$ соответствует $b_j < \beta_2^j$. А $(x_1^j = 1) \wedge (x_3^j = 0)$ соответствует $\beta_1^j \le b_j < \beta_3^j$ (рисунок 2.3).

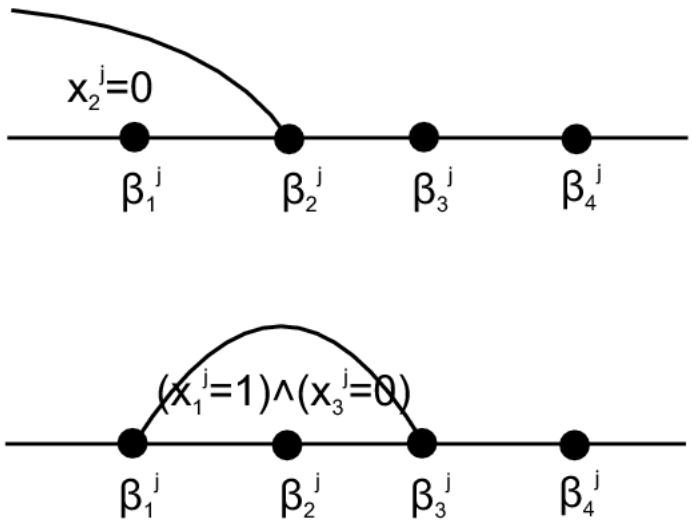


Рисунок 2.3 - Бинарные переменные и интервалы вещественной переменной

Это позволяет работать с интервалами значений исходного признака, причем интервал, задаваемый двумя бинарными переменными, не обязательно располагается между двумя последовательно расположенными порогами, а между любыми двумя выбранными порогами.

Для сравнения, при использовании двоичного кода выражение $x_1^j = 1$ соответствует множеству отдельных интервалов (рисунок 2.4). А задать интервал между произвольными двумя порогами можно лишь путем перечисления отдельных интервалов. Так, например, $\beta_1^j \le b_j < \beta_3^j$ будет соответствовать $\left((x_1^j = 1) \wedge (x_2^j = 0) \wedge (x_3^j = 0)\right) \vee \left((x_1^j = 0) \wedge (x_2^j = 1) \wedge (x_3^j = 0)\right)$ в случае использования двоичного кода. Это делает использование двоичного кода для бинаризации признаков объектов при распознавании крайне неудобным и необоснованным.

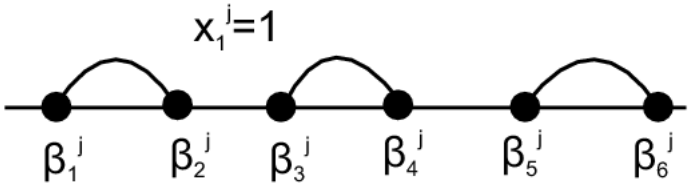


Рисунок 2.4 - Пример интервалов при двоичном кодировании

Затронем теперь вопрос размерности. Как показывает практика решения задач распознавания [92], число необходимых порогов для дискретизации одной вещественной переменной лежит в диапазоне от одного до четырёх. Так, использование четырёх порогов соответствуют пяти интервалам значений исходной переменной, что можно ещё легко интерпретировать, например: нормальное значение, высокое, очень высокое, низкое, очень низкое. Большее число порогов необходимо использовать лишь в исключительных случаях. И вполне возможно, что такие случаи возникают вследствие неудачного выбора расположения порогов. При использовании четырёх порогов требуется задать четыре бинарных переменных для «единичного» кодирования либо три переменных для двоичного. Как видно, разница в размерности не существенна и не может повлиять на выбор способа кодирования с учетом описанных выше особенностей.

Все рассматриваемые в дальнейшем подходы основаны на использовании «единичного» кода.

2.4.2 Оптимизационная модель для поиска наименьшего числа порогов

Известный способ выбора порогов при бинаризации действительных признаков состоит в решении задачи оптимизации [71]. Модель оптимизации направлена на снижение числа порогов, достаточных для разделения наблюдений различных классов.

$$\sum_{j=1}^{d}\sum_{i=1}^{k_i} y_{ij} \to \min_{Y},$$

$$\sum_{j=1}^{d}\sum_{i=1}^{k_i} a_{ij}^{zw} y_{ij} \geq 1, z \in K^{+}, w \in K^{-},$$

где $y_{ij} \in \{0,1\}$ - управляющая переменная, определяющая использование соответствующего порога, $a_{ij}^{zw} = \left|x_{ij}^{z} - x_{ij}^{w}\right|$ для $z \in K^{+}$ и $w \in K^{-}$.

При практическом использовании приведенной выше модели оптимизации выяснилось, что получаемые с помощью нее решения имеют недостатки. Во-первых, число порогов в найденном решении может быть слишком мало, что уменьшает разделяющую способность классификатора, который строится на основании этих порогов, и негативно влияет на точность распознавания. Для преодоления этого недостатка используется модификация - в правой части ограничений вместо единицы используется целое число $h \geq 1$. Но оказывается, что при назначении в правой части ограничений большего числа (то есть любые два объекта различных классов должны различаться более чем одним порогом) число порогов может быть не уменьшено вообще по отношении к исходному числу потенциальных порогов.

Во-вторых, в этой модели пороги всех признаков смешаны. В полученном решении для некоторого признака может оставаться большое число порогов, в то время как для многих других признаков пороги могут отсутствовать.

Поэтому часто оказывается, что приближенное решение этой задачи может быть предпочтительней, чем оптимальное, так как это позволяет добиться лучшего компромисса в балансе "вычислительная сложность / разделяющая способность".

В связи с этим возникла необходимость решать новую задачу: выбрать такое расположение приемлемого числа порогов, при котором объекты различных классов разделены как можно сильнее.

Пусть имеется два класса наблюдений K^+ и K^-: $K^+ \cup K^- = K$, $K^+ \cap K^- = \text{ø}$. Наблюдения описываются действительными признаками. Расположим всевозможные значения каждого признака j=1,…,d для наблюдений данной выборки в порядке возрастания: $b_1^{(j)} < b_2^{(j)} < b_3^{(j)} <\ldots$

Потенциальные пороги можно взять как середины между двумя ближайшими значениями признака:

$$\beta_i^j = \left(b_i^{(j)} + b_{i+1}^{(j)}\right)/2.$$

Очевидно, что в качестве потенциальных порогов имеет смысл брать только те (из выше указанных), для которых ближайшие значения признака с разных сторон от порога соответствуют наблюдениям разных классов, т.е. существуют $z \in K^+$ и $w \in K^-$ (или наоборот) такие, что $z_j = b_i^{(j)}$ и $w_j = b_{i+1}^{(j)}$.

Таким образом, для каждого действительного признака, описывающего объект распознавания, имеем упорядоченное по возрастанию множество потенциальных порогов $\beta_1^j, \beta_2^j, \ldots, \beta_{k_j}^j$.

Бинарные переменные указывают, превысило ли значение признака объекта соответствующий порог

$$x_{ij} = \begin{cases} 1, & b_j \geq \beta_i^j, \\ 0, & b_j < \beta_i^j. \end{cases}$$

Рассмотрим наблюдения разных классов $z \in K^+$ и $w \in K^-$, для которых определим величины

$$a_{ij}^{zw} = \left|x_{ij}^z - x_{ij}^w\right|.$$

Если $a_{ij}^{zw} = 1$, то наблюдения z и w различны по бинарной переменной x_{ij}, то есть значения численного признака b_j у этих наблюдений лежат по разные стороны порога β_i^j.

Введем бинарную переменную y_{ij}, указывающую, будет ли порог β_i^j использоваться при дискретизации

$$y_{ij} = \begin{cases} 1, & \text{если порог } \beta_i^j \text{ используется,} \\ 0, & \text{в противном случае.} \end{cases}$$

Предлагается оптимизационная модель для выбора таких порогов, при которых объекты разных классов оказываются наиболее различимы, а число самих порогов ограничено. В качестве критерия при выборе порогов примем суммарное число признаков, по которым объекты всевозможных пар различаются между собой. При этом любые два объекта должны быть различны хотя бы по одному признаку. Кроме того, следует ограничить число применяемых для каждого численного признака порогов.

Рассмотрим пороги некоторого численного признака b_j. Чтобы объекты $z \in K^+$ и $w \in K^-$ были различны по этому признаку после бинаризации, необходимо, чтобы хотя бы для одного порога β_i^j, для которого значения признака b_j объектов z и w находятся по разные стороны (то есть $a_{ij}^{zw} = 1$), значение решающей переменной y_{ij} было равно 1. При этом значение выражения

$$\prod_{i=1}^{k_j} (1 - a_{ij}^{zw} y_{ij})$$

будет равно 0. Если же это выражение будет равно 1, то это значит, что объекты z и w оказались неразличимы по признаку b_j после бинаризации.

Таким образом, можно вычислить число (действительных) признаков, по которым объекты z и w различаются после бинаризации

$$B^{zw} = \sum_{j=1}^{d} (1 - \prod_{i=1}^{k_j} (1 - a_{ij}^{zw} y_{ij})).$$

В качестве критерия при выборе порогов примем суммарное число признаков, по которым объекты всевозможных пар $(z,w) \in K^+ \times K^-$ различаются между собой

$$\sum_{z,w} B^{zw} \to \max.$$

При этом любые два объекта $z \in K^+$ и $w \in K^-$ должны быть различны хотя бы по одному признаку

$$\sum_{j=1}^{d} \sum_{i=1}^{k_j} a_{ij}^{zw} y_{ij} \geq 1.$$

Кроме того, следует ограничить число применяемых для каждого численного признака b_j порогов

$$\sum_{i=1}^{k_j} y_{ij} \leq h,$$

где h - наибольшее заданное число порогов для каждого численного признака.

Таким образом, получаем следующую задачу оптимизации:

$$\sum_{(z,w)\in(K^+)\times(K^-)} \sum_{j=1}^{d} \left(1 - \prod_{i=1}^{k_j} \left(1 - a_{ij}^{zw} y_{ij}\right)\right) \to \max_{Y},$$

$$\sum_{j=1}^{d} \sum_{i=1}^{k_i} a_{ij}^{zw} y_{ij} \geq 1, z \in K^+, w \in K^-,$$

$$\sum_{i=1}^{k_j} y_{ij} \leq h, j = 1, \dots, d,$$

$$y_{ij} \in \{0,1\}, i = 1, \dots, k_j, j = 1, \dots, d,$$

где d – число вещественных признаков,

k_j – число потенциальных порогов для j-го признака,

h – наибольшее заданное число порогов для каждого численного признака.

Выводы к главе 2

Разработана новая модель оптимизации для нахождения оптимального назначения порогов количественных признаков, которая не просто определяет наименьшее число порогов, достаточных для разделения наблюдений разных классов, а позволяет выбрать такие пороги, которые наилучшим образом разделяют наблюдения разных классов в пространстве булевых признаков. Предлагаемая модель обеспечивает нахождение оптимального размещения порогов вещественных признаков, при котором наблюдения разных классов являются наиболее различимыми при заданном максимальном числе порогов, что приводит к повышению покрытий отдельных закономерностей и повышению точности классификатора.

ГЛАВА 3. МОДЕЛИ ОПТИМИЗАЦИИ ДЛЯ ВЫЯВЛЕНИЯ ЗАКОНОМЕРНОСТЕЙ

3.1 Закономерности в данных

Рассмотрим задачу распознавания объектов, описываемых бинарными признаками и разделенных на два класса $K = K^+ \cup K^- \subset B_2^n$, где $B_2^n = \{0,1\}^n$, $B_2^n = B_2 \times B_2 \times \cdots \times B_2$ [74]. При этом классы не пересекаются: $K^+ \cap K^- = \emptyset$.

Наблюдение $X \in K$ описывается бинарным вектором $X = (x_1, x_2, \ldots, x_n)$ и может быть представлен как точка в гиперкубе пространства бинарных признаков B_2^n. Наблюдения класса K^+ будем называть положительными точками выборки K, а наблюдения класса K^- - отрицательными точками выборки.

Рассмотрим подмножество точек из B_2^n, у которых некоторые переменные фиксированы и одинаковы, а остальные принимают произвольное значение [75]:

$$T = \{x \in B_2^n \,|\, x_i = 1 \text{ для } \forall i \in A \text{ и } x_j = 0 \text{ для } \forall j \in B\},$$

для некоторых подмножеств $A, B \subseteq \{1,2,\ldots,n\}$, $A \cap B = \emptyset$. Это множество может быть также определено в виде булевой функции, принимающей истинное значение для элементов множества: $t(x) = (\underset{i \in A}{\wedge} x_i) \wedge (\underset{j \in B}{\wedge} \bar{x}_j)$.

Множество точек x, для которых $t(x) = 1$, обозначим $S(t)$. $S(t)$ является подкубом в булевом гиперкубе B_2^n, число точек подкуба равно $2^{(n-|A|-|B|)}$.

Бинарная переменная x_i или её отрицание $\bar{x}_i$ в терме называется литералом. Запись x_i^α обозначает x_i, если $\alpha = 1$, и $\bar{x}_i$, если $\alpha = 0$. Таким образом, терм представляет собой конъюнкцию различных литералов, которая не содержит одновременно некоторую переменную и ее отрицание. Множество литералов в терме t обозначим $Lit(t)$.

Будем считать, что терм t покрывает точку $a \in B_2^n$, если $t(a) = 1$, то есть эта точка принадлежит соответствующему подкубу.

Основное понятие в логическом анализе данных — это понятие закономерности. Положительная закономерность — это просто подкуб целого гиперкуба, который пересекается с K^+ и не пересекается с K^- [76]. Отрицательные закономерности имеют аналогичное определение.

Другими словами, под *закономерностью P* (или *правилом*) понимается терм, который покрывает хотя бы одно наблюдение некоторого класса и не покрывает ни одного наблюдения другого класса. То есть закономерность соответствует подкубу, имеющему непустое пересечение с одним из множеств (K^+ или K^-) и пустое пересечение с другим множеством (K^- или K^+ соответственно). Закономерность P, которая не пересекается с K^-, будем называть положительной, а закономерность P', которая не пересекается с K^+ - отрицательной.

Более формально [76], терм С называется положительной (отрицательной) закономерностью набора данных (K^+, K^-), если

1. C (w) = 0 для каждого $w \in K^-$ ($w \in K^+$), и

2. C (w) = 1 хотя бы для одного вектора $w \in K^+$ ($w \in K^-$).

Множество наблюдений, которые покрываются закономерностью P, обозначим $Cov(P)$. Закономерности являются элементарными блоками для построения логических решающих функций.

Пример 3.1 [76]. Рассмотрим набор бинарных данных, приведенный в таблице 3.1. В этой таблице *a, b, c, d* и *e* - положительные наблюдения, *a p, q, r, s* и *t* - отрицательные наблюдения. Например, можно проверить, что $x_1x_2x_3$ - это положительная закономерность, а $x_1\bar{x}_2\bar{x}_3$ - отрицательная. Также видно, что $x_1\bar{x}_2$ не является ни положительной, ни отрицательной закономерностью, поскольку охватывает как положительные, так и отрицательные наблюдения. С другой стороны, $x_1x_2\bar{x}_3$ не является ни положительной, ни отрицательной закономерностью, поскольку не охватывает ни одного наблюдения.

Таблица 3.1 - Бинарные данные (K^+, K^-)

		x1	*x*2	*x*3	*x*4	*x*5
K^+	*a*	1	0	1	1	1
	b	0	0	0	1	1
	c	1	1	1	1	1
	d	1	1	1	0	1
	e	1	1	1	0	0
	p	1	0	0	1	0
	q	0	0	1	0	1
K^{--}	*r*	1	0	1	0	0
	s	1	0	0	0	0
	t	0	0	1	0	0

Так как термы геометрически интерпретируются как подкубы n-мерного куба $\{0,1\}^n$, то положительные (отрицательные) закономерности соответствуют тем подкубам, которые пересекаются с множеством K^+ (соответственно, K^{--}), но не пересекаются с множеством K^{--} (соответственно, K^+). Рассмотрим снова Пример 3.1. Терм $C = \bar{x}_1\bar{x}_3x_4$ является положительной закономерностью. Множество тех точек, где C принимает значение 1, то есть точек, для которых $x1 = 0$, $x3 = 0$, $x4 = 1$, является подкубом $Q = \{(01011), (00011), (01010), (00010)\}$, Обратим внимание, что $Q \cap K^- = \varnothing$. Всякий раз, когда это не вызывает путаницы, мы будем ссылаться на термы и соответствующие подкубы взаимозаменяемо.

Поскольку свойства положительных и отрицательных закономерностей являются полностью симметричными, без потери общности сосредоточимся в этом разделе на положительных закономерностях и часто будем называть положительные закономерности просто закономерностями.

Поскольку закономерности играют центральную роль в LAD, были изучены различные типы закономерностей (например, первичные, охватывающие, максимальные), были разработаны алгоритмы для их перечисления [77-79] и проанализирована их относительная эффективность [76].

К сожалению, не существует единственного однозначного критерия для сравнения логических закономерностей между собой. При анализе различных данных к качеству и особенностям формируемых закономерностей могут предъявляться разные требования. В соответствие с работой [76] для оценки качества чистых (однородных, не покрывающих наблюдений других классов) закономерностей используем три отношения частичного предпорядка — простота, избирательность и доказательность, а также их возможные совмещения.

Напомним, что бинарное отношение ρ, определенное на конечном множестве *S*, называется отношением частичным предпорядка, если оно

1. рефлексивно, то есть *x*ρ*x* для любого *x* ϵ *S* выполняется *x*ρ*x*;

2. транзитивно, то есть для любых *x*, *y*, *z* ϵ *S*, если выполняются *x*ρ*y* и *y*ρ*z*, то также выполняется *x*ρ*z*.

Отношение частичного предпорядка называется отношением частичным порядка, если оно

3. антисимметрично, то есть для любых *x*, *y* ϵ *S*, если выполняется *x*ρ*y* и *y*ρ*x*, то *x* = *y*.

Частичный предпорядок ρ' называется уточнением частичного предпорядка ρ, если для любых *x*, *y* ϵ *S* из отношения *x*ρ*y* следует отношение *x*ρ'*y*.

Отношение простоты (или компактности) часто используется для сравнения закономерностей между собой, в том числе получаемых разными алгоритмами обучения. Закономерность P_1 предпочтительнее P_2 по отношению *простоты* (обозначим $P_1 \succeq_\sigma P_2$), если $Lit(P_1) \subseteq Lit(P_2)$.

Закономерность *P* является *первичной*, если после удаления любого литерала из *Lit*(*P*) образуется терм, который не является (чистой) закономерностью (то есть покрывает наблюдения другого класса). Очевидно, что

оптимальность закономерности по отношению простоты тождественна утверждению, что эта закономерность является первичной.

Пример 3.2. В наборе бинарных данных, приведенном в таблице 3.1, можно указать следующие первичные закономерности:

$$x_2, \bar{x}_1\bar{x}_3, \bar{x}_1x_4, x_1x_5, x_3x_4, \bar{x}_3x_5, x_4x_5.$$

С другой стороны, закономерность x_1x_2 предоставляет пример непервичной закономерности. □

Поиск более простых закономерностей имеет вполне обоснованные предпосылки. Во-первых, такие закономерности являются лучше интерпретируемыми и понятными для человека при их использовании при принятии решения. Во-вторых, часто считается, что более простые закономерности имеют лучшую обобщающую способность, и их использование приводит к лучшей точности распознавания. Однако, это утверждение спорно, более того, было показано, что уменьшение простоты может приводить к большей точности.

Использование простых, а значит, коротких закономерностей приводит к тому, что уменьшается число неверно распознанных положительных наблюдений (ошибочных отрицательных), но в то же время это может приводить к увеличению числа неверно распознанных отрицательных наблюдений (ошибочных положительных). Естественный путь к уменьшению числа ошибочных положительных наблюдений — это формирование более избирательных наблюдений, что достигается уменьшением размера подкуба, определяющего закономерность.

Закономерность P_1 предпочтительнее P_2 по отношению *избирательности* (обозначим $P_1 \succeq_\Sigma P_2$), если $S(P_1) \subseteq S(P_2)$.

Следует отметить, что два рассмотренных выше отношения противоположны друг другу, то есть $Lit(P_1) \subseteq Lit(P_2) \Leftrightarrow S(P_1) \supseteq S(P_2)$.

Закономерность, максимальная по отношению избирательности, является *минтермом*, то есть закономерностью, покрывающей единственное положительное наблюдение. Использование этого отношения самого по себе, естественно, не эффективно, так как получаемые закономерности (минтермы) не обладают какой-либо обобщающей способностью. Но отношение избирательности является чрезвычайно полезным при использовании совместно с другими отношениями, что будет рассмотрено далее.

Еще одним полезным отношением является отношение, основанное на покрытии $Cov(P)$ закономерности P, то есть множестве положительных наблюдений обучающей выборки $X \in K^+$, удовлетворяющих условиям закономерности $P(X)=1$. Несомненно, что закономерности с большим покрытием обладают большей обобщающей способностью. А наблюдения обучающей выборки, покрываемые закономерностью, являются своего рода доказательством применимости этой закономерности для принятия решения.

Но отметим следующий момент. Хотя отношение $|\ Cov\ (P_1)\ | > |\ Cov\ (P_2)\ |$ можно интерпретировать как означающее, что закономерность P_1 является более представительной, чем закономерность P_2, но фактически оно учитывает только количество элементов в двух множествах $Cov\ (P_1)$ и $Cov\ (P_2)$. Однако замена вышеупомянутого сравнения количества элементов в этих двух множествах на более сильное отношение, учитывающее сами элементы этих множеств, позволяет учитывать отдельные наблюдения, охватываемые этими двумя закономерностями. Наблюдения в $Cov\ (P)$ можно рассматривать как «совокупность доказательств», подтверждающих закономерность P.

Закономерность P_1 предпочтительнее P_2 по отношению *доказательности* (обозначим $P_1 \succeq_\varepsilon P_2$), если $Cov(P_1) \supseteq Cov(P_2)$. Закономерности, максимальные по отношению доказательности, называются *сильными*. То есть закономерность P является сильной в том случае, если не существует закономерности P' такой, что $Cov(P') \supset Cov(P)$.

Пример 3.3. Видно, что для набора бинарных данных, приведенного в таблице 3.1, закономерность x_3x_4 не является сильной. В наборе бинарных данных, приведенном в таблице 3.1, можно указать следующие сильные закономерности:

$$x_2,\ x_1x_2,\ x_2x_3,\ x_1x_2x_3,\ x_1x_5,\ x_1x_3x_5,\ x_4x_5.$$

Можно заметить, что, к примеру, закономерность x_3x_4 не является сильной, так как $Cov\ (x_4x_5) = \{a, b, c\} \supset \{a, c\} = Cov\ (x_3x_4)$. □

Важно отметить, что рассматриваемые отношения не являются полностью независимыми. Так, отношения простоты и избирательности противоположны друг другу. Более того, можно отметить следующие зависимости:

$$P_1 \succeq_\sigma P_2 \Rightarrow P_1 \succeq_\varepsilon P_2;$$
$$P_1 \succeq_\Sigma P_2 \Rightarrow P_2 \succeq_\varepsilon P_1.$$

Для любого отношения $\succeq$ одновременное выполнение соотношений $P_1 \succeq P_2$ и $P_2 \succeq P_1$ будет обозначаться как $P_1 \approx P_2$. Одновременное выполнение соотношений $P_1 \succeq P_2$ и $P_2 \not\succeq P_1$ будет обозначаться как $P_2 \succ P_1$.

Для того чтобы комбинировать рассмотренные отношения между собой, используются два способа их совмещения.

Для заданных критериев π и ρ закономерность P_1 предпочтительнее P_2 по пересечению $\pi \wedge \rho$ (обозначим $P_1 \succeq_{\pi\wedge\rho} P_2$), если и только если $P_1 \succeq_\pi P_2$ и $P_1 \succeq_\rho P_2$.

Для заданных критериев π и ρ закономерность P_1 предпочтительнее P_2 по лексикографическому уточнению $\pi \mid \rho$ (обозначим $P_1 \succeq_{\pi|\rho} P_2$), если и только если $P_1 \succ_\pi P_2$ или ($P_1 \approx_\pi P_2$ и $P_1 \succeq_\rho P_2$).

Следует отметить, что в то время как отношения $\pi \wedge \rho$ и $\rho \wedge \pi$ идентичны, отношения $\pi \mid \rho$ и $\rho \mid \pi$ обычно различны.

Поскольку каждое из представленных отношений выражает различные аспекты предпочтения закономерностей, представляется разумным

проанализировать различные их комбинации. Можно проверить, что единственными новыми отношениями, которые могут быть получены путем применения пересечения и лексикографического уточнения, являются $\Sigma \wedge \varepsilon$, $\varepsilon \mid \sigma$ и $\varepsilon \mid \Sigma$. Кратко рассмотрим эти три отношения далее.

Закономерности, максимальные по $\Sigma \wedge \varepsilon$, называются *охватывающими*. Закономерности, максимальные по $\varepsilon \mid \sigma$, называются *сильными первичными*. И закономерности, максимальные по $\varepsilon \mid \Sigma$, называются *сильными охватывающими*.

В таблице 3.2 приведены примеры, подчеркивающие существование закономерностей, имеющих различные комбинации свойств, описанных выше. Единственная комбинация свойств закономерностей, которая отсутствует в таблице: первичная, несильная, охватывающая. Эта комбинация невозможна, так как закономерность, которая является первичной и охватывающей, также должна быть сильной.

Таблица 3.2 - Закономерности для бинарных данных, представленных в таблице 3.1

Свойства закономерностей			Примеры закономерностей
Первичная	Сильная	Охватывающая	
нет	нет	нет	$\bar{x}_1\bar{x}_2\bar{x}_3$, $x_2x_3\bar{x}_4$, $x_3x_4x_5$
да	нет	нет	$\bar{x}_1\bar{x}_3$, $\bar{x}_1x_4$, x_3x_4, $x_3\bar{x}_5$
нет	да	нет	x_1x_2, x_2x_3
нет	нет	да	$x_1x_2x_3\bar{x}_4$, $\bar{x}_1\bar{x}_2\bar{x}_3x_4x_5$
да	да	нет	x_2, x_1x_5
нет	да	да	$x_1x_2x_3$, $x_1x_3x_5$
да	да	да	$x_1x_4x_5$

Из всех типов закономерностей, полученных в соответствие с рассмотренными выше отношениями и их комбинациями, наиболее полезными для выявления информативных закономерностей и их использования для

поддержки принятия решений при распознавании представляются закономерности первичные, сильные первичные и сильные охватывающие.

3.2 Поиск закономерности как задача оптимизации

Один из путей в составлении набора закономерностей для алгоритма распознавания – это поиск закономерностей, опирающихся на значения признаков конкретных объектов.

Выделим некоторое наблюдение $a \in K^+$. Обозначим P^a закономерность, покрывающую наблюдение *a*. Те переменные, которые зафиксированы в P^a, равны соответствующим значениям признаков объекта *a* [81].

Рассмотрим задачу поиска максимальной закономерности P^a, то есть такого терма, который помимо наблюдения *a* покрывает как можно больше положительных наблюдений и ни одного отрицательного.

Для задания закономерности P^a введем бинарные переменные $Y = (y_1, y_2, \ldots, y_n)$

$$y_j = \begin{cases} 1, & i\text{-ый признак фиксирован в } P^a, \\ 0, & \text{в противном случае.} \end{cases}$$

Некоторая точка $b \in K^+$ будет покрываться закономерностью P^a только в том случае, если $y_i = 0$ для всех i, для которых $b_i \neq a_i$. С другой стороны, некоторая точка $c \in K^-$ не будет покрываться закономерностью P^a в том случае, если $y_i = 1$ хотя бы для одной переменной i, для которой $c_i \neq a_i$.

Таким образом, задачу нахождения максимальной закономерности можно записать в виде задачи поиска таких значений $Y = (y_1, y_2, \ldots, y_n)$, при которых получаемая закономерность P^a покрывает как можно больше точек $b \in K^+$ и не покрывает ни одной точки $c \in K^-$ [79]:

$$\sum_{b\in K^+}\prod_{\substack{i=1\\ b_i\neq a_i}}^{n}(1-y_i)\to \max, \tag{3.1}$$

$$\sum_{\substack{i=1\\ c_i\neq a_i}}^{n} y_i \geq 1 \text{ для всех } c\in K^-. \tag{3.2}$$

Эта задача является задачей условной псевдобулевой оптимизации, то есть задачей, в которой целевая функция и левые части ограничений являются псевдобулевыми функциями – вещественными функциями булевых переменных. Целевая функция и функции ограничений в этой задаче унимодальны и монотонны.

Для поиска максимальных отрицательных закономерностей задача формулируется подобным образом.

Следует заметить, что любая точка $Y=(y_1,y_2,...,y_n)$ соответствует подкубу в пространстве булевых признаков $X=(x_1,x_2,...,x_n)$, включающему в себя базовое наблюдение *a*. При $Y\in O_k(Y^1)$ (то есть *Y* отличается от Y^1 значением *k* координат), где $Y^1=(1,1,...,1)$, число точек этого подкуба равно 2^k.

Целевая функция (3.1) является нелинейной. Bonates, Hammer, Kogan [79] рассматривают сведение задачи (3.1)-(3.2) к задаче целочисленного линейного программирования (ЦЛП). Но в результате сильно возрастает размерность задачи, и они отказываются от практического применения такого подхода и прибегают к эвристическим алгоритмам: жадному алгоритму и линейной аппроксимации целевой функции, позволяющей свести задачу к задаче ЦЛП, решение которой является приближенным решением исходной задачи.

3.3 Свойства задачи оптимизации

Рассмотрим свойства задачи оптимизации (3.1)-(3.2). Для этого, прежде всего, приведем основные понятия [82].

- Точки $X^1, X^2 \in B_2^n$ назовем *k-соседними*, если они отличаются значением k координат, $k = \overline{1,n}$.

- Множество $O_k(X)$, $k = \overline{0,n}$, всех точек B_2^n, k-соседних к точке X, назовем *k-м уровнем* точки X.

- Точку $X \in B_2^n$ назовем k-соседней множеству $A \subset B_2^n$, если $A \cap O_k(X) \neq \emptyset$ и $\forall l \equiv \overline{0,k-1}: A \cap O_l(X) = \emptyset$.

- Точку $X^* \in B_2^n$, для которой $f(X^*) < f(X), \forall X \in O_1(X^*)$, назовем *локальным минимумом* псевдобулевой функции *f*.

- Псевдобулевую функцию, имеющую только один локальный минимум, будем называть *унимодальной* на B_2^n функцией.

- Унимодальную функцию f назовем *монотонной* на B_2^n, если $\forall X^k \in O_k(X^*), k = \overline{1,n}: \quad f(X^{k-1}) \leq f(X^k), \forall X^{k-1} \in O_{k-1}(X^*) \cap O_1(X^k)$, и *строго монотонной*, если это условие выполняется со знаком строгого неравенства.

- Множество точек $W(X^0, X^l) = \{X^0, X^1, \ldots, X^l\} \subset B_2^n$ назовем *путем* между точками X^0 и X^l, если $\forall\, i = 1, \ldots, l$ точка X^i является соседней к X^{i-1}

- Путь $W(X^0, X^l) \subset B_2^n$ между k-соседними точками X^0 и X^l назовем *кратчайшим*, если $l = k$.

- $\forall X, Y \in B_2^n$ объединение всех кратчайших путей $W(X,Y)$ будем называть *подкубом* B_2^n и обозначать $K(X,Y)$.

Основываясь на приведенных понятиях, рассмотрим основные свойства множества допустимых решений задачи условной псевдобулевой оптимизации. Имеется задача следующего вида

$$C(X) \to \max_{X \in S \subset B_2^n}, \tag{3.3}$$

где $C(X)$ - монотонно возрастающая от X^0 псевдобулевая функция, $S \subset B_2^n$ - некоторая подобласть пространства булевых переменных, определяемая заданной системой ограничений, например: $A_j(X) \le H_j$, $j = \overline{1,m}$.

Для исследования свойств задачи введем ряд понятий для подмножества точек пространства булевых переменных.

- Точка $Y \in S$ является *граничной точкой* множества S, если существует $X \in O_1(Y)$, для которой $X \notin S$.
- Точку $Y \in O_i(X^0) \cap S$ будем называть *крайней точкой* множества S с базовой точкой $X^0 \in S$, если $\forall X \in O_1(Y) \bigcap O_{i+1}(X^0)$ выполняется $X \notin S$.
- Ограничение, определяющее подобласть пространства булевых переменных, будем называть *активным*, если оптимальное решение задачи условной оптимизации (3.3) не совпадает с оптимальным решением соответствующей задачи оптимизации без учета ограничения.

Ранее доказано следующее свойство множества допустимых решений [83]:

Если целевая функция является строго монотонной унимодальной функцией, а ограничение активно, то оптимальным решением задачи (3) будет точка, принадлежащая подмножеству крайних точек множества допустимых решений S с базовой точкой X^0, в которой целевая функция принимает наименьшее значение:

$$C(X^0) = \min_{X \in B_2^n} C(X).$$

Рассмотрим отдельное ограничение в задаче оптимизации (1)-(2):

$$A_j(Y) \ge 1, \text{ [84]}$$

где $A_j(Y) = \sum_{\substack{i=1 \\ c_i^j \ne a_i}}^{n} y_i$ для всех $c^j \in K^-$, $j = \{1,2,...,|K^-|\}$.

Введем обозначение

$$\delta_i^j = \begin{cases} 1, \text{ если } c_i^j \neq a_i; \\ 0, \text{ если } c_i^j = a_i. \end{cases}$$

Тогда $$A_j(Y) = \sum_{i=1}^{n} \delta_i^j y_i$$

Функция $A_j(Y)$ монотонно убывает от точки $Y^1 = (1,1,\ldots,1)$.

Крайними точками допустимой области являются точки $Y_k \in O_{n-1}(Y^1)$ (либо, что то же самое, $Y_k \in O_1(Y^0)$), причем такие, что Y_k отличаются от $Y^0 = (0,0,\ldots,0)$ значением *k*-ой компоненты, при которой $\delta_k^j = 1$.

Множеством допустимых решений является объединение подкубов, образуемых крайними точками допустимой области и точкой Y^1:

$$\bigcup_{k:\delta_k^j=1} K(Y_k, Y^1).$$

Теперь перейдем ко всей системе ограничений:

$$A_j(Y) \geq 1, \text{ для всех } j = 1,2,\ldots,|K^-|.$$

Этой системе будут удовлетворять точки, принадлежащие множеству

$$\bigcap_{j=1}^{|K^-|} \bigcup_{k:z_k^j=1} K(Y_k^j, Y^1),$$

которое, в конечном счете, является объединением конечного числа подкубов. Крайние точки допустимой области могут находиться на совершенно разных уровнях точки Y^1. А их количество, в худшем случае, может достигать значения $C_n^{[n/2]}$, то есть мощности среднего уровня.

«Двойственные» алгоритмы оптимизации, то есть алгоритмы, начинающие поиск из недопустимой области (в данном случае, например, из точки Y^0), приводят к поиску ближайших допустимых точек, которые зачастую имеют не очень высокие значения целевой функции, так как лучшие решения могут находиться на любом уровне Y^0.

Далее рассмотрим целевую функцию

$$C(Y) = \sum_{b \in K^+} \prod_{\substack{i=1 \\ b_i \neq a_i}}^{n} (1 - y_i)$$

для некоторого «базового» наблюдения $a \in K^+$. Либо можно записать

$$C(Y) = \sum_{j=1}^{|K^+|} \prod_{i=1}^{n} (1 - \Delta_i^j y_i),$$

$$\text{где } \Delta_i^j = \begin{cases} 1, \text{ если } b_i^j \neq a_i; \\ 0, \text{ если } b_i^j = a_i. \end{cases}$$

Функция $C(X)$ монотонно возрастает от точки $Y^1 = (1,1,\ldots,1)$, принимая в ней значение 1, что соответствует покрытию «базового» наблюдения a. Наибольшее значение, равное $|K^+|$, функция $C(X)$ принимает в точке $Y^0 = (0,0,\ldots,0)$.

Положим, что ближайшее к наблюдению a наблюдение $b \in K^+$ отличается значением s компонент

$$s = \min_{b \in K^+ \setminus \{a\}} \sum_{i=1}^{n} |a_i - b_i|.$$

Во всех точках $Y \in O_k(Y^1)$, $k = 0,1,\ldots,s-1$, значение целевой функции будет одинаковым и равным 1. Наличие такого множества постоянства затрудняет работу алгоритмов оптимизации, начинающих поиск из допустимой точки Y^1 и ведущих его по соседним точкам, так как вычисление целевой функции в системе окрестностей, состоящей из соседних точек, не дает информации о наилучшем

направлении поиска. При решении практических задач больших размерностей это множество постоянства может быть таким, что ему принадлежит большая часть точек допустимой области. Это усложняет или делает невозможной работу таких алгоритмов как генетический алгоритм, локальный поиск с мультистартом.

Другим следствием наличия множеств постоянства целевой функции является то, что оптимальным решением является не только точка, принадлежащая подмножеству крайних точек допустимой области, а это может быть целое множество точек, представляющее собой множество постоянства целевой функции. При этом справедливо следующее утверждение.

Утверждение 3.1. Крайние точки допустимой области задачи (3.1)-(3.2) соответствуют первичным закономерностям.

Доказательство следует из определений крайней точки и первичной закономерности.

При этом следует отметить, что найденная первичная закономерность не обязательно будет являться сильной закономерностью. Этим свойством обладает оптимальное решение задачи.

Утверждение 3.2. Оптимальное решение задачи (3.1)-(3.2) соответствует сильной первичной закономерности.

Доказательство. Из прошлого утверждения следует, что оптимально решение соответствует первичной закономерности. Значение целевой функции задачи – это покрытие закономерности. Если решение оптимально, то не существует закономерности P^a (покрывающей базовое наблюдение a), лучшей по критерию доказательности, а значит, это решение соответствует сильной закономерности. □

Таким образом, применяя приближенные алгоритмы оптимизации, можно утверждать, что найденная закономерность будем являться первичной, но она не обязательно будет являться сильной. Если же использовать точный алгоритм оптимизации, то найденная закономерность будет являться сильной первичной закономерностью.

3.4 Поиск сильных охватывающих закономерностей

Модель оптимизации определяется, прежде всего, тем, каким образом введены переменные, определяющие альтернативы задачи оптимизации. В рассмотренной выше модели оптимизации альтернативы, то есть закономерности, определяются включением или не включением в закономерность условий, которые выполняются в некотором базовом наблюдении. Рассмотрим альтернативный способ задания закономерности [85].

Введем в рассмотрение бинарную переменную

$$z_b = \begin{cases} 1, \text{если наблюдение } b \in K^+ \text{ покрывается } P^a, \\ 0, \text{в противном случае.} \end{cases}$$

Для того чтобы получаемая закономерность была охватывающей, введем в ее состав все литералы, имеющиеся у всех положительных наблюдений, которые эта закономерность покрывает, то есть те литералы, для которых выполняется условие:

$$\prod_{b \in K^+} \left(1 - \left|b_j - a_j\right| \cdot z_b\right) = 1.$$

Теперь сформулируем задачу поиска максимальной закономерности как максимизацию покрытия положительных наблюдений при условии недопустимости покрытия отрицательных наблюдений:

$$\sum_{b \in K^+} z_b \to \max_Z, \tag{3.4}$$

$$\sum_{\substack{j=1 \\ c_j \neq a_j}}^{n} \prod_{b \in K^+} \left(1 - \left|b_j - a_j\right| \cdot z_b\right) \geq 1 \text{ для любого } c \in K^-. \tag{3.5}$$

Решив эту задачу оптимизации, мы определим множество положительных наблюдений, покрываемых искомой закономерностью. Саму закономерность определим, используя характеристические переменные $Y = (y_1, y_2, ..., y_n)$:

$$y_j = \prod_{b \in K^+} \left(1 - \left|b_j - a_j\right| \cdot z_b\right), \; j = 1, ..., n.$$

В такой постановке задачи оптимизации целевая функция является строго монотонной, а множества постоянства целевой функции отсутствуют.

Утверждение 3.3. В задаче (3.4)-(3.5) крайние точки допустимой области, и только они, соответствуют сильным охватывающим закономерностям.

Доказательство. Возьмем некоторую точку $Z_k \in O_k(Z^0)$, где $Z^0 = (z_1, ..., z_{m^+})$. Для любой точки $Z_{k+1} \in O_1(Z_k) \cap O_{k+1}(Z^0)$ выполняется $Cov(T_k) \subset Cov(T_{k+1})$, где T_k и T_{k+1} – термы, соответствующие точкам Z_k и Z_{k+1} соответственно. Если Z_k является крайней точкой допустимого множества, то любая точка $Z_{k+1} \in O_1(Z_k) \cap O_{k+1}(Z^0)$ не является допустимой, а терм T_{k+1} – не является закономерностью, следовательно, терм T_k является сильной закономерностью (согласно определению сильной закономерности).

В свою очередь, если точка Z_k не является крайней точкой, то существует допустимая точка $Z_{k+1} \in O_1(Z_k) \cap O_{k+1}(Z^0)$ и соответствующий терм T_{k+1}, который является закономерностью с большим покрытием $Cov(T_{k+1}) \supset Cov(T_k)$, поэтому терм T_k не является сильной закономерностью. □

Выводы к главе 3

Проведён анализ существующих способов выявления закономерностей в данных и исследованы свойства моделей оптимизации для нахождения закономерностей. Показано, что существующие алгоритмы не гарантируют

получения сильных закономерностей с максимальным покрытием. Разработана новая модель оптимизации для нахождения сильных охватывающих закономерностей в данных. В такой постановке задачи оптимизации целевая функция является строго монотонной, а множества постоянства целевой функции отсутствуют. В построенной модели крайние точки допустимой области соответствуют сильным охватывающим закономерностям.

ГЛАВА 4. ЛОГИЧЕСКИЕ ЗАКОНОМЕРНОСТИ В РАСПОЗНАВАНИИ

В настоящей главе описывается применение закономерностей для принятия решений при распознавании, и исследуются схемы использования семейств закономерностей различных типов.

4.1 Классификация с помощью закономерностей

Основное предположение рассматриваемого подхода состоит в том, что наблюдение, покрытое некоторыми положительными закономерностями, но не покрытое какой-либо отрицательной закономерностью, является положительным, и аналогично, наблюдение, покрытое некоторыми отрицательными закономерностями, но не покрытое какой-либо положительной закономерностью, является отрицательным.

Пусть M^+ будет набором положительных закономерностей, а M^- набором отрицательных закономерностей. Если каждое наблюдение из Q^+ покрыто хотя бы одной закономерностью из M^+, а каждое наблюдение из Q^- покрыто хотя бы одной закономерностью из M^-, то считается, что $M^+ \cup M^-$ является моделью для выборки Q. В некоторых случаях при построении модели может потребоваться, чтобы каждое наблюдение в наборе обучающих данных покрывалось, по крайней мере, k закономерностями в модели, для целого числа $k > 1$. Это дополнительное требование обычно приводит к увеличению числа закономерностей в модели, принося в жертву простоту модели. Однако это может привести к созданию более надежных моделей для классификации новых наблюдений, как это предлагается в [86] и в других экспериментах с комбинациями классификаторов [87, 88].

Построение модели принятия решения для данного набора данных обычно включает в себя создание большого множества закономерностей и выбор из них подмножества, которое удовлетворяет определению представленной выше модели принятия решения, и такой, что каждая закономерность в модели

удовлетворяет определенным требованиям с точки зрения покрытия и однородности.

4.1.1 Представление наблюдений в пространстве закономерностей

Для данного набора данных $\Omega = \{\omega^1, \omega^2, ..., \omega^m\}$, состоящего из m положительных и отрицательных наблюдений, представленных в виде точек в B^n (где B - булево пространство), мы свяжем с каждым наблюдением ω^i выход $\omega_0^i \in \{0,1\}$, который показывает 1, если наблюдение положительное, и 0, если оно отрицательное.

Пусть C будет набором положительных закономерностей P_1, P_2,…, P_q и отрицательных закономерностей N_1, N_2,..., Nr. С каждым наблюдением x можно связать новое представление $\tilde{x} = (\tilde{x}_1, ..., \tilde{x}_q, \tilde{x}_{q+1}, ..., \tilde{x}_{q+r})$, 0/1-вектор с $q + r$ компонентами, являющийся характеристическим вектором, показывающим закономерности в C, удовлетворяющие x. Множество всех возможных таких представлений называется *пространством закономерностей*, связанным с набором C.

Простой подход к классификации логическими закономерностями может быть описан следующим образом. Точка $x \in B^n$ классифицируется как положительная, если она покрыта хотя бы одной из закономерностей P_1, P_2,…, P_q, но не покрыта ни одним из закономерностей N_1, N_2,..., Nr. Аналогично, она классифицируется как отрицательная, если она покрыта, по меньшей мере, одной из закономерностей N_1, N_2,..., Nr, но не покрыта ни одной из закономерностей P_1, P_2,…, P_q. Если точка покрыта как положительными, так и отрицательными закономерностями в C, а также в случае, когда точка не покрыта ни одной закономерностью из C, точка остается неклассифицированной.

Далее будем использовать представление набора данных в пространстве закономерностей для построения двух классификаторов. Эти классификаторы предоставляют альтернативные способы классификации для случая, когда точка

покрывается как некоторыми положительными, так и некоторыми отрицательными закономерностями.

4.1.2 Классификация по индексу «компаса»

Вектор f = (1, 1, ..., 1, 0, 0, ..., 0), первые q компонент которого равны 1, а последние r компонент равны 0, будем называть положительным полюсом пространства закономерностей, соответствующего C, тогда как его «дополнение» g = (0, 0, ..., 0, 1, 1, ..., 1) будет называться отрицательным полюсом. Определение полюсов не требует существования наблюдения x в исходном пространстве данных, представление которого в пространстве закономерностей равно f или g.

Естественно ожидать, что результат точки, которая «ближе» к одному из двух полюсов, будет положительным или отрицательным в соответствии с классом ближайшего полюса. Для любой точки $x \in B^n$ определим притяжения a_f (x) и a_g (x) от x к полюсам как *corr* (f, x) и *corr* (g, x) соответственно, где *corr* обозначает коэффициент корреляции Пирсона.

Индекс компаса C является классификатором, который присваивает положительный (отрицательный) знак наблюдению x, если a_f (x) > 0 (соответственно a_g (x) > 0), и оставляет его неклассифицированным в противном случае. Числовое значение a_g (x) называется оценкой компаса для x.

4.1.3 Балансовый индекс и балансовая оценка

Ассоциируем с набором данных Ω семейство дискриминантных функций D, определенных как линейные комбинации закономерностей в C:

$$D(a) = \sum_{i=1}^{p} \alpha_i P_i(a) - \sum_{j=1}^{n} \beta_j N_j(a)$$

где $\alpha_i, \beta_j \geq 0$ для всех i и j.

Балансовая оценка представляет собой значение дискриминанта, полученного путем присвоения всем параметров α_i и β_j значения $1/q$ и $1/r$ соответственно:

$$D(a) = \frac{1}{p}\sum_{i=1}^{p} P_i(a) - \frac{1}{n}\sum_{j=1}^{n} N_j(a).$$

Классификация π (x) по индексу баланса выполняется следующим образом:

$$\pi = \begin{cases} 1, D(x) > 0, \\ 0, D(x) < 0, \\ ?, D(x) = 0, \end{cases}$$

где «?» означает «неклассифицировано».

Поскольку предполагается, что каждая точка в данном наборе данных покрыта хотя бы одной закономерностью в наборе *C*, можно легко увидеть, что индекс баланса является точным классификатором для этого набора данных.

Интуитивно ясно, что наблюдение имеет более сильное притяжение к положительному (соответственно отрицательному) полюсу, если доля положительных (соответственно отрицательных) закономерностей, которые его покрывают, превышает долю отрицательных (соответственно положительных) закономерностей, которые его покрывают. На самом деле, эта предположение полностью подтверждается теоремой [89], которая утверждает, что *C*(x) = π(x), для каждого $x \in B^n$. Таким образом, эти два классификатора являются взаимозаменяемыми.

4.1.4 Схемы классификации на основе закономерностей

Основным этапом процедуры логического анализа данных является определение набора положительных и отрицательных закономерностей, охватывающих все наблюдения в обучающем наборе (то есть подмножество набора данных, используемого для построения модели принятия решений). Чтобы проверить эту модель, наблюдения в контрольной выборке данных (являющейся дополнением к обучающей выборке и отличной от нее) классифицируются моделью принятия решений. Рассмотрим две схемы классификации, прямая и улучшенная, которая расширяет множество тех наблюдений, которые можно классифицировать [72].

Прямая схема классификации (Definite classification scheme - DECS)

Схема классификации, используемая логическим анализом данных, объявляет наблюдение положительным (соответственно отрицательным), если оно покрыто хотя бы одной положительной (соответственно отрицательной) закономерностью в модели, и оно не покрыто ни одной из отрицательных (соответственно положительных) закономерностей в модели; наблюдение, которое охватывается как положительными, так и отрицательными закономерностями или ни одной из закономерностей в модели, объявляется нераспознанным.

Схема классификации по индексу баланса (Balance index classification scheme - BICS)

BICS - это разновидность вышеуказанной схемы классификации, использующая индекс баланса для определения класса нового наблюдения. Эта классификация оставляет неизменной классификацию тех наблюдений, которые классифицированы DECS как положительные или отрицательные, и расширяет набор наблюдений, классифицированных по DECS.

Сравнение между DECS и BICS

В исследовании [89] сообщается о результатах серии экспериментов, проведенных с четырьмя эталонными наборами данных, взятыми из репозитория машинного обучения Ирвина [90]: рак молочной железы, нарушения функции печени, протоколы голосования конгресса и болезнь сердца. В исследовании делается вывод о том, что все те наблюдения, которые классифицированы DECS как положительные или отрицательные, остаются тем же образом классифицированными BICS, и почти все наблюдения, которые объявлены как неклассифицированные DECS, классифицированы BICS, причем подавляющее большинство из них классифицированы верно. Принимая во внимание, что доля наблюдений, не классифицированных DECS, может быть довольно большой (от 8,21% до 97,3% для эталонных наборов данных, рассмотренных выше), улучшение классификации на основе BICS по сравнению с классификацией на основе DECS может быть значительным. Точность, полученная с использованием BICS, оказывается неизменно выше точности, обеспечиваемой DECS.

В том же исследовании сообщается об аналогичной серии экспериментов, сравнивающих точность BICS с точностью линейного дискриминанта Фишера, и было установлено, что в среднем BICS обеспечивает повышение точности на 4% по сравнению с точностью дискриминантов Фишера. Что наиболее важно, для каждого из используемых наборов эталонных данных было обнаружено, что BICS сопоставим или превосходит самый точный из 33 методов классификации, описанных в [91].

4.2 Выбор логических закономерностей для построения решающего правила распознавания

Предположим, что в результате выполнения процедуры поиска закономерностей по обучающей выборке найден ряд положительных закономерностей P_i, i=1,…,p, и отрицательных закономерностей N_j, j=1,…,n.

Решающая функция может быть задана выражением

$$D(a) = \frac{1}{p}\sum_{i=1}^{p} P_i(a) - \frac{1}{n}\sum_{j=1}^{n} N_j(a)$$

для некоторого наблюдения a,

где $P_i(a)$=1, если закономерность P_i покрывает объект a, и $P_i(a)$=0 в противном случае. То же самое для $N_j(a)$.

В главе 3 описаны алгоритмы поиска закономерностей. В частности, это алгоритмы, которые ведут поиск закономерности, опираясь на некоторый объект обучающей выборки. Поэтому, в результате их работы может быть записано большое число закономерностей, вплоть до числа объектов обучающей выборки, некоторые из которых, впрочем, могут повторяться. При решении многих задач встает вопрос отбора закономерностей из общего их числа для формирования решающего правила, что способно не только уменьшить его размер, но, в некоторых случаях, и улучшить распознавание.

Основное преимущество, которое предоставляют логические алгоритмы распознавания при решении практических задач – это прозрачность процесса распознавания новых объектов по полученной модели. Все выявленные закономерности представлены в явном виде. Но если этих закономерностей много, более, допустим, 10-15, то алгоритм распознавания становится трудно интерпретируемым. В связи с этим исследуем некоторые способы отбора из общего числа найденных закономерностей.

4.2.1 Минимизация числа закономерностей

Введем переменные, определяющие, будет ли закономерность присутствовать в решающей функции.

$$x_i = \begin{cases} 1, P_i \text{ присутствует в решающей функции,} \\ 0, \text{в противном случае.} \end{cases}$$

$$y_j = \begin{cases} 1, N_j \text{ присутствует в решающей функции,} \\ 0, \text{в противном случае.} \end{cases}$$

Один из способов произвести отбор закономерностей – выделить подмножество закономерностей, которые необходимы для покрытия всех объектов обучающей выборки [92]. Каждый объект обучающей выборки должен при этом покрываться хотя бы одной закономерностью. Используя введенные переменные, это условие можно записать в виде

$$\sum_{i=1}^{p} x_i P_i(a) \geq 1 \text{ для любого } a \in K^+,$$

$$\sum_{j=1}^{n} y_j N_j(a) \geq 1 \text{ для любого } a \in K^-.$$

Для повышения робастности алгоритма число 1 в правой части неравенств следует заменить целым положительным числом d. В таком случае каждый объект обучающей выборки должен подчиняться заданному количеству d закономерностей.

Таким образом, имеем следующую задачу минимизации числа используемых в решающем правиле закономерностей:

$$\sum_{i=1}^{p} x_i + \sum_{j=1}^{q} y_j \to \min$$

при ограничениях на переменные:

$$\sum_{i=1}^{p} x_i P_i(a) \geq d \text{ для любого } a \in K^+,$$

$$\sum_{j=1}^{n} y_j N_j(a) \geq d \text{ для любого } a \in K^-.$$

Полученная оптимизационная модель представляет собой задачу условной псевдобулевой оптимизации, в которой целевая функция и функции в ограничениях являются унимодальными монотонными псевдобулевыми функциями. Для решения задачи использовались приближенные алгоритмы условной псевдобулевой оптимизации, основанные на поиске оптимального решения среди граничных точек допустимой области.

Для того чтобы оценить, как влияет уменьшение числа закономерностей в решающем правиле на точность распознавания, была проведена серия экспериментов на задачах распознавания и прогнозирования. Поиск закономерностей производился на основе оптимизационной модели [93], позволяющей находить максимальные закономерности, то есть закономерности с наибольшим покрытием объектов некоторого класса. Каждая выборка данных была разделена на две части – обучающую и тестовую. На основе каждого объекта обучающей выборки производился поиск закономерности. Проводилось сравнения качества распознавания решающих правил, построенных из полного набора закономерностей и из уменьшенного набора, полученного путем решения описанной выше оптимизационной задачи.

При проведении экспериментов использовались следующие задачи распознавания [90]:

breast-cancer – задача диагностики рака молочной железы, объем выборки – 699 объектов, описываемых 9 разнотипными признаками (в результате бинаризации получено 80 бинарных признаков);

wdbc – задача диагностики рака молочной железы, объем выборки – 569 объектов, описываемых 30 разнотипными признаками (120 бинарных признаков);

hepatitis – задача диагностики наследственного гепатита, объем выборки – 155 объектов, описываемых 19 разнотипными признаками (37 бинарных признаков);

spect – данные по сердечной компьютерной томографии объем выборки – 80 объектов, описываемых 22 бинарными признаками.

Результаты экспериментов приведены в таблице 4.1.

Таблица 4.1 - Результаты распознавания

Задача распознавания	Набор закономерностей	Число положительных закономерностей	Число отрицательных закономерностей	Точность распознавания положительных объектов	Точность распознавания отрицательных объектов
breast-cancer	полный набор	419	209	0,97	0,91
	умень-шенный набор	12	14	0,97	0,88
wdbc	полный набор	291	163	0,94	0,98
	умень-шенный набор	9	11	0,92	0,96
hepatitus	полный набор	27	97	0,8	0,85
	умень-шенный набор	7	7	0,8	0,81
spect	полный набор	38	34	1	0,83
	умень-шенный набор	7	8	1	0,83

Как видно из результатов, применение решающего правила, основанного на уменьшенном наборе закономерностей, в некоторых задачах приводит к незначительному снижению качества распознавания, но в то же время сопровождается значительным снижением числа закономерностей, которые необходимо использовать для принятия решения, что положительно сказывается на прозрачности получаемых решений.

4.2.2 Максимизация разделяющей полосы

Еще один способ заключается в том, чтобы произвести отбор таких закономерностей, которые при совместном использовании увеличат разделяющую способность решающего правила [94].

В качестве критерия при формировании решающего правила рассмотрим ширину «разделяющей полосы»

$$\min\{D(a): a \in K^+\} - \max\{D(a): a \in K^-\},$$

где $D(a) = \frac{1}{p}\sum_{i=1}^{p} P_i(a) - \frac{1}{n}\sum_{j=1}^{n} N_j(a)$ для некоторого объекта a.

Учтем наличие выбросов, которые могут присутствовать в реальных задачах. Для этого введем переменную

$$z^a = \begin{cases} 1, a \text{ принимаетс я за выброс}, \\ 0, \text{в противном случае}. \end{cases}$$

Тогда задачу отбора закономерностей можно записать в следующем виде.

$$v^+ + v^- - C\sum_{a \in K} z^a \cdot \left|b^a\right| \to \max,$$

$$\text{где } v^+ = \min\{D'(a): a \in K^+, z^a = 0\},$$

$$v^- = \min\{-D'(a): a \in K^-, z^a = 0\},$$

$$D'(a) = \frac{\sum_{i=1}^{p} x_i P_i(a)}{\sum_{i=1}^{p} x_i} - \frac{\sum_{j=1}^{n} y_j N_j(a)}{\sum_{j=1}^{n} y_j},$$

$$b^a = \begin{cases} v^+ - D'(a), a \in K^+, \\ v^- + D'(a), a \in K^-. \end{cases}$$

4.2.3 Декомпозиция обучающей выборки при выявлении закономерностей

Рассматриваемые в главе 3 способы поиска закономерностей предполагают использование в качестве «опорной» точки объект обучающей выборки (прецедент), частичное повторение свойств которого может быть обнаружено в других объектах этого же класса. Описанный выше способ предписывает использовать большое число таких опорных объектов (возможно, всех объектов обучающей выборки) для получения закономерностей, а затем проводить отбор из найденных.

Рассмотрим другой способ, заключающийся в отборе самих этих опорных объектов. Всё множество объектов обучающей выборки некоторого класса, скажем, K^+ можно разбить на группы объектов так, чтобы объекты были схожи внутри каждой группы:

$$K^+ = K_1^+ \cup K_2^+ \cup \ldots \cup K_k^+$$

Для этого можно использовать алгоритм k-средних, в результате работы которого получаем набор центроидов $c_1, c_2, \ldots, c_k$, так что будет выполняться правило:

$$a \in K_j^+, \text{ если } \|a - c_j\| < \|a - c_i\|$$
$$\text{для всех } i = 1,2,\ldots,k,\ i \neq j,$$

где K_j^+ - множество объектов, входящих в кластер с центроидом c_j.

Эти центроиды можно использовать в качестве опорных объектов для выявления логических закономерностей.

Описанный подход позволяет существенно снизить трудоемкость работы логического алгоритма распознавания, производя отбор объектов, используемых в качестве опорных при поиске закономерностей.

Рассмотрим результаты использования этого подхода применительно к задаче прогнозирования осложнений инфаркта миокарда: фибрилляции предсердий (ФП) и фибрилляции желудочков (ФЖ) [95]. Для нахождения центроидов использовался алгоритм k-средних программного приложения Weka [96], для поиска закономерностей и оценки точности построенного решающего правила использовалось авторское программное обеспечение.

Выборка для задачи ФП состояла из 184 положительных и 184 отрицательных объектов, описываемых 112 разнотипными признаками. Число бинаризованных признаков составило 215. Для каждого класса выделено по 15 центроидов, которые использовались для поиска закономерностей.

Выборка для задачи ФЖ состояла из 80 положительных и 80 отрицательных объектов, описываемых 112 разнотипными признаками. Число бинаризованных признаков составило 200. Для каждого класса выделено по 10 центроидов, которые использовались для поиска закономерностей.

10% объектов выборки было выделено для тестирования полученного решающего правила. Результаты распознавания приведены в таблице 4.2.

Таблица 4.2 - Сравнение результатов распознавания [97]

Задача распознавания	Набор закономерностей	Число положительных закономерностей	Число отрицательных закономерностей	Точность распознавания положительных объектов	Точность распознавания отрицательных объектов
ФП	полный набор	165	165	0,7	0,79
	умень-шенный набор	15	15	0,68	0,77
ФЖ	полный набор	72	72	0,87	0,71
	умень-шенный набор	10	10	0,9	0,88

В результате использования декомпозиции объектов обучающей выборки и соответствующего отбора объектов, используемых в качестве опорных для поиска закономерностей, получаем упрощение решающего правила – число используемых в решающем правиле закономерностей уменьшается в 7-10 раз. При этом для некоторых задач наблюдается даже увеличение точности распознавания тестовых объектов.

4.3 Принятие решения по набору закономерностей

Предположим, что найдено некоторое число положительных и отрицательных закономерностей. Согласно логическому анализу данных, для принятия решения относительно принадлежности некоторого распознаваемого наблюдения одному из классов, используется следующее правило (рисунок 4.1):

1) Если наблюдение покрывается только положительными закономерностями, то оно считается положительным.

2) Если наблюдение покрывается только отрицательными закономерностями, то оно считается отрицательным.

3) Если наблюдение подчиняется условиям t закономерностей одного класса и f другого, то класс наблюдения определяется в результате голосования, например как результат разности $t/T - f/F$, где T и F – число закономерностей этих классов.

4) Если наблюдение не покрывается ни одной закономерностью, то оно считается нераспознанным.

Здесь имеются однозначные области, которые покрываются закономерностями только одного класса; конфликтная область, точки которой покрываются закономерностями разных классов (в этом случае принадлежность к классу определяется голосованием закономерностей); а также область, не покрываемая ни одной закономерностью (наблюдения этой области не могут быть распознаны).

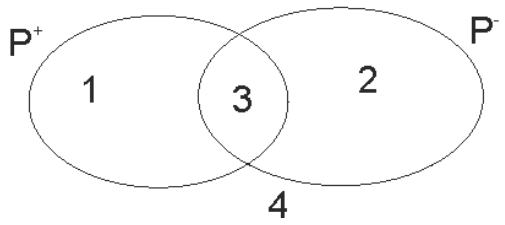


Рисунок 4.1 - Поясняющая схема для правила принятия решения

Использование закономерностей различных видов имеет свои существенные особенности. Влияние вида закономерности на результат распознавания обобщенно представлено в таблице 4.3.

Таблица 4.3 - Влияние видов закономерностей при распознавании

	Первичные закономерности	**Охватывающие закономерности**
Результат распознавания	Меньше нераспознанных наблюдений	Ниже ошибка распознавания
Интерпретируемость	Более короткие правила	Бóльшая уверенность в результате распознавания

Первичные закономерности более простые, состоят из меньшего числа условий. Использование первичных закономерностей уменьшает число нераспознанных наблюдений. Использование сильных охватывающих закономерностей позволяет получать классификаторы с лучшей обобщающей способностью.

Предлагается подход, состоящий в совместном использовании двух видов закономерностей. А именно, построение и использование закономерностей попарно – сильной охватывающей и сильной первичной. Это позволяет совместить преимущества этих двух видов закономерностей.

Если взять некоторую сильную первичную закономерность и соответствующую ей сильную охватывающую (отличающуюся от первичной

наличием дополнительных литералов), то по отношению к ним можно записать следующие выражения:

$$Cov(P^{CO}) = Cov(P^{CП}),$$
$$S(P^{CO}) \subseteq S(P^{CП}),$$
$$Lit(P^{CП}) \subseteq Lit(P^{CO}),$$

где $P^{со}$ – сильная охватывающая закономерность, $P^{сп}$ – сильная первичная закономерность.

Охватывающие закономерности являются более надежными. Первичные – более простые и затрагивают больше потенциальных наблюдений. Это позволит повысить интерпретируемость распознавания и сделать принятие решения более обоснованным (рисунок 4.2).

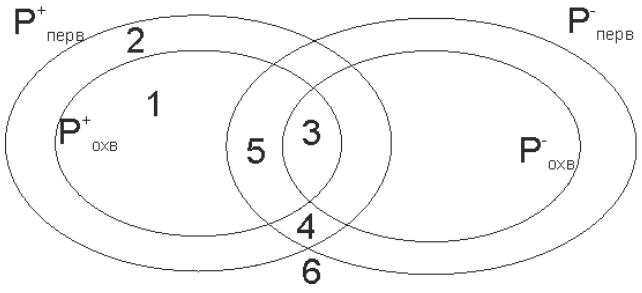


Рисунок 4.2 - Схема, поясняющая принятие решения при использовании пар закономерностей. Расшифровка областей:

- 1- покрытие охватывающей закономерностью одного класса,
- 2- покрытие только первичными закономерностями одного класса,
- 3- покрытие охватывающими закономерностями разных классов,
- 4- покрытие только первичными закономерностями разных классов,
- 5- покрытие охватывающей закономерностью одного класса и первичными закономерностями другого класса,
- 6- нет покрытия закономерностями.

Предлагаемый подход даёт возможность более детально оценить уровень надёжности результата распознавания за счёт уточнённой схемы принятия решения, используя информацию о числе и виде закономерностей, покрывающих наблюдение.

При использовании закономерностей одного вида для принятия решения по классификации наблюдения ситуацию можно отнести к одному из следующих уровней по убыванию определенности в результате распознавания:

- Уровень 1: выполнение закономерностей одного класса.
- Уровень 2: закономерности двух классов → голосование.
- Уровень 3: нет удовлетворяющих закономерностей.

При использовании закономерностей двух видов – первичных (ПЗ) и охватывающих (ОЗ) число таких уровней возрастает:

- Уровень 1: охватывающие закономерности (ОЗ) одного класса.
- Уровень 2: только первичные закономерности (ПЗ) одного класса.
- Уровень 3: ОЗ одного класса и только ПЗ другого.
- Уровень 4: ОЗ двух классов → голосование.
- Уровень 5: только ПЗ (двух классов) → голосование.
- Уровень 6: нет удовлетворяющих закономерностей.

При принятии решения на основе закономерностей одного вида (первичных или охватывающих) имеется четыре возможных варианта (таблица 4.4).

Таблица 4.4 - Классификация по закономерностям одного вида

Выполнение правил		**«-» правила**	
		да	нет
«+» правила	да	Голосование	«+» набл.
	нет	«-» набл.	не распознано

При принятии решения на основе закономерностей двух видов (первичных и охватывающих) число возможных вариантов возрастает (таблица 4.5).

Таблица 4.5 - Классификация по закономерностям двух видов

Выполнение правил		**«-» правила**		
		охв.	перв.	нет
«+» пра-вила	охв.	голос.	+	+
	перв.	--	голос.	+
	нет	--	--	не расп.

Рассмотрим наборы пар закономерностей, состоящих из первичной и соответствующей ей охватывающей закономерностей (с наличием дополнительных литералов $Lit(P^{СП}) \subseteq Lit(P^{СО})$) с одинаковыми покрытиями ($Cov(P^{СО}) = Cov(P^{СП})$). Несмотря на то, что покрытия (на обучающих наблюдениях) этих двух закономерностей в каждой паре одинаковы, подкуб, получаемый первичной закономерностью, может быть больше ($S(P^{СО}) \subseteq S(P^{СП})$). Отсюда следует, что область, получаемая объединением первичных закономерностей одного класса, включает в себя (и может быть шире, чем) область, образуемую объединением соответствующих охватывающих закономерностей.

Рассмотрим классификацию только по первичным закономерностям. Во множестве контрольных наблюдений выделим подмножество наблюдений, которые покрываются закономерностями обоих классов. Для распознавания этих наблюдений применяется голосование. Как показывают эксперименты, большинство ошибок распознавания приходится именно на эти наблюдения. Теперь рассмотрим классификацию этого подмножества наблюдений по

закономерностям двух видов. Рассматриваемое подмножество наблюдений в этом случае можно разбить на четыре группы по комбинациям видов закономерностей, которые их покрывают:

1) охватывающие (и первичные) положительные закономерности и только первичные (без охватывающих) отрицательные закономерности;
2) охватывающие (и первичные) отрицательные закономерности и только первичные (без охватывающих) положительные закономерности;
3) охватывающие (и первичные) закономерности обоих классов;
4) первичные (без охватывающих) закономерности обоих классов.

Ввиду того, что охватывающие закономерности по определению более «избирательные», чем первичные, наблюдения первой группы следует классифицировать как положительные, а наблюдения второй группы как отрицательные, таким образом уменьшая неопределенность, имеющуюся при использовании только первичных закономерностей (сравните таблицы на рисунке 4.3).

Таким образом, использование закономерностей двух видов приводит к повышению точности распознавания по сравнению с использованием одних первичных закономерностей.

Теперь рассмотрим классификацию только по охватывающим закономерностям. Во множестве контрольных наблюдений выделим подмножество наблюдений, которые не покрываются ни одной закономерностью. Это те наблюдения, которые остаются нераспознанными. Применим к этому подмножеству наблюдений первичные закономерности, которые, как показано выше, обладают большим охватом (соответствуют подкубам большего размера). В этом случае часть не покрытых охватывающими закономерностями наблюдений будем покрыта первичными закономерностями. Рассматриваемое подмножество наблюдений в этом случае можно разбить на четыре группы по комбинациям классов закономерностей, которые их покрывают:

1) только положительные (первичные) закономерности;
2) только отрицательные (первичные) закономерности;
3) положительные и отрицательные (первичные) закономерности;
4) нет покрытия.

Выполнение правил		**«-» правила**	
		да	нет
«+» правила	да	Голосование	«+» набл.
	нет	«-» набл.	не распознано

Выполнение правил		**«-» правила**		
		охв.	перв.	нет
«+» правила	охв.	голос.	+	+
	перв.	--	голос.	+
	нет	--	--	не расп.

Рисунок 4.3 – Использование закономерностей двух видов по сравнению только с первичными

Наблюдения первых двух групп следует однозначно распознать как соответственно положительные и отрицательные наблюдения. Для наблюдений

третьей группы следует применить голосование. И лишь наблюдения четвертой группы остаются нераспознанными (сравните таблицы на рисунке 4.4).

Выполнение правил		**«-» правила**	
		да	нет
«+» правила	да	Голосование	«+» набл.
	нет	«-» набл.	не распознано

Выполнение правил		**«-» правила**		
		охв.	перв.	нет
«+» пра-вила	охв.	голос.	+	+
	перв.	--	голос.	+
	нет	--	--	не расп.

Рисунок 4.4 – Использование закономерностей двух видов по сравнению только с охватывающими

Таким образом, использование закономерностей двух видов приводит к снижению числа нераспознанных наблюдений по сравнению с использованием одних охватывающих закономерностей.

4.4 Алгоритмы поиска пары закономерностей

Для нахождения пары закономерностей предлагается следующий алгоритм. Вначале выявляется первичная закономерность в результате поиска решения согласно первой модели оптимизации. Затем первичная закономерность однозначно преобразуется в соответствующую ей охватывающую закономерность.

Алгоритм поиска пары закономерностей (первый вариант)

1. Найти сильную первичную закономерность $P_{перв}$, решив задачу

$$\sum_{b\in K^+}\prod_{\substack{i=1\\ b_i\neq a_i}}^{n}(1-y_i)\to\max_{Y},\ \sum_{\substack{i=1\\ c_i\neq a_i}}^{n}y_i\geq 1 \text{ для всех } c\in K^-.$$

2. Определить множество наблюдений S, которые покрываются $P_{перв}$: S=Cov(P).

3. Найти соответствующую $P_{перв}$ охватывающую закономерность $P_{охв}=\prod_{i\in I}x_i^{\alpha_i}$, где I – множество всех индексов i, для которых i-ые компоненты всех векторов $X\epsilon S$ имеют одинаковой значение.

В этом способе проблема заключается в том, что для нахождения сильной первичной закономерности требуется точное решение задачи оптимизации – нахождение наилучшей крайней точки. Приближенное решение даст лишь первичную закономерность с возможно далеко не лучшим покрытием.

Решение проблемы заключается в использовании более совершенного алгоритма оптимизации, способного найти точное решение, который описывается в следующем разделе.

Альтернативная схема поиска закономерностей попарно состоит в использовании второй модели оптимизации для поиска сильной охватывающей закономерности, а затем нахождении сильной первичной закономерности путем исключения части литералов. Следует отметить, что одной сильной

охватывающей закономерности соответствует множество сильных первичных закономерностей с разным числом условий. Поэтому необходимо решать дополнительную задачу оптимизации, чтобы найти первичную закономерность с наименьшим числом условий.

Алгоритм поиска пары закономерностей (второй вариант)

1. Найти сильную охватывающую закономерность $P_{охв}$, решив задачу $\sum_{b\in K^+} z_b \to \max\limits_{Z}$, $\sum_{\substack{j=1 \\ c_j\neq a_j}}^{n} \prod_{b\in K^+} \left(1-\left|b_j-a_j\right|\cdot z_b\right)\geq 1$ для всех $c\in K^-$.

2. Определить $y_j=\prod_{b\in K^+}\left(1-\left|b_j-a_j\right|\cdot z_b\right)$, $j=1,\ldots,n$.

3. Найти соответствующую $P_{охв}$ первичную закономерность $P_{перв}$, решив задачу: $\sum\limits_{i=1}^{n} y_i' \to \min\limits_{Y}$, $y_i'\leq y_i$ для всех $i=1,\ldots,n$, $\sum\limits_{\substack{i=1 \\ c_i\neq a_i}}^{n} y_i'\geq 1$ для всех $c\in K^-$.

Предлагается единая модель оптимизации для одновременного выявления пары закономерностей. За основу взята исходная модель оптимизации для поиска первичных закономерностей. Имеющиеся неоднозначности исключены путем добавления дополнительных ограничений.

Переменные $Y=(y_1,y_2,\ldots,y_n)$ определяют сильную охватывающую закономерность. Переменные $Y'=(y_1',y_2',\ldots,y_n')$ определяют сильную первичную закономерность. Добавлены ограничения для выявления первичной закономерности:

$$\exists c\in K^-:\sum_{\substack{i=1 \\ c_i\neq a_i}}^{n} y_i'=1 \text{ или иначе } \min_{c\in K^-}\sum_{\substack{i=1 \\ c_i\neq a_i}}^{n} y_i'=1.$$

В свою очередь, для охватывающей закономерности должно выполняться условие: $y_i\geq 1-\sum_{b\in K^+}\left|a_i-b_i\right|\cdot z_b$ для всех $i=1,\ldots,n$,

$$\text{где } z_b = \prod_{\substack{i=1 \\ b_i \neq a_i}}^{n} (1 - y_i) = \prod_{i=1}^{n} (1 - |a_i - b_i| \cdot y_i).$$

Общая модель оптимизации для нахождения пары закономерностей имеет следующий вид:

$$\sum_{b \in K^+} \prod_{\substack{i=1 \\ b_i \neq a_i}}^{n} (1 - y_i) \to \max_{Y},$$

$$y_i \geq 1 - \sum_{b \in K^+} |a_i - b_i| \cdot z_b \text{ для всех } i = 1,\ldots,n,$$

$$y_i \geq y_i' \text{ для всех } i = 1,\ldots,n,$$

$$\min_{c \in K^-} \sum_{\substack{i=1 \\ c_i \neq a_i}}^{n} y_i' = 1,$$

$$\text{где } z_b = \prod_{\substack{i=1 \\ b_i \neq a_i}}^{n} (1 - y_i).$$

Выводы к главе 4

Итак, в данном разделе рассмотрен вопрос отбора закономерностей из общего их числа для формирования решающего правила, что способно не только уменьшить его размер, но и повысить качество распознавания.

Один из способов произвести отбор закономерностей – выделить подмножество закономерностей, которые необходимы для покрытия всех объектов обучающей выборки. Эта задача формулируется в виде задачи оптимизации. Полученная оптимизационная модель представляет собой задачу условной псевдобулевой оптимизации, в которой целевая функция и функции в ограничениях являются унимодальными монотонными псевдобулевыми функциями.

Другой способ заключается в том, чтобы произвести отбор таких закономерностей, которые при совместном использовании увеличат разделяющую способность решающего правила. В качестве критерия при формировании решающего правила рассматривается ширина «разделяющей полосы». Еще один способ заключается в отборе опорных объектов, на основе которых формируются правила.

Отбор логических закономерностей, произведенный в соответствие с предлагаемым подходом, позволяет значительно снизить их число и упростить решающее правило, практически не снижая точность распознавания. Это делает решающее правило прозрачным, а результаты более интерпретируемыми, что необходимо для поддержки принятия решений при распознавании.

Подводя итог, следует заключить, что отбор логических закономерностей, произведенный в соответствие с некоторым критерием, позволяет значительно снизить их число и упростить решающее правило, лишь немного снижая точность распознавания. При решении ряда практических задач распознавания и прогнозирования большое значение имеет интерпретируемость получаемых решений и возможность их обосновать, опираясь на правила и закономерности, которые, в свою очередь, основаны на прецедентах в виде объектов выборки данных. Поэтому использование описанных в этой работе подходов представляется полезным для решения таких задач.

ГЛАВА 5. АЛГОРИТМЫ ОПТИМИЗАЦИИ

5.1 Задача псевдобулевой оптимизации и ее свойства

Вещественные функции, определенные на множестве булевых переменных, по аналогии с булевыми функциями принято называть *псевдобулевыми* [98]. Соответственно, задачи оптимизации псевдобулевых функций называются *задачами псевдобулевой оптимизации*.

Многие проблемы в экономике, банковском деле, промышленности, а также проблемы управления сложными техническими объектами приводят к необходимости решения задач условной оптимизации с булевыми переменными. В работе [99] рассматривается проблема автоматизации планирования товарного ассортимента торговых предприятий, которая сводится к решению многокритериальной задачи псевдобулевой оптимизации с ограничениями. Задача нахождения набора кредитных заявок (задача формирования кредитного портфеля банка) решается как поток задач условной оптимизации псевдобулевых функций [100]. Большое внимание уделяется задаче оптимального проектирования структуры отказоустойчивых систем управления с использованием подхода мультиверсионного программирования, формулируемой в виде задачи условной псевдобулевой оптимизации [101, 102]. Для ее решения построен ряд оптимизационных моделей, в которых максимизируется критерий надежности с учетом стоимостного ограничения. Структура определяется набором булевых переменных, по которым и происходит максимизация критерия. В основном для решения таких задач нашли применение алгоритмы случайного поиска, например, алгоритмы схемы метода изменяющихся вероятностей [103]. В некоторых работах [104, 105] были предприняты попытки построения регулярных алгоритмов.

Псевдобулевые функции играют важную роль в оптимизационных моделях в различных областях, таких как проектирование [106-109], теория надежности

[110], теория вычислительных систем [111], статистика (классификация) [112-114], экономика [115, 116], финансы [117-119], менеджмент [120-122], исследование операций [123], дискретная математика (оптимизация на графах [124, 125]), промышленность (календарное планирование и составление расписания [126-128]).

Помимо задач оптимизации, псевдобулевые функции также появились во многих других моделях, представляющих интерес в настоящее время. Они составляют, к примеру, основной объект исследования в теории кооперативных игр, где они рассматриваются как характеристические функции игр с побочными платежами [129, 130]. Псевдобулевые функции встречаются в комбинаторной теории как функции ранга матроидов [131, 132] или как функции, связанные с определенными параметрами графа, такими как число стабильности, хроматическое число и т.д. [133-135].

Оптимизация псевдобулевых функций используется в распознавании образов, как при отборе информативных признаков [136-138], так и при построении классификаторов [139, 140].

При построении оптимизационных моделей многие задачи естественным образом формализуются в виде задач псевдобулевой оптимизации. Рассмотрим типичную постановку задачи псведобулевой оптимизации. Дано множество n независимых бинарных переменных $X=(x_1,\ldots,x_n)$ и действительная функция $f(X)$, которую необходимо оптимизировать: $f{:}S\to R$, где $S\subset\{0,1\}^n$ – подобласть пространства булевых переменных, определяемая заданной системой ограничений, накладываемых на значения переменных X.

Если $S=\{0,1\}^n$, то есть на выбор переменных $x_1,\ldots,x_n$ не накладываются какие-либо ограничения, то такая задача называется задачей безусловной псевдобулевой оптимизации. Для её решения в [141] разработаны точные алгоритмы, основанные на выявлении в оптимизируемой функции особенностей её поведения в пространстве бинарных переменных. Эти особенности были использованы для построения и обоснования эффективных точных алгоритмов. В

частности, для строго монотонной псевдобулевой функции разработан точный алгоритм оптимизации, требующий n+1 вычислений функции.

Особенностью этих алгоритмов является то, что они не требуют алгебраического задания целевой функции. Требуется лишь возможность вычисления значения функции при заданных аргументах, то есть в точках $\{0,1\}^n$. Такие алгоритмы часто называют поисковыми, и именно о них пойдет речь в данной работе.

Наиболее ”примитивный“ способ найти точное решение задачи псевдобулевой оптимизации - это перебрать все возможные комбинации значений бинарных переменных. Число таких комбинаций равно 2^n. Для многих реальных задач это неприемлемо. Чтобы снизить число вычислений, следует убрать из рассмотрения неперспективные комбинации значений переменных (образующие подобласти исходного пространства бинарных переменных). А для того, чтобы их выявить, нужно знать свойства функции, то есть поведение этой функции на точках (комбинации переменных). На исключении наборов неперспективных альтернатив основаны такие подходы, как метод динамического программирования и метод ветвей и границ.

Многие практические задачи выбора формализуются в виде задач псевдобулевой оптимизации с ограничениями на переменные, при этом в поведении целевой функции и ограничений наблюдаются особенности, которые позволяют строить приемлемые алгоритмы для нахождения точного решения. Вопрос построения таких алгоритмов и рассматривается в этой работе для распространенного класса задач.

Подчеркнем некоторые особенности прикладных задач псевдобулевой оптимизации (141):

1. Неопределенность - скудность (или полное отсутствие) априорных сведений о свойствах целевого функционала, чаще всего заданного алгоритмически (неформализуем).

2. Вычислительная сложность - значительные затраты машинного времени на вычисление значений целевого функционала в точках поиска, что в основном также обусловлено алгоритмическим заданием целевого функционала.

3. Строгая булевость - как правило, значения целевого функционала не могут быть получены в точках $R^n \setminus \{0,1\}^n$ (предполагая погружение $\{0,1\}^n$ в R^n).

4. Часто унимодальность - значительно большая часть унимодальных задач по сравнению со случаем непрерывной оптимизации.

5. Часто потоковость - необходимость решения некоторого множества однотипных задач.

Часто при построении алгоритмов оптимизации из множества возможных псевдобулевых функций выделяют специальные классы функций. Свойства этих классов используют для построения наиболее эффективных (или хотя бы просто работоспособных) алгоритмов.

Например, в работах [142, 143] исследуются специальные классы псевдобулевых функций: квадратичные, монотонные, супермодулярные, субмодулярные и другие. В работах [82, 103] строятся алгоритмы для специальных классов функций, которые заданы алгоритмически.

И действительно, в практических задачах пседобулевой оптимизации очень часто в поведении целевой функции можно наблюдать наличие свойств, которое позволяет отнести эту функции к одному из специальных классов. Но при этом в большинстве практических задач также присутствуют ограничения на выбор значений переменных. Эти ограничения порой сильно сужают допустимую область решений, и решение безусловной задачи (без учета ограничения) будет скорее всего недопустимым и может быть сколь угодно далеким от решения задачи условной (с ограничением). Наиболее очевидный и часто используемый подход для учета ограничений - это составление обобщенной целевой функции, в которой нарушение ограничения наказывается штрафом, делая значение целевой функции при этом хуже, чем для допустимого решения.

Но если у целевой функции имеются специальные свойства, то у обощенной функции со штрафом этих свойств уже не будет. Более подробно это будет рассмотрено далее.

Таким образом, необходимо построение алгоритмов оптимизации, которые изначально направлены на решение задач условной оптимизации. Кроме того, по аналогии с целевыми псевдобулевыми функциями, в ограничениях также можно выделять свойства, образуя специальные классы ограничений.

Широко известны алгоритмы решения задач математического программирования, и в частности, линейного программирования. Это задачи условной оптимизации, в которых целевые функции и ограничения заданы аналитически, в виде уравнений: линейных, квадратичных и т.д.

В этой работе мы обращаем внимание на задачи условной псевдобулевой оптимизации, в которых целевые функции и ограничения предполагаются заданными алгоритмически.

5.1.1 Состояние проблемы

Впервые задачи псевдобулевой оптимизации подробно исследовались в монографии [142]. В этой работе также были разработаны методы решения аналитически заданных задач псевдобулевой оптимизации.

В работе [143] псевдобулевые функции рассматриваются как функции множеств, т.е. функции, отображающие семейство подмножеств конечного исходного множества во множество действительных чисел.

Функции множеств зачастую а заданными алгоритмом, способным выдавать их значения для любого подмножества заданного конечного исходного множества. В некоторых задачах функция множества может быть определена аналитическим выражением, что является весьма благоприятным случаем.

Явное задание функций множеств делает возможным применение большого числа методов для их анализа и оптимизации. Считается, что любая функция

множеств, определенная на конечном исходном множестве, допускает аналитический способ задания. Однако в некоторых случаях определение аналитического выражения может быть значительно более трудоемкой процедурой, чем решение исходной задачи.

Эта работа сосредоточена на тех функциях множеств, которые определены на конечном исходном множестве, представляющем собой множество булевых переменных, и не имеют известного аналитического задания. Особое внимание уделено специальным классам – монотонные и унимодальные псевдобулевые функции.

Известно несколько эффективных схем для решения задач, в которых целевая функция и ограничения заданы аналитическими выражениями. Это алгебраические методы [144-159] (наиболее известный - базовый алгоритм Хаммера [142, 143]), в том числе методы линеаризации [160-164], методы квадратичной оптимизации [165-170]; различные методы с применением релаксации [171].

В практических задачах очень часто целевая функция, а иногда и ограничения заданы алгоритмом или наблюдаются на выходе реальной системы [172]. Это обстоятельство исключает возможность практического применения стандартных процедур математического программирования, опирающихся на известную структуру и вид целевой функции и ограничений. Именно для таких случаев разрабатываются поисковые методы.

Обратим внимание на общую структуру поискового метода [173, 174]. *Алгоритм решения задачи оптимизации* представляет собой последовательную процедуру, имеющую рекуррентный характер. Это означает, что процесс поиска состоит из повторяющихся этапов, каждый из которых определяет переход от одного решения к другому, лучшему, что и образует процедуру последовательного улучшения решения:

$$X_0 \to X_1 \to \ldots \to X_N \to X_{N+1} \to \ldots$$

В этой последовательности каждое последующее решение в определенном смысле лучше, предпочтительнее предыдущего, т. е.

$$X_N \succ X_{N-1},\ N = 1,2,\ldots$$

Алгоритм поиска оптимального решения, таким образом, связывает следующие друг за другом решения. В простейшем случае

$$X_N = F(X_{N-1}),$$

где F – алгоритм поиска, указывающий, какие операции следует сделать в одной точке X_{N-1}, чтобы получить следующую X_N, более предпочтительную.

Рассмотрим специфику поискового алгоритма F. Вполне очевидно, что информации в точке X_{N-1}совершенно недостаточно для перехода в другую, лучшую точку X_N. Необходимо иметь информацию о поведении оптимизируемой функции $f(X)$ и ограничения S в области точки X_{N-1}. Без этого нельзя составить разумный алгоритм F, обеспечивающий условие улучшения.

Поисковый алгоритм для решения оптимизационной задачи состоит обычно из двух этапов: сбор информации о поведении целевой функции и ограничений, часто посредством пробных шагов, и осуществление рабочего шага. Возможны и отклонения от этой схемы, когда оба этапа совмещены и неразделимы, но при этом обязательно сохраняются.

В оптимизации для функций, заданных алгоритмически, широкое распространение получили адаптивные поисковые процедуры [175-178] – эволюционные и генетические алгоритмы, алгоритм имитации отжига. Однако эти алгоритмы требуют настройки большого числа параметров, к тому же отсутствуют аналитические оценки точности таких алгоритмов. Большое распространение в последнее время получили стохастические и локально-

стохастические алгоритмы, разрабатываемые для конкретных классов задач [159, 179-182].

Для конкретных классов задач безусловной псевдобулевой оптимизации разработаны точные регулярные поисковые алгоритмы, в том числе и неулучшаемые [183-186]. В частности, точное определение минимума произвольной строго монотонной унимодальной псевдобулевой функции требует вычисления значений функции в $(n+1)$-ой точке пространства булевых переменных B_2^n.

Однако на практике редко встречаются задачи без ограничений. Существующие в настоящее время методы решения задач с ограничениями либо не имеют эффективных оценок трудоемкости (исключающих полный перебор), либо не имеют априорных оценок точности. Всевозможные методы решения задач с ограничениями на переменные различаются в значительной степени способами учета ограничений. Методы, связанные со сведением условной задачи к безусловной, т. е. методы типа штрафных функций, обычно не учитывают специфику ограничений и приводят к появлению других, не менее существенных трудностей типа овражности и многоэкстремальности штрафной функции [187, 188]. Поэтому необходима разработка методов, в полной мере учитывающих и выявляющих специфику и особенность ограничений.

В задачах псевдобулевой оптимизации допустимое множество решений конечно, поэтому существует универсальный способ отыскания оптимального решения посредством полного перебора всех допустимых решений. Однако такой алгоритм поиска применим только в тех исключительных случаях, когда мощность множества допустимых решений сравнительно невелика. В интересных с практической точки зрения задачах, как правило, мощность множества допустимых решений быстро (например, экспоненциально) растет с увеличением размерности (объема исходных данных) задачи. Это приводит к тому, что в задачах реальной размерности количество допустимых решений становится величиной астрономического порядка, что делает перебор практически

невозможным. Назначение теории задач псевдобулевой оптимизации – создание работоспособных в практическом смысле алгоритмических инструментов решения задач.

Среди комбинаторных оптимизационных задач, частью которых являются и задачи псевдобулевой оптимизации, можно выделить *полиномиально разрешимые* и *NP*-трудные. Для каждой полиномиально разрешимой задачи известен по крайней мере один эффективный алгоритм ее решения, то есть точный алгоритм, трудоемкость которого (время решения) ограничена сверху некоторым полиномом от размера входных данных задачи. Никакую *NP*-трудную задачу нельзя решить никаким известным полиномиальным алгоритмом. Во всяком случае, если бы существовал полиномиальный алгоритм решения какой-нибудь одной *NP*-трудной задачи, то существовали бы полиномиальные алгоритмы для всех *NP*-трудных задач.

Более подробно с вопросами оценки сложности алгоритмов и теорией полиномиальной сводимости комбинаторных задач можно ознакомиться в работах [171, 189-191].

К сожалению, почти все естественные и интересные постановки задач условной псевдобулевой оптимизации оказываются *NP*-трудными, поэтому попытки построения точных и полиномиальных алгоритмов для этих задач вероятней всего обречены на неудачу.

Рассмотрим наиболее известные методы, используемые в настоящее время для решения задач условной псевдобулевой оптимизации. Их можно подразделить на точные (комбинаторные) и приближенные.

Точные методы, очевидно, являются всегда детерминированными, т.е. при одинаковых исходных данных задачи выдают одинаковый результат – оптимальное решение задачи.

Метод ветвей и границ с использованием решения релаксированной задачи для получения нижних границ. Для этого метода необходимо получение значений функций вне точек целочисленной решетки, что ограничивает его применение для задач с неявным заданием функций. Кроме того, нет оценок трудоемкости

алгоритмов, основанных на схеме метода ветвей и границ. Показано лишь, что существуют задачи, число ветвлений в которых (т.е. число решаемых релаксированных задач) незначительно отличается от полного перебора.

Метод динамического программирования получил большое распространение при решении аддитивных (сепарабельных) задач математического программирования. Основу этого метода составляет принцип оптимальности Беллмана. Смысл этого принципа состоит в том, что оптимальная стратегия при любом первоначальном состоянии и любом первоначальном решении предполагает, что последующие решения должны быть определены относительно состояния, полученного в результате предыдущего решения. Алгоритм динамического программирования при решении задачи линейного целочисленного программирования имеет псевдополиномиальные оценки трудоемкости. Но этот метод не нашел применения при решении задач с алгоритмически заданными функциями.

Алгебраические методы: базовый алгоритм Хаммера для решения задач линейного программирования с булевыми переменными и его модификации, алгоритмы квадратичной оптимизации для решения задач квадратичного программирования с булевыми переменными, методы линеаризации, позволяющие решать задачи оптимизации нелинейных псевдобулевых функций. Эти методы предназначены для задач с аналитически заданными целевыми функциями и ограничениями. Алгебраические методы в наиболее полной мере используют структуру заданных функций, составляющих постановку задачи. В основном эти алгоритмы рассматривают задачи безусловной псевдобулевой оптимизации и задачи условной оптимизации с линейными и квадратичными псевдобулевыми целевыми функциями и ограничениями.

Аппроксимационно-комбинаторные методы решения задач псевдобулевой оптимизации. Эти методы основаны на построении аппроксимирующей функции для целевой функции на множестве допустимых значений. Использование такого подхода принципиально затруднено при алгоритмическом задании целевых функций и ограничений.

Точные поисковые алгоритмы псевдобулевой оптимизации, не требующие аналитического задания функций, были построены для решения произвольных задач безусловной псевдобулевой оптимизации. Эти алгоритмы в наиболее полной мере учитывают специфику решаемых классов задач и являются существенно не улучшаемыми. Одной из основных задач данной работы является построение точных поисковых алгоритмов для решения задач условной псевдобулевой оптимизации.

Рассмотрим основные аспекты разработки и исследования приближенных алгоритмов дискретной и псевдобулевой оптимизации [192-195].

Как уже отмечалось выше, интересующие нас задачи являются *NP*-трудными. Это означает, что построение точных полиномиальных алгоритмов невозможно. Одним из основных направлений исследования таких задач является построение приближенных алгоритмов с априорными оценками их точности. Такие алгоритмы не для всякой индивидуальной задачи находят оптимальное решение, но всегда отыскивают допустимое решение задачи, которое удовлетворяет некоторым требованиям по точности, установленными еще до получения приближенного решения.

Существуют два наиболее распространенных подхода к анализу приближенных алгоритмов и определению их точности: анализ поведения в худшем случае и вероятностный анализ поведения алгоритма. При *анализе в худшем случае* оценивается погрешность алгоритма на наихудшей из возможных индивидуальной задаче, а в случае *вероятностного анализа* предполагается, что на множестве индивидуальных задач задано некоторое вероятностное распределение и оценивается математическое ожидание погрешности алгоритма на этом распределении.

Необходимость разработки приближенных алгоритмов связана также с тем, что существуют задачи дискретной оптимизации, для нахождения точного решения которых необходимо перебрать все допустимые решения задачи, т.к. одно допустимое решение не имеет никаких априорных преимуществ перед другим. Таким образом, для решения таких задач нет каких-либо точных

процедур кроме полного перебора множества допустимых решений. Одной из подобных задач является задача вида

$$\begin{cases} f(X) \to \max\limits_{X \in B_2^n}, \\ \sum\limits_{j=1}^{n} x_j = m, \end{cases} \qquad m < n,$$

со множеством допустимых значений

$$S = \{X \in B_2^n : \sum_{j=1}^{n} x_j = m\}.$$

Большую часть вариантов здесь можно исключить как недопустимые решения, но никакие допустимые решения, число которых составляет

$$C_n^m = \frac{n!}{m!(n-m)!},$$

исключать нельзя, и для нахождения точного решения необходимо перебрать их все. К данному виду сводится, например, задача выбора системы информативных факторов [103, 136-138, 196, 197].

5.1.2 Свойства псевдобулевых функций

Определение 5.1. *Псевдобулевой функцией* называют вещественную функцию на множестве булевых переменных: $f : B_2^n \to R^1$, где $B_2 = \{0,1\}$, $B_2^n = B_2 \times B_2 \times \cdots \times B_2$. [183].

Псевдобулевые функции можно рассматривать как функции множеств [143]. Так как существует взаимно однозначное соответствие между

подмножествами множества $N=\{1,2,...,n\}$ и элементами B_2^n, то любая псевдобулевая функция может быть интерпретирована как вещественная *функция множеств,* определенная на *P(N),* показательном множестве множества $N=\{1,2,...,n\}$, т.е. ставит вещественное число в соответствии каждому подмножеству конечного множества. Интерпретация псевдобулевой функции f как функции множеств выделяет тот факт, что значения f получены некоторым неявным путем.

Любая псевдобулевая функция $f(x_1,...,x_n)$ может быть единственным образом представлена как *мультилинейный полином* вида

$$f(x_1,...,x_n)=c_0+\sum_{k=1}^{m}c_k\prod_{i\in A_k}x_i\,, \qquad (5.1)$$

где $c_0,c_1,...,c_m$ – вещественные коэффициенты, $A_1,A_2,...,A_m$ – непустые подмножества $N=\{1,2,...,n\}$ [143].

Кроме того, любая псевдобулевая функция может быть представлена в виде

$$f(x_1,...,x_n)=b_0+\sum_{k=1}^{m}b_k\left(\prod_{i\in A_k}x_i\prod_{j\in B_k}\bar{x}_j\right), \qquad (5.2)$$

где $b_0,b_1,...,b_m$ – вещественные коэффициенты, $\bar{x}_j=1-x_j$, $j=\overline{1,n}$. Если $b_k\geq 0$, $k=\overline{1,m}$, то выражение (5.2) является *позиформой* функции *f*. Любая псевдобулевая функция может быть записана в виде позиформы.

Другие представления псевдобулевых функций представлены в работах [198, 199].

Таким образом, любую псевдобулевую функцию теоретически можно записать в виде алгебраического выражения. Однако при нелинейности ($c_k\neq 0$ при $|A_k|>1$ в выражении (5.1) или $b_k\neq 0$ при $|A_k|+|B_k|>1$ в (5.2)) запись функции в явном виде может потребовать вычисления функции в точках,

количество которых экспоненциально растет с ростом размерности n, и, соответственно, может являться более трудоемкой задачей, чем оптимизация алгоритмически заданной функции. Это обуславливает необходимость изучения свойств псевдобулевых функций, не имеющих явного задания, и разработки поисковых алгоритмов оптимизации.

Рассмотрим основные определения и понятия [183].

Введем понятие соседства точек в пространстве B^n, это схоже с использованием нормы $L1$, но позволяет избежать явной метризации пространства и уделить основное внимание его топологическим свойствам.

Определение 5.2. Точки $X^1, X^2 \in B_2^n$ назовем *k-соседними*, если они отличаются значением k координат, $k = \overline{1,n}$. 1-соседние точки будем называть просто *соседними*.

Определение 5.3. Множество $O_k(X)$, $k = \overline{0,n}$, всех точек B_2^n, k-соседних к точке X, назовем *k-м уровнем* точки X. При этом $O_0(X) = X$.

Лемма 5.1. $\forall X \in B_2^n : card\ O_k(X) = C_n^k$, где C_n^k – число сочетаний из n по k.

Лемма 5.2. $\forall X \in B_2^n : B_2^n = \bigcup_{k=0}^{n} O_k(X)$.

Таким образом, множество всех точек B_2^n можно представить в виде последовательности уровней $O_k(X)$, $k = \overline{0,n}$, какой-либо точки X, каждый из которых состоит из C_n^k точек (рисунок 5.1).

Лемма 5.3. $\forall X^k \in O_k(X)$, $X \in B_2^n$, $k = \overline{0,n}$:

$card\{O_1(X_k) \cap O_{k-1}(X)\} = k$,

$card\{O_1(X_k) \cap O_{k+1}(X)\} = n - k$.

Введем определение подкуба в пространстве булевых переменных.

Определение 5.4. Множество точек $W(X^0, X^l) = \{X^0, X^1, \ldots, X^l\} \subset B_2^n$ назовем *путем* между точками X^0 и X^l, если $\forall\ i = 1, \ldots, l$ точка X^i является соседней к X^{i-1}.

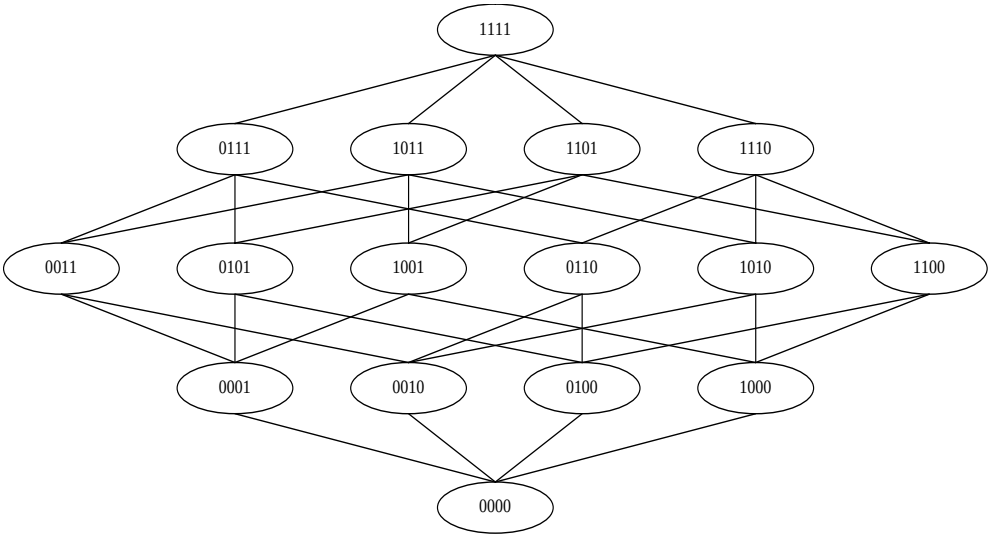


Рисунок 5.1 - Условное графическое представление пространства B_2^n, n=4

Определение 5.5. Путь $W(X^0, X^l) \subset B_2^n$ между k-соседними точками X^0 и X^l назовем *кратчайшим*, если $l = k$.

Определение 5.6. Множество $A \subset B_2^n$ назовем *связным множеством*, если $\forall X^0, X^l \in A$ существует путь $W(X^0, X^l) \subset A$.

Определение 5.7. $\forall X, Y \in B_2^n$ объединение всех кратчайших путей $W(X,Y)$ будем называть *подкубом* B_2^n и обозначать $K(X,Y)$.

Предположим, что значения некоторых координат в двух точках $X, Y \in B_2^n$ совпадают: $x_i = y_i$, $i \in A \subset \{1,...,n\}$; а в остальных различаются: $x_j \neq y_j$, $j \notin A$.

Подкубом $K(X,Y)$ является множество всех точек Z, у которых значения переменных с индексом $i \in A \subset \{1,...,n\}$ фиксированы и равны $z_i = x_i = y_i$, а значения остальных переменных могут принимать любые значения.

Для того чтобы определить, принадлежит ли точка подкубу, воспользуемся следующим свойством.

Свойство 5.1. Точка $Z \in B_2^n$ принадлежит подкубу $K(X,Y)$, если $z_i = x_i$ для $\forall i : x_i = y_i$.

Лемма 5.4. $\forall X^k \in O_k(X), X \in B_2^n, k = \overline{0,n}: \quad O_k(X) \subset \bigcup_{l=0}^{N} O_{2l}(X^k)$, где $N = \min\{k, n-k\}$, а $card(O_{2l}(X^k) \cap O_k(X)) = C_k^l \cdot C_{n-k}^l \ \ \forall l = \overline{0,N}$.

5.1.3 Классы псевдобулевых функций

Ниже рассматриваются понятия модальности и монотонности, на основании которых проводится классификация псевдобулевых функций, а также основные свойства выделяемых классов [183].

Определение 5.8. Точку $X^* \in B_2^n$, для которой $f(X^*) < f(X), \forall X \in O_1(X^*)$, назовем *локальным минимумом* псевдобулевой функции *f*.

Определение 5.9. Псевдобулевую функцию, имеющую только один локальный минимум, будем называть *унимодальной* на B_2^n функцией.

Определение 5.10. Унимодальную функцию f назовем *монотонной* на B_2^n, если $\forall X^k \in O_k(X^*), k = \overline{1,n}: \ \ f(X^{k-1}) \le f(X^k), \quad \forall X^{k-1} \in O_{k-1}(X^*) \cap O_1(X^k)$, и *строго монотонной*, если это условие выполняется со знаком строгого неравенства.

Определение 5.11. Унимодальную функцию f назовем *разнозначной* на B_2^n, если для каждого кратчайшего пути $W(X^0, X^l), \quad X^0 \in B_2^n, \quad X^l = X^*$: $f(X^i) \ne f(X^j)$ при $i \ne j$, $X^i, X^j \in W(X^0, X^l)$, $i, j = \overline{0,l}$.

Определение 5.12. Произвольную унимодальную функцию *f*, для которой условие монотонности нарушается, по крайней мере, в одной точке B_2^n, будем называть *немонотонной* на B_2^n.

Определение 5.13. Путь $W^f(X^0, X^l) \subset B_2^n$ назовем *путем невозрастания* функции f, если $\forall X^i, X^{i-1} \in W^f(X^0, X^l), \ i = 1,2,\ldots,l: f(X^i) \le f(X^{i-1})$, и *путем*

убывания функции, если неравенство выполняется со знаком строгого неравенства.

Замечание 5.1. Из определений 5.11 и 5.13 следует, что произвольная (разнозначная) функция монотонна (строго монотонна) на B_2^n, если $\forall X \in B_2^n$ все пути ее невозрастания (убывания), начинающиеся в точке X^0, являются кратчайшими до X^*.

Лемма 5.5. Для любой унимодальной на B_2^n функции f в каждой точке $X \in B_2^n \setminus \{X^*\}$ среди точек $X_j^1 \in O_1(X)$, $j = \overline{1,n}$, существует, по крайней мере, одна точка $X_{j_i}^1$ такая, что $f(X_{j_i}^1) \le f(X)$.

Следствие 5.1. Если f – унимодальная на B_2^n функция, то $\forall X \in B_2^n \setminus \{X^*\}$ существует по крайней мере один путь ее невозрастания $W^f(X, X^*)$.

Лемма 5.6. Если f – унимодальная на B_2^n функция, то

$$\min_{X_j^k \in O_k(X^0)} f(X_j^k) \le \min_{X_j^{k+1} \in O_{k+1}(X^0)} f(X_j^{k+1}).$$

Определение 5.14. Унимодальную немонотонную на B_2^n функцию f назовем *слабо немонотонной*, если $\forall X^k \in O_k(X^*)$, $k = \overline{1,n}$, точка $X_{\min}^1$ такая, что

$$f(X_{\min}^1) = \min_{X_j^1 \in O_1(X^k)} f(X_j^1),$$

принадлежит $O_{k-1}(X^*)$.

Замечание 5.2. Из определения 5.14 следует, что необходимым и достаточным условием слабой немонотонности немонотонной на B_2^n функции f является: $\forall X^0 \in B_2^n$ путь $W_{\min}^f(X^0, X^*)$ – кратчайший.

Определение 5.15. Унимодальную псевдобулевую функцию f назовем *(строго) структурно монотонной*, если для $\forall X \in O_l(X^*)$, $\forall Y \in O_m(X^*)$, $l < m$, выполняется $f(X) \le f(Y)$ ($f(X) < f(Y)$).

Определение 5.16. Псевдобулевая функция, имеющая более одного локального минимума, называется *полимодальной* на B_2^n функцией.

Замечание 5.3. Из определений 5.10 и 5.15 следует, что структурно-монотонная функция является монотонной функцией.

Пример унимодальной и монотонной псевдобулевой функции дает следующая лемма.

Лемма 5.7. Полином от булевых переменных

$$\varphi(x_1,\ldots,x_n) = \sum_{j=1}^{m} a_j \prod_{i\in\beta_j} x_i ,$$

где $\beta_j \subset \{1,\ldots,n\}$, является унимодальной и монотонной псевдобулевой функцией при $a_j \geq 0$, $j = \overline{1,m}$.

Доказательство. При $a_j \geq 0$, $j = \overline{1,m}$, функция $\varphi(x_1,\ldots,x_n)$ имеет минимум в точке $X^0 = (0,\ldots,0)$, $\varphi(0,\ldots,0) = a_{j_1}$, где j_1 определяется из условия $\beta_{j_1} = \emptyset$.

Предположим, что $\varphi(x_1,\ldots,x_n)$ имеет другой локальный минимум в точке $X^* \in O_k(X^0)$, т.е. имеет *k* единичных компонент $i_1,\ldots,i_k$.

$$\varphi(x_1^*,\ldots,x_n^*) = \sum_{p=1}^{l} a_{j_p} ,$$

где j_p - индексы, для которых $\beta_{j_p} \subset \{i_1,\ldots,i_k\}$.

Возьмем $\forall X' \in O_{k-1}(X^0) \cap O_1(X^*)$. Пусть это будет точка с единичными компонентами $i_2,\ldots,i_k$.

$$\varphi(x_1',\ldots,x_n') = \sum_{p=1}^{l} a_{j_p} ,$$

где j_p - индексы, для которых $\beta_{j_p} \subset \{i_2,...,i_k\}$.

Очевидно, что

$$\varphi(x_1^*,...,x_n^*) \geq \varphi(x_1',...,x_n'),$$

что противоречит предположению о существовании локального минимума в точке X^*. Таким образом, $\varphi(x_1,...,x_n)$ является унимодальной функцией.

Так как для $\forall X^k \in O_k(X^0)$, $k = \overline{1,n}$:

$$\varphi(x_1^k,...,x_n^k) \geq \varphi(x_1^{k-1},...,x_n^{k-1}),\ \forall X^{k-1} \in O_{k-1}(X^0) \cap O_1(X^k),$$

функция $\varphi(x_1,...,x_n)$ является монотонной. ■

5.1.4 Постановка задачи условной псевдобулевой оптимизации

Формальная постановка задачи условной псевдобулевой оптимизации выглядит следующим образом:

$$f(X) \to extr, \tag{5.3}$$

где

$$f: S \to R^1, \tag{5.4}$$

а $S \subset B_2^n$ – некоторая подобласть пространства булевых переменных, определяемая заданной системой ограничений.

В данной работе рассматривается задача вида

$$\begin{cases} C(X) \to \max\limits_{X \in B_2^n}, \\ A_j(X) \le H_j,\ j = \overline{1,m}, \end{cases} \qquad (5.5)$$

где $C(X)$ и $A_j(X)$ – псевдобулевые функции, обладающие некоторыми конструктивными свойствами (модальность, монотонность и т.п.).

Задача (5.5) является *NP*-полной задачей дискретной оптимизации, т.е. ее нельзя решить никаким известным полиномиальным алгоритмом.

Для простоты изложения результатов и описания работы алгоритмов в дальнейшем часто будем рассматривать задачу с одним ограничением

$$A(X) \le H\,. \qquad (5.6)$$

Рассмотрим свойства множества допустимых решений рассматриваемой задачи. Введем несколько понятий для точек, особым образом расположенных в бинарном пространстве.

Определение 5.17. Точка $Y \in \mathbf{A}$ является *граничной точкой* множества $\mathbf{A}$, если существует $X \in O_1(Y)$, для которой $X \notin \mathbf{A}$.

Если за множество $\mathbf{A}$ принять множество точек, в которых выполняются ограничения, то есть допустимое множество, то точки, описываемые приведенным выше определением, можно называть граничными точками допустимой области.

Выберем во множестве $\mathbf{A}$ некоторую базовую точку X^0. Тогда среди всего множества граничных точек можно выделить подмножество точек, удовлетворяющих приведенному ниже определению, которые назовем крайними точками.

Определение 5.18. Точку $Y \in O_i(X^0) \cap \mathbf{A}$ будем называть *крайней точкой* множества $\mathbf{A}$ с базовой точкой $X^0 \in \mathbf{A}$, если $\forall X \in O_1(Y) \bigcap O_{i+1}(X^0)$ выполняется $X \notin \mathbf{A}$.

Пример расположения крайних точек множества приведен на рисунке 5.2.

Определение 5.19. Ограничение, определяющее подобласть пространства булевых переменных, будем называть *активным*, если оптимальное решение задачи условной оптимизации не совпадает с оптимальным решением соответствующей задачи оптимизации без учета ограничения.

Другими словами, ограничение активно, если оптимальное решение безусловной задачи является недопустимым для задачи с ограничением.

В дальнейшем будем рассматривать ситуацию, когда ограничение (или система ограничений), определяющее допустимую область пространства булевых переменных, является активным.

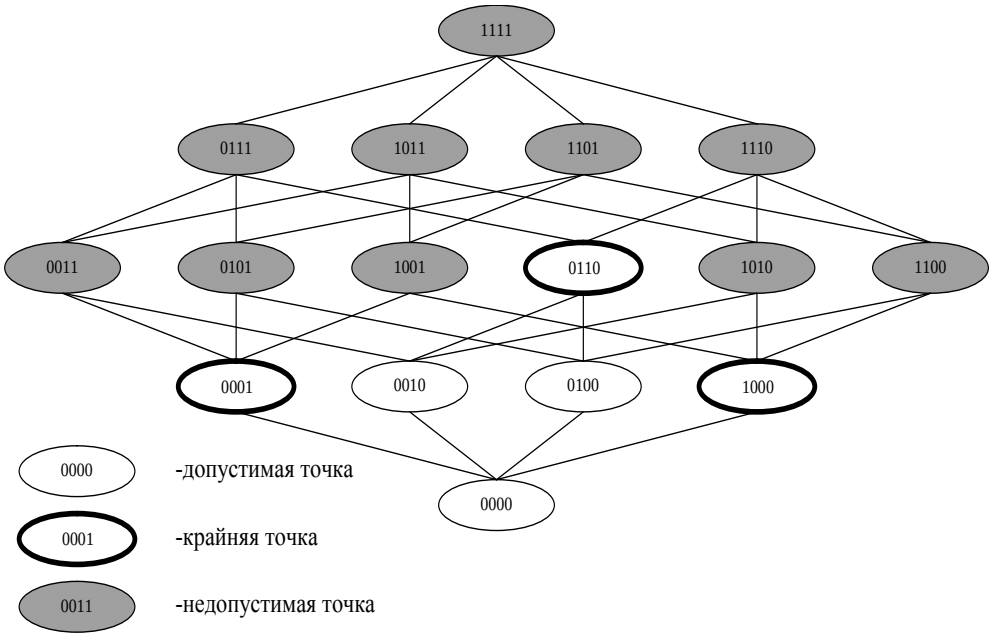


Рисунок 5.2 - Условное представление пространства B_2^n, n=4, $X^0 = (0,...,0)$, пример расположения крайних точек

Свойство 5.2. Если ограничение является активным, то оптимальное решение принадлежит подмножеству граничных точек допустимой области.

Доказательство. Следует из определений 5.17 и 5.18. ■

Замечание 5.4. Обратное не верно.

Одно из свойств допустимого множества решений выглядит следующим образом:

Теорема 5.1. Если целевая функция является монотонной унимодальной функцией, а ограничение активно, то оптимальным решением задачи (5) будет точка, принадлежащая подмножеству крайних точек множества S допустимых решений с базовой точкой X0, в которой целевая функция принимает наименьшее значение:

$$C(X^0) = \min_{X \in B_2^n} C(X).$$

Доказательство. Если бы решение задачи X^* не являлось крайней точкой, то нашлась хотя бы одна точка $X' \in O_1(X^*)$ из области допустимых решений, принадлежащая более высокому уровню точки X^0, для которой $C(X') < C(X^*)$, что противоречит условию монотонности функции. ■

Рассмотрим задачу (5.5) с одним ограничением (5.6).

Теорема 5.2. Если функция ограничения (5.6) является унимодальной псевдобулевой функцией, то множество допустимых решений S задачи (5) представляет собой связное множество.

Доказательство. Из определения 5.6 и следствия 5.1 следует, что если функция ограничения $A(X)$ унимодальная, то для любой допустимой точки $X \in B_2^n : A(X) \le H$ $(X \subset S)$ существует, по крайней мере, один путь убывания $W^f(X, X^*)$, поэтому между любыми двумя точками допустимой области $\forall X, Y \subset S$ существует путь $W(X, Y) \subset S$. ■

Лемма 8. Количество крайних точек связного множества допустимых решений задачи (5.5)

$$s \le s_{\max} = C_n^{[n/2]},$$

где $[n/2]$ – целая часть числа $n/2$. В случае $s = s_{\max}$ все крайние точки расположены на $n/2$-ом уровне точки X^0 при четном n и на $(n-1)/2$-ом (или $(n+1)/2$-ом) уровне при нечетном n.

Доказательство. Пусть две крайние точки X_1 и X_2 принадлежат разным уровням точки X^0: $X_1 \in O_{k_1}(X^0)$, $X_2 \in O_{k_2}(X^0)$, $0 < k_1 < k_2 < n$. (Если крайняя точка принадлежит уровню 0 или n, то больше крайних точек не существует.) Из определений 5.8 и 5.10 следует, что точки подмножества $\Omega = K(X^0, X_1) \cup K(X_1, X^1) \cup K(X^0, X_2) \cup K(X_2, X^1)$ не могут быть крайними. Введем величину $g_k = card(O_k(X^0) \cap \Omega)$. На k-ом уровне может быть не более $C_n^k - g_k$ крайних точек,

$$g_k > \begin{cases} \max\{C_{n-k_1}^{k-k_1}, C_{k_2}^k\}, k_1 \le k \le k_2, \\ \max\{C_{k_1}^k, C_{k_2}^k\}, 0 < k < k_1, \\ \max\{C_{n-k_1}^{k-k_1}, C_{n-k_2}^{k-k_2}\}, k_2 < k < n. \end{cases}$$

Так как $g_k \ge 2$, то выполняется

$$C_n^k - g_k + 2 < C_n^k,\ 0 < k < n,$$

где $C_n^k - g_k + 2$ - общее число крайних точек, максимально возможное в этом случае, C_n^k - число крайних точек, если бы они все принадлежали k-му уровню. Из этого и из $C_n^{[n/2]} = \max_i C_n^i$ следует, что максимально возможное число крайних точек $s_{\max} = C_n^{[n/2]}$. ■

Если функция ограничения является монотонной псевдобулевой функцией с базовой точкой $X^0 \in S$, то допустимое множество является объединением подкубов, образованных точкой X^0 и крайними точками допустимой области:

$$S = \bigcup_{j=1}^{s} K(X^0, Y^j),$$

где $Y^j, j = 1,...,s$ - крайние точки допустимой области.

В то же время недопустимое множество является объединением подкубов, образованных точкой $X^1 \equiv O_n(X^0)$ и крайними точками допустимой области, за исключением самих крайних точек:

$$R = B^n \setminus S = \bigcup_{j=1}^{s} \left(K(X^j, Y^1) \setminus Y^j \right).$$

5.1.5 Преобразование в задачу безусловной оптимизации

Одним из способов учета ограничений в условных задачах является составление обобщенной штрафной функции. Характерной особенностью задач псевдобулевой оптимизации является тот факт, что даже при унимодальной целевой функции обобщенная функция становится многоэкстремальной при введении даже простых (линейных) ограничений.

Для решения задачи условной оптимизации

$$\begin{cases} C(X) \to \max\limits_{X \in B_2^n} \\ A_j(X) \le H_j, \; j = \overline{1, m} \end{cases} \qquad (5.7)$$

составляется обобщенная функция со штрафом и оптимизируется алгоритмом локального поиска с мультистартом. Лучший из найденных локальных (условных) максимумов принимается за ответ.

Известна следующая обобщенная функция со штрафом [202]

$$F(X) = C(X) - r \cdot \sum_{j=1}^{m} (A_j(X) - H_j). \qquad (5.8)$$

Исследуем ее свойства. Для простоты рассмотрим случай с одним ограничением в виде неравенства

$$F(X) = C(X) - r \cdot (A(X) - H) \text{ или } F(X) = C(X) - r \cdot A(X).$$

Пусть $X' \in O_k(X^0)$ - крайняя точка, принадлежащая *k*-му уровню X^0, т.е. для $\forall X \in O_1(X') \bigcap O_{k+1}(X^0)$: $A(X) > H$.

В случае монотонности целевой функции и ограничения

1) для $\forall Y \in O_1(X') \bigcap O_{k-1}(X^0)$: $C(Y) \le C(X')$ и $A(Y) \le A(X')$;
2) для $\forall Z \in O_1(X') \bigcap O_{k+1}(X^0)$: $C(Z) \ge C(X')$ и $A(Z) > A(X')$.

Обозначим

$$C(X') - C(Y) = L_1 \ge 0,\ C(Z) - C(X') = L_2 \ge 0,$$
$$A(X') - A(Y) = M_1 \ge 0,\ A(Z) - A(X') = M_2 > 0.$$

Для адекватного решения задачи обобщенная функция должна принимать локальный максимум в точке X':

$$\forall Y : F(X') > F(Y); \qquad \forall Z : F(X') > F(Z).$$

Иначе

$$C(X') - C(Y) - r(A(X') - A(Y)) > 0,$$
$$C(X') - C(Z) - r(A(X') - A(Z)) > 0,$$

откуда

$$\frac{C(Z)-C(X')}{A(Z)-Z(X')} < r < \frac{C(X')-C(Y)}{A(X')-A(Y)}$$

или

$$\frac{L_2}{M_2} < r < \frac{L_1}{M_1}.$$

Последнее выражение показывает, что очень трудно подобрать подходящий параметр для такой обобщенной функции (5.8). Поэтому лучше несколько изменить обобщенную функцию со штрафом:

$$F(X) = C(X) - r \cdot \max\{0, A(X) - H\}. \qquad (5.9)$$

Проделав аналогичные преобразования, получаем следующее условие для параметра r, при котором обобщенная функция принимает локальный оптимум в точке X':

$$r > \frac{C(Z)-C(X')}{A(Z)-H} \text{ или } r > \frac{L_2}{M_2}. \qquad (5.10)$$

Отсюда следует, что для адекватности обобщенной функции (5.8) необходимо взять достаточно большое r.

Для случая задачи (5.7) с несколькими ограничениями обобщенная функция (5.9) записывается в виде

$$F(X) = C(X) - r \cdot \sum_{j=1}^{m} \max\{0, A_j(X) - H_j\}. \qquad (5.11)$$

В этом случае имеет место следующее утверждение.

Утверждение 5.1. Крайние точки множества допустимых решений S исходной задачи (5.7) с базовой точкой X^0 являются точками локальных

максимумов для обобщенной штрафной функции (5.11) при параметре *r*, удовлетворяющем условию

$$r > \frac{C(Z) - C(X')}{\sum_{j=1}^{m} \max\{0, A_j(Z) - H\}}$$

для любой крайней точки $X' \in O_k(X^0)$ и любой точки $Z \in O_{k+1}(X^0) \cap O_1(X')$.

Доказательство. Пусть $X' \in O_k(X^0)$ - крайняя точка, т.е. $A_j(X') \le H_j$, $j = \overline{1,m}$, и для $\forall Z \in O_1(X') \bigcap O_{k+1}(X^0)$: $A_j(Z) > H_j$ по крайней мере для одного ограничения. Тогда

$$F(Z) \le C(Z) - r \cdot (A_j(Z) - H_j) < C(X') = F(X').$$

Для $\forall Y \in O_1(X') \bigcap O_{k-1}(X^0)$ выполняется

$$A_j(Y) \le H_j, \ j = \overline{1,m},$$
$$\text{и } F(Y) = C(Y) \le C(X') = F(X').$$

Отсюда следует, что X' - точка локального максимума. ■

Таким образом, задача (5.7) идентична задаче

$$F(X) \to \max_{X \in B^n},$$

в которой количество локальных максимумов (в случае связности множества допустимых решений) $s \le s_{\max} = C_n^{[n/2]}$. В общем случае, когда допустимое множество не является связным, число локальных максимумов теоретически может достигать 2^{n-1}.

Существенным недостатком такого подхода является потеря свойства монотонности целевой функции $C(X)$. При введении даже простых (например, линейных) ограничений обобщенная функция будет полимодальной немонотонной функцией с экспоненциальным числом точек локальных максимумов.

5.1.6 Идентификация свойств псевдобулевых функций

При решении практических задач с алгоритмически заданными функциями в отдельных частных случаях свойства функций можно идентифицировать исходя из некоторых априорных сведений. Но во многих встречающихся на практике задачах принадлежность функции к какому-либо классу вовсе не очевидна, поэтому необходимо проводить идентификацию свойств функций посредством осуществления пробных шагов. Только после этого возможно применение соответствующего поискового алгоритма.

Один из подходов к идентификации свойств псевдобулевых функций задачи (1.5) основан на применении регулярного алгоритма оптимизации строго монотонных псевдобулевых функций [200].

Этот алгоритм заключается в следующем [183]. Произвольным образом выбирается точка $X \in B_2^n$, вычисляется значение функции в выбранной точке и во всех соседних к ней точках, т.е. отличающихся от X значением одной координаты. На основании полученных данных однозначно определяется точка оптимума. Таким образом, алгоритм требует вычисления значений функции (трудоемкость) в n+1 точке и является не улучшаемым по трудоемкости. Этот алгоритм является точным при оптимизации строго монотонных псевдобулевых функций, но результат его применения к немонотонной функции может быть сколь угодно плохим. В практических задачах для удостоверения, является ли найденная точка действительно точкой оптимума, требуется "просмотреть" еще n+1 точку. Различные модификации этого алгоритма описаны в [201].

Подход заключается в многократном применении регулярного алгоритма из различных стартовых точек. Если во всех случаях алгоритм выдаст одну и ту же точку, являющуюся точкой оптимума, то более вероятно, что проверяемая функция является унимодальной и строго монотонной. Очевидно, при увеличении числа прогона алгоритма из различных стартовых точек вероятность этого растет. Если алгоритм выдаст две разные точки, то функция не является унимодальной строго монотонной.

К сожалению, данный подход позволяет лишь проверить гипотезу о том, что функция является строго монотонной. Точный алгоритм оптимизации монотонных псевдобулевых функций имеет экспоненциальную оценку трудоемкости сверху [183], поэтому проверка гипотезы о монотонности функции (имеющей множества постоянства) подобным образом является чрезмерно трудоемкой процедурой.

Рассмотрим возможности идентификации свойств с помощью аппроксимации алгоритмически заданной псевдобулевой функции полиномом.

В [202] показано, что любая псевдобулевая функция $f(x_1,\ldots,x_n)$ может быть единственным образом представлена как *мультилинейный полином* вида

$$f(x_1,\ldots,x_n) = c_0 + \sum_{k=1}^{m} c_k \prod_{i\in A_k} x_i\,, \qquad (5.12)$$

где $c_0, c_1,\ldots,c_m$ - вещественные коэффициенты, $A_1, A_2,\ldots,A_m$ - непустые подмножества множества $N = \{1,2,\ldots,n\}$.

Таким образом, любую псевдобулевую функцию теоретически можно записать в виде алгебраического выражения. Так, например, произвольную линейную псевдобулевую функцию *n* переменных можно записать в явном виде

$$f(x_1,\ldots,x_n) = c_0 + \sum_{i=1}^{n} c_i x_i$$

(т.е. определить коэффициенты c_i, $i=\overline{0,n}$), вычислив ее значения в n+1 точке пространства B_2^n. Однако при нелинейности ($c_k \neq 0$ при $|A_k|>1$ в (5.12)) запись функции в явном виде может потребовать вычисления функции в точках, количество которых экспоненциально растет с ростом размерности n.

Предлагается следующий подход для идентификации свойств псевдобулевых функций. Неявно заданная функция $f(X)$ аппроксимируется функцией $g(X)$, заданной алгебраическим выражением. Затем по виду функции $g(X)$ определяется ее принадлежность к одному из классов.

Линейная аппроксимация даст не больший результат, чем подход, описанный в предыдущем разделе, т.к. линейный полином с ненулевыми коэффициентами является всегда унимодальной строго монотонной функцией. Возможности линейной аппроксимации для оптимизации ограничены точным нахождением оптимума строго монотонной псевдобулевой функции. Легко показать, что точка оптимума линейной функции $g(X)$, аппроксимирующей строго монотонную функцию $f(X)$, совпадает с точкой оптимума функции $f(X)$.

Рассмотрим квадратичную псевдобулевую функцию

$$g(X) = c_0 + \sum_{i=1}^{n} c_i x_i + \sum_{i=1}^{n-1} \sum_{j=i+1}^{n} c_{ij} x_i x_j . \qquad (5.13)$$

Функция (5.13) в общем случае является полимодальной псевдобулевой функцией, т.е. может иметь несколько локальных максимумов (минимумов). Анализ значений коэффициентов квадратичной функции позволяет установить ее принадлежность как к выделенным выше классам функций, так и к более узким классам субмодулярных и супермодулярных функций, которым некоторые авторы уделяют особое внимание (например, в [203, 204]).

Один из простейших в вычислительном отношении способов квадратичной аппроксимации алгоритмически заданной псевдобулевой функции $f(X)$ функцией (5.13) заключается в следующем.

1. Вычисляется значение $f(X)$ в точке $X^0 = (0,...,0)$, $c_0 = f(X^0)$.
2. Вычисляется значение $f(X)$ в точках $X_i \in O_1(X^0)$ (полученных из X^0 путем присвоения координате x_i значения 1), $c_i = f(X_i) - c_0$, $i = \overline{1,n}$.
3. Вычисляется значение $f(X)$ в точках $X_{ij} \in O_2(X^0)$ (полученных из X^0 путем присвоения координатам x_i и x_j значения 1), $c_{ij} = f(X_{ij}) - f(X_i) - f(X_j) - c_0$, $i = \overline{1,n-1}$, $j = \overline{i+1,n}$

Таким образом, для данной операции необходимо вычислить значение $f(X)$ в $1 + n + C_n^2 = n(n+1)/2 + 1$ точке B_2^n.

Недостаток такого подхода очевиден. Аппроксимация проводится только на малой части B_2^n, по близко расположенным друг к другу точкам, вследствие чего значение функции $g(X)$ в точках $X_k \in O_k(X^0)$, где k близко к n, может значительно отличаться от $f(X_k)$ по двум причинам: либо в силу нарастания погрешности, либо из-за того, что $f(X)$ неадекватно описывается квадратичной функцией.

Поэтому предлагается второй способ. Случайным образом выбираются $n(n+1)/2+1$ различных точек B_2^n, в которых вычисляется значение $f(X)$. Затем составляется система линейных уравнений, из которой находятся значения коэффициентов квадратичной функции.

Произвольной точке $X \in B_2^n$ соответствует уравнение

$$c_0 + \sum_{\substack{i=1 \\ x_i=1}}^{n} c_i + \sum_{\substack{i=1 \\ x_i=1}}^{n-1} \sum_{\substack{j=i+1 \\ x_j=1}}^{n} c_{ij} = f(X).$$

Рассмотрим вопрос определения свойств квадратичной функции

В [202] были введены понятия *i-ой производной*

$$\Delta_i(X) = \frac{\partial f}{\partial x_i}(X) = f(x_1,\ldots,x_{i-1},1,x_{i+1},\ldots,x_n) - f(x_1,\ldots,x_{i-1},0,x_{i+1},\ldots,x_n)$$

псевдобулевой функции $f(X)$ и *i-го остатка*

$$\Theta_i(X) = f(X) - x_i\Delta_i(X).$$

В соответствие с этим псевдобулевая функция является монотонной, если для любого $i = \overline{1,n}$ выполняется: $\Delta_i(X) \geq 0$ для всех $X \in B_2^n$ либо $\Delta_i(X) \leq 0$ для всех $X \in B_2^n$.

Обозначим

$$\Delta_i^{\max} = \max_X \Delta_i(X),$$

$$\Delta_i^{\min} = \min_X \Delta_i(X).$$

Псевдобулевая функция является монотонной, если

$$\Delta_i^{\min} \cdot \Delta_i^{\max} \geq 0 \qquad (5.14)$$

для любого $i = \overline{1,n}$, и строго монотонной, если это условие выполняется со знаком строгого неравенства.

Для квадратичной функции (5.13) имеем

$$\Delta_i(X) = c_i + \sum_{\substack{j=1 \\ j\neq i}}^{n} c'_{ij}x_j,$$

$$\text{где } c'_{ij} = \begin{cases} c_{ij}, i < j, \\ c_{ji}, i > j. \end{cases}$$

Значения $\Delta_i^{\max}$ и $\Delta_i^{\min}$ определяем следующим образом

$$\Delta_i^{\max} = \Delta_i(X^{i\max}),$$

$$\text{где } x_j^{i\max} = \begin{cases} 1, c'_{ij} > 0 \\ 0, c'_{ij} \le 0 \end{cases}, \ j = \overline{1,n}, \ j \ne i, \ x_i^{i\max} \text{ - произвольное;}$$

$$\Delta_i^{\min} = \Delta_i(X^{i\min}),$$

$$\text{где } x_j^{i\min} = \begin{cases} 1, c'_{ij} < 0 \\ 0, c'_{ij} \ge 0 \end{cases}, \ j = \overline{1,n}, \ j \ne i, \ x_i^{i\min} \text{ - произвольное.}$$

Иначе

$$\Delta_i^{\max} = c_i + \sum_{\substack{j=1, j\ne i \\ c'_{ij} > 0}}^{n} c'_{ij}, \qquad \Delta_i^{\min} = c_i + \sum_{\substack{j=1, j\ne i \\ c'_{ij} < 0}}^{n} c'_{ij}. \tag{5.15}$$

Для проверки свойства монотонности (строгой монотонности) квадратичной функции (5.13) необходимо вычислить $\Delta_i^{\max}$ и $\Delta_i^{\min}$ для $i = \overline{1,n}$ по выражениям (5.15) и подставить полученные значения в условие (5.14). Если это условие выполняется для всех $i = \overline{1,n}$, то функция является монотонной (строго монотонной) с точкой минимума $X^{\min}$, определяемой следующим образом

$$x_j^{\min} = \begin{cases} 1, \Delta_j^{\min} < 0, \\ 0, \Delta_j^{\max} > 0. \end{cases}$$

Предположим, что условие (5.14) нарушилось для коэффициентов $i_1, i_2, ..., i_p$. В таком случае 2^p точек, получаемых варьированием коэффициентов $i_1, i_2, ..., i_p$, являются подозрительными на минимум. Для определения количества локальных минимумов необходимо сравнить эти точки между собой.

Как заключение по рассмотренным вопросам, отметим, что в общем случае задача условной псевдобулевой оптимизации с алгоритмически заданными функциями не решается стандартными алгоритмами математического программирования. Приведены основные топологические свойства пространства булевых переменных, базирующиеся на понятиях соседства, уровней и подкубов. Исследование свойств множеств допустимых решений задачи условной псевдобулевой оптимизации показывает, что в случае унимодальности ограничения множество допустимых решений является связным множеством. Кроме того, доказано, что при монотонной целевой функции оптимальное решение задачи принадлежит подмножеству крайних точек множества допустимых решений.

Подход к идентификации свойств путем проверки их выполнения при многократном применении регулярных поисковых алгоритмов практически приемлем лишь для проверки свойства строгой монотонности унимодальных псевдобулевых функций. Для более детальной идентификации свойств построена процедура, основанная на аппроксимации алгоритмически заданных функций алгебраическими полиномами. Свойства полиномов определяются исходя из значений их коэффициентов. Полинома второй степени достаточно для описания рассматриваемых свойств. После того, как все функции идентифицированы, задачу можно решать соответствующим алгоритмом. С помощью квадратичной аппроксимации возможно также определение других свойств псевдобулевых функций, таких как субмодулярность, супермодулярность, полярность, унимодулярность и т.д., не рассмотренных в этой работе.

5.2 Точные алгоритмы условной псевдобулевой оптимизации

Для построения регулярных алгоритмов проведем классификацию задач условной псевдобулевой оптимизации, основанную на конструктивных свойствах целевых функций и ограничений [205].

Рассматриваются классы задач вида

$$\begin{cases} C(X) \to \max\limits_{X \in B_2^n}, \\ A(X) \le H, \end{cases}$$

в которых целевая функция $C(X)$ и функция $A(X)$, определяющая систему ограничений, обладают определенными конструктивными свойствами.

При разработке регулярных алгоритмов рассматриваются следующие классы псевдобулевых функций:

- унимодальная;
- монотонная;
- структурно монотонная.

Всевозможным сочетанием классов целевых функций и классов функций ограничений получаются соответствующие классы задач условной псевдобулевой оптимизации.

Из определений монотонности и структурной монотонности функций следуют свойства:

Свойство 5.3. Если функция f монотонно возрастает от $X^0 \in B_2^n$, то для любой точки $Y \in B_2^n$ выполняется:

a) $f(X) \le f(Y)$ для всех $X \in K(X^0, Y)$;

b) $f(X) \ge f(Y)$ для всех $X \in K(Y, X^1)$.

Свойство 5.4. Если функция f структурно монотонно возрастает от $X^0 \in B_2^n$, то для любой точки $Y \in O_k(X^0)$ выполняется:

a) $f(X) \le f(Y)$ для всех $X \in O_l(X^0)$, $l < k$;

b) $f(X) \ge f(Y)$ для всех $X \in O_l(X^0)$, $l > k$.

Рассмотрим свойства целевых функций и функций ограничений, принадлежащих перечисленным выше классам. Эти свойства необходимо учитывать при построении регулярных алгоритмов.

В первую очередь рассмотрим *свойства функций ограничений*.

Унимодальная функция ограничения имеет единственный минимум в точке $X^0 \in B_2^n$. Обозначим $X^1 \equiv X \in O_n(X^0)$.

Согласно теореме 5.2 множество допустимых точек в этом случае является связным множеством. Основное свойство, которое вытекает из введенных выше определений, заключается в следующем.

Свойство 5.5. Из леммы 5.6 следует, что если функция $A(X)$ унимодальна и на уровне $O_k(X^0)$ нет допустимых точек, то их нет на уровне $O_l(X^0)$, где $l > k$.

Следствие 5.2. Если функция $A(X)$ унимодальна и на уровне $O_k(X^0)$ все точки являются недопустимыми или крайними, то на уровне $O_l(X^0)$, где $l > k$, нет допустимых точек (рисунок 5.3).

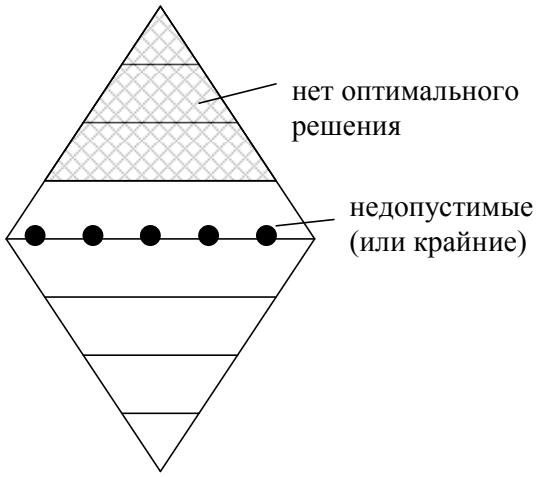


Рисунок 5.3 - Иллюстрация следствия 5.2

Монотонная функция ограничения возрастает от точки $X^0 \in B_2^n$. Основываясь на понятиях монотонности и подкуба, можно вывести следующие свойства.

Свойство 5.6.

a) если функция $A(X)$ монотонна и точка $Y \in B_2^n$ является допустимой (удовлетворяет ограничению $A(Y) \le H$), то любая точка $X \in K(X^0, Y)$ является также допустимой;

b) если функция $A(X)$ монотонна и точка $Y \in B_2^n$ является недопустимой (не удовлетворяет ограничению $A(Y) \le H$), то любая точка $X \in K(Y, X^1)$ является также недопустимой (рисунок 5.4).

Следствие 5.3. Если функция $A(X)$ монотонна и точка $Y \in B_2^n$ является крайней точкой, то любая точка $X \in K(X^0, Y)$ не является крайней точкой, а также любая точка $X \in K(Y, X^1)$ не является крайней точкой (т.е. при поиске остальных крайних точек подкубы $K(X^0, Y)$ и $K(Y, X^1)$ можно исключить из рассмотрения).

Следствие 5.4. Если функция $A(X)$ монотонна и точки $X_1 \in O_{k_1}(X^0)$ и $X_2 \in O_{k_2}(X^0)$, $k_1 < k_2$, являются граничными точками, то на любом уровне $O_k(X^0)$, $k_1 < k < k_2$, существует хотя бы одна граничная точка.

Структурно монотонная функция ограничения возрастает от точки $X^0 \in B_2^n$.

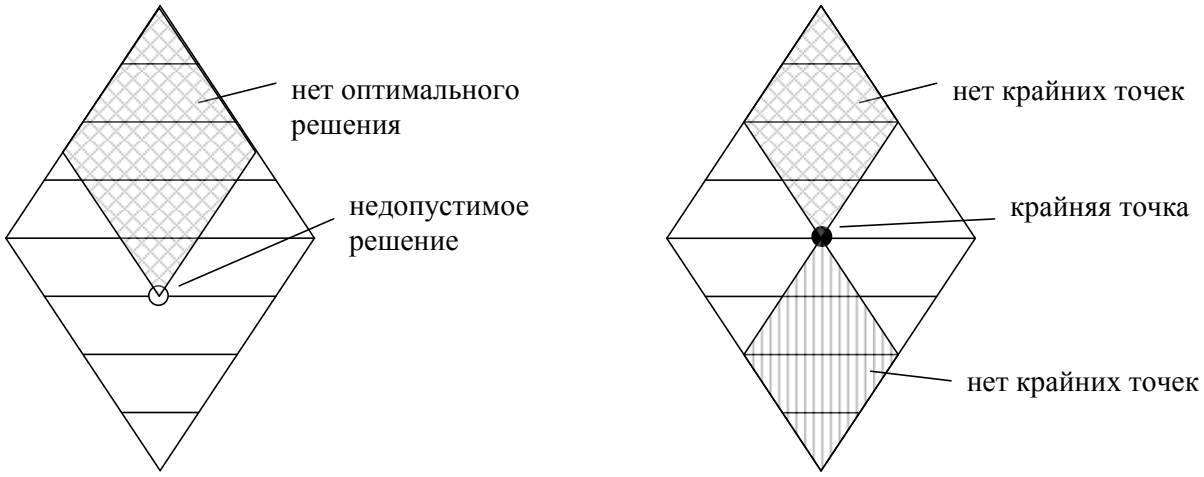


Рисунок 5.4 - Иллюстрации свойства 5.6

Свойство 5.7.

a) если функция $A(X)$ структурно монотонна и точка $Y \in O_k(X^0)$ является допустимой (удовлетворяет ограничению $A(Y) \le H$), то любая точка $X \in O_l(X^0)$, $l < k$, является также допустимой;

b) если функция $A(X)$ структурно монотонна и точка $Y \in O_k(X^0)$ является недопустимой (не удовлетворяет ограничению $A(Y) \le H$), то любая точка $X \in O_l(X^0)$, $l > k$, является также недопустимой (рисунок 5.5).

Следствие 5.5. Если функция $A(X)$ структурно монотонна, то возможен один из двух вариантов:

a) существует такое K, $0 \le K \le n$, что:

$$\forall X \in O_k(X^0),\ k \le K:\ A(X) \le H;$$

$$\forall X \in O_k(X^0),\ k > K:\ A(X) > H;$$

в этом случае все граничные (а также крайние) точки принадлежат $O_K(X^0)$;

b) существует такое K, $0 \le K \le n$, что:

$$\forall X \in O_k(X^0),\ k < K:\ A(X) \le H;$$

$$\forall X \in O_k(X^0),\ k > K:\ A(X) > H;$$

$$\exists X \in O_K(X^0):\ A(X) \le H;\ \exists X \in O_K(X^0):\ A(X) > H;$$

в этом случае граничные (а также крайние) точки принадлежат $O_K(X^0) \cup O_{K-1}(X^0)$.

Далее рассмотрим *свойства целевых функций*.

Унимодальная целевая функция имеет единственный максимум в точке $X^1 \equiv X \in O_n(X^0)$.

Согласно свойству 5.2 оптимальное решение принадлежит подмножеству граничных точек.

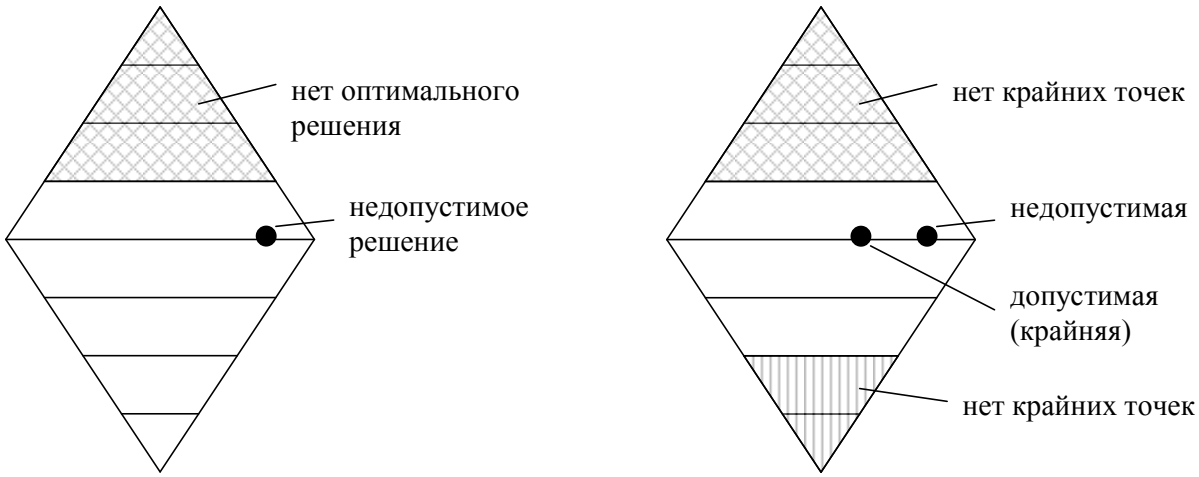


Рисунок 5.5 - Иллюстрация свойства 7

Свойство 5.8. Из леммы 5.6 следует, что если функция $C(X)$ унимодальна и решение, дающее наибольшее значение функции $C(X)$ на уровне $O_k(X^0)$, является допустимым, то на уровне $O_l(X^0)$, где $l < k$, нет оптимального решения (рисунок 5.6).

Монотонная целевая функция возрастает от точки $X^0 \in B_2^n$.

Согласно теореме 5.1 оптимальное решение принадлежит подмножеству крайних точек.

Свойство 5.9. Из свойства 5.3 следует, что если функция $C(X)$ монотонна и решение $Y \in B_2^n$ является допустимым (удовлетворяет ограничению $A(Y) \le H$), то в подкубе $K(X^0, Y)$ нет оптимального решения (рисунок 5.7).

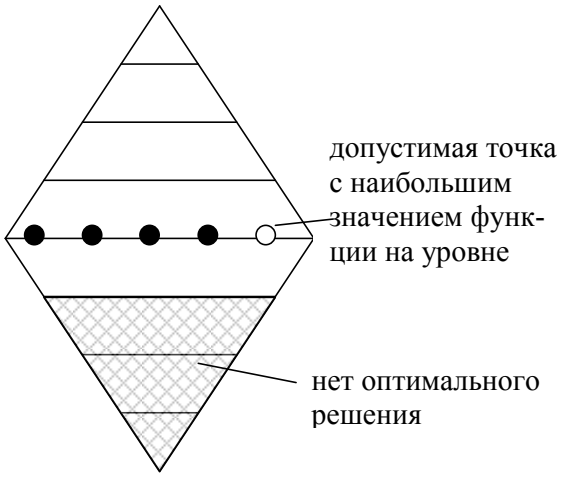


Рисунок 5.6 - Иллюстрация свойства 5.8

Структурно монотонная целевая функция возрастает от точки $X^0 \in B_2^n$.

Свойство 5.10. Из свойства 5.4 следует, что если функция $C(X)$ монотонна и решение $Y \in O_k(X^0)$ является допустимым (удовлетворяет ограничению $A(Y) \le H$), то на уровне $O_l(X^0)$, где $l < k$, нет оптимального решения (рисунок 5.8).

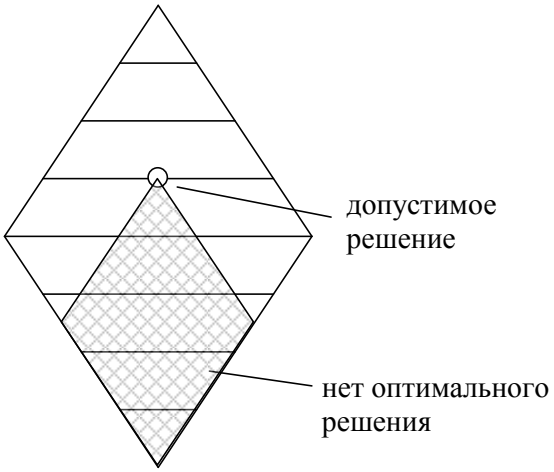


Рисунок 5.7 - Иллюстрация свойства 5.9

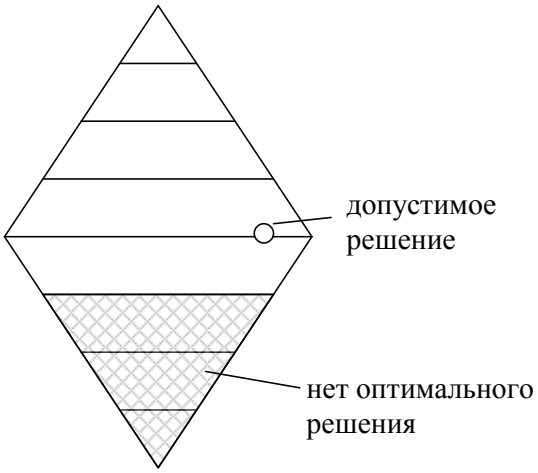


Рисунок 5.8 - Иллюстрация свойства 5.10

Описанные свойства являются основой для построения регулярных алгоритмов поиска точных решений задач условной псевдобулевой оптимизации.

Свойство монотонности довольно часто встречается у псевдобулевых функций в практических задачах оптимизации. Как видно из описанных выше результатов исследования свойств псевдобулевых функций, наличие свойства монотонности дает возможность значительно сократить перебор при поиске оптимального решения и построить эффективные алгоритмы оптимизации.

Рассмотрим построение алгоритма оптимизации для случая, когда целевая функция $C(X)$ и функция ограничения $A(X)$ монотонно возрастают от $X^0 \in B_2^n$. Оптимальное решение принадлежит подмножеству крайних точек множества допустимых решений. Если решение $Y \in B_2^n$ является крайней точкой, то в подкубах $K(X^0, Y)$ и $K(Y, X^1)$ крайних точек нет, и их можно исключить из рассмотрения при поиске остальных крайних точек (рисунок 5.9).

С применением этих свойств был построен описанный ниже регулярный алгоритм нахождения точного решения поставленной задачи.

На первом этапе поиска необходимо найти какую-либо одну граничную точку. Алгоритм начинает работу из точки X^0 (стартовая точка). Если эта точка не удовлетворяет ограничению, то задача не имеет решения. Берется любая допустимая точка $X_1 \in O_1(X^0)$ и принимается за стартовую. Если среди точек,

соседних к X^0, нет допустимых, то решением задачи является X^0. Далее берется любая допустимая точка $X_2 \in O_1(X_1) \cap O_2(X^0)$. Если среди точек подмножества $O_1(X_k) \cap O_{k+1}(X^0)$, где $X_k \in O_k(X^0)$, нет допустимых, то X_k является первой найденной граничной точкой. $Y_1 = X_k$.

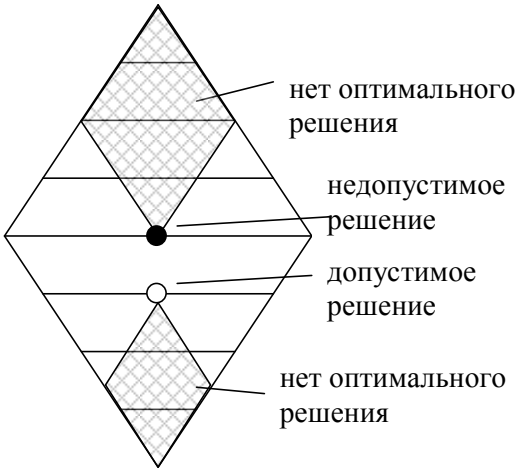


Рисунок 5.9. Исключаемые подкубы для случая монотонных функций

Все точки подкубов $K(X^0, Y_1)$ и $K(Y_1, X^1)$ исключаются из дальнейшего рассмотрения.

На следующем этапе поиска берется точка $X \in O_k(X^0) \setminus Y_1$. Если эта точка не удовлетворяет ограничению, то подкуб $K(X, X^1)$ исключается, и рассматривается точка из подмножества $(O_1(X) \cap O_{k-1}(X^0)) \setminus K(X^0, Y_1)$. В противном случае, если точка X допустима, и допустима хотя бы одна точка из подмножества $(O_1(X) \cap O_{k+1}(X^0)) \setminus K(Y_1, X^1)$, то алгоритм переходит к этой новой точке. Если же X допустима, но ни одна из точек $O_1(X) \cap O_{k+1}(X^0)$ не выполняет ограничение, то X является граничной точкой, $Y_2 = X$.

Далее процедура повторяется.

Если после нахождения Y_s подмножество $O_{k_s}(X^0) \setminus (K(X^0, Y_1) \cup ... \cup K(X^0, Y_s) \cup K(Y_1, X^1) \cup ... \cup K(Y_s, X^1))$ пусто, то алгоритм прекращает поиск граничных точек.

До данного момента алгоритм вычислял значения лишь функции $A(X)$ в рассматриваемых точках и сравнивал его с величиной H.

После того как найдены граничные точки, в них вычисляются значения функции $C(X)$, и точка с наибольшим значением целевой функции принимается за оптимальное решение задачи условной оптимизации.

Алгоритм 5.1.

1. Полагаем $k=1$, $\Omega=\varnothing$, $s=0$, $X_0 = X^0$. Если $A(X^0) > H$, то задача не имеет допустимого решения.
2. Выбираем $X_{k+1} \in (O_1(X_k) \cap O_{k+1}(X^0)) \setminus \Omega$, для которого $A(X_{k+1}) \le H$. Если для какого-либо $X \in O_1(X_k) \cap O_{k+1}(X^0)$ не выполняется $A(X) \le H$, то $\Omega = \Omega \cup K(X, X^1)$.
3. Если такой X_{k+1} существует, то $k = k+1$ и возвращаемся к п. 2.
4. Принимаем $s = s+1$, $Y_s = X_k$, $\Omega = \Omega \cup K(X^0, Y_s) \cup K(Y_s, X^1)$.
5. Выбираем $X \in (O_k(X^0) \cap O_{2m}(Y_s)) \setminus \Omega$ таким образом, чтобы m было наибольшим, $1 \le m \le \min\{k, n-k\}$. Если таких точек нет, то переходим к п. 11.
6. Если $A(X) > H$, то $\Omega = \Omega \cup K(X, X^1)$ и выбираем $X' \in (O_1(X) \cap O_{k-1}(X^0)) \setminus \Omega$, иначе переходим к п. 8.
7. Принимаем $X = X'$, $k = k-1$. Если $A(X) > H$, то возвращаемся к п. 6.
8. Если $A(X) \le H$, то выбираем $X' \in (O_1(X) \cap O_{k+1}(X^0)) \setminus \Omega$, для которого $A(X) \le H$. Если таких точек нет, то переходим к п. 10.
9. Принимаем $X = X'$, $k = k+1$ и возвращаемся к п.8.

10. Принимаем $s = s+1, \quad Y_s = X, \quad \Omega = \Omega \cup K(X^0, Y_s) \cup K(Y_s, X^1)$ и переходим к п. 5.

11. Определяем оптимальное решение X^* из условия $C(X^*) = \max\limits_{i=\overline{1,s}} C(Y_i)$.

Теорема 5.3. Нахождение решения задачи условной псевдобулевой оптимизации с целевой функцией и ограничением, монотонно возрастающими от точки $X^0 \in B_2^n$, в случае, когда все крайние точки Y_s принадлежат $O_k(X^0)$, $s = \overline{1, C_n^k}$, требует просмотра T^k точек B_2^n,

$$T^k = k + C_n^k + C_n^{k+1}.$$

Следствие 5.6. Оценка сверху трудоемкости алгоритма 5.1

$$T_3 \le \max_{0 \le k < n} T^k = T^{\left[\frac{n}{2}\right]} = \left[\frac{n}{2}\right] + C_n^{\left[\frac{n}{2}\right]} + C_n^{\left[\frac{n}{2}\right]+1}.$$

Теорема 5.4. Для определения точного решения задачи условной псевдобулевой оптимизации с целевой функцией и ограничением, монотонно возрастающими от точки $X^0 \in B_2^n$, в худшем случае (при максимально возможном количестве крайних точек) требуется просмотреть не менее $C_n^{[n/2]} + C_n^{[n/2]+1}$ точек B_2^n.

Оценки трудоемкости сверху разработанных точных алгоритмов псевдобулевой оптимизации для задач с целевыми функциями и функциями ограничений, обладающих определенными конструктивными свойствами, обобщены в таблице 5.1 (при условии структурной монотонности функций ограничений) [206]

Таблица 5.1 – Оценки трудоемкости сверху регулярных алгоритмов оптимизации, предназначенных для различных классов задач

C(X) \ *A(X)*	монотонная	немонотонная унимодальная	полимодальная
монотонная	$\left[\frac{n}{2}\right] + C_n^{\left[\frac{n}{2}\right]} + C_n^{\left[\frac{n}{2}\right]+1}$	$C_n^{\left[\frac{n}{2}\right]-1} + C_n^{\left[\frac{n}{2}\right]} + C_n^{\left[\frac{n}{2}\right]+1}$	2^n
немонотонная унимодальная	$C_n^{\left[\frac{n}{2}\right]-1} + C_n^{\left[\frac{n}{2}\right]} + C_n^{\left[\frac{n}{2}\right]+1}$	$\left[\frac{n}{2}\right] + C_n^{\left[\frac{n}{2}\right]-1} + C_n^{\left[\frac{n}{2}\right]} + C_n^{\left[\frac{n}{2}\right]+1}$	2^n
полимодальная	2^n	2^n	2^n

Рассмотрим случай, когда система ограничений состоит из нескольких неравенств:

$$\begin{cases} C(X) \to \max\limits_{X \in B_2^n}, \\ A_j(X) \le H_j,\ j = \overline{1,m}. \end{cases}$$

Введем характеристическую функцию системы неравенств

$$\Phi(X) = \begin{cases} 0, A_j(X) \le H_j, \forall j = \overline{1,m}, \\ 1, \text{в противном случае}. \end{cases}$$

При $\Phi(X) = 0$ решение X удовлетворяет системе ограничений, а при $\Phi(X) = 1$ решение X не выполняет хотя бы одно неравенство.

Отметим следующее свойство. Если все функции $A_j(X)$ системы ограничений являются монотонно возрастающими от точки X^0, то и функция $\Phi(X)$ является монотонной.

Таким образом, вместо системы неравенств можно использовать неравенство $\Phi(X) \le 0$ и решать задачу соответствующим алгоритмом.

Итак, рассмотрены различные классы задач условной псевдобулевой оптимизации и соответствующие алгоритмы. Оценивая эффективность регулярных алгоритмов в целом, можно заключить, что они гарантируют точное решение задачи условной оптимизации практически любых псевдобулевых функций с различными типами ограничений при средней трудоемкости меньшей, чем трудоемкость полного перебора. Алгоритм 1 в наибольшей мере учитывает свойства класса задач условной оптимизации, в которых целевая функция и ограничение представляют собой монотонные унимодальные псевдобулевые функции, и, таким образом, реализует информационную сложность данного класса задач.

Для применения регулярных алгоритмов необходимо предварительно соотнести задачу с одним из выделенных классов. Если нет никаких априорных сведений о свойствах псевдобулевых функций, заданных алгоритмически, необходимо провести процедуру идентификации свойств функций.

5.3 Приближенные алгоритмы условной псевдобулевой оптимизации

Теория алгоритмов указывает на то, что всякий алгоритм должен обладать свойством однозначности, т. е. при одинаковых исходных данных результат его работы должен быть одинаковым. Случайный поиск расширяет понятие алгоритма, допуская тем самым неоднозначность результата при одинаковых исходных данных.

Естественно подразделить все возможные алгоритмы поиска на два класса:

- детерминированные, регулярные алгоритмы поиска, обладающие указанным свойством однозначности;
- недетерминированные (случайные, стохастические, вероятностные и т.д.) алгоритмы поиска, не обладающие свойством однозначности, результат работы которых имеет статистический характер.

В работе [174] показано, что при решении дискретных задач оптимизации очень естественно использовать метод случайного поиска. Это связано с тем, что всякого рода градиентные представления, свойственные детерминированным методам, в дискретном случае теряют смысл.

Случайный поиск как метод решения условной задачи отличается рядом преимуществ по сравнению с детерминированными методами [73, 207]. Здесь у случайного поиска имеется ряд возможностей, связанных со случайным характером поиска, которых в принципе не может иметь ни один детерминированный метод решения задачи.

Одними из наиболее эффективных алгоритмов случайного поиска для задач безусловной псевдобулевой оптимизации (а также для задач условной оптимизации специального вида) являются алгоритмы метода изменяющихся вероятностей [103]. Однако для данной задачи эти алгоритмы не учитывают в полной мере специфику решаемой задачи.

Так как в условной задаче оптимальное решение находится среди граничных точек множества допустимых решений, то алгоритм оптимизации должен быть направлен на поиск именно граничных точек.

Рассмотрим задачу оптимизации монотонно возрастающей от точки X^0 псевдобулевой функции при произвольном ограничении. Оптимальное решение в этом случае, как показано выше, принадлежит подмножеству крайних точек множества допустимых решений.

Простейший алгоритм случайного поиска крайних точек состоит в следующем. Поиск начинается из X^0. На каждом шаге алгоритм выбирает соседнюю точку на следующем уровне, двигаясь, таким образом, по пути возрастания целевой функции до границы допустимой области. При

необходимости процедура повторяется несколько раз, и из найденных крайних точек выбирается лучшая.

Рассмотренные выше два алгоритма на каждом шаге случайно выбирают допустимую точку на следующем уровне (если такая существует). Таким образом, при поиске граничных точек эти алгоритмы не учитывают значения целевой функции $C(X)$ и функции $A(X)$, кроме как для определения, удовлетворяет ли точка ограничению, и используют значения целевой функции лишь при сравнении найденных граничных точек (последний пункт алгоритма).

Альтернативой случайному поиску граничных точек является регулярный алгоритм, использующий гриди эвристику [208].

Гриди алгоритмы являются интуитивными эвристиками, в которых на каждом шаге принимается решение, являющееся наиболее выгодным в данный момент, без учета того, что происходит на последующих шагах поиска [209].

Описанные выше и исследуемые далее схемы приближенных алгоритмов поиска граничных точек можно агрегировать в общую схему [210]. Для любой эвристики поиска граничных точек будем рассматривать пару алгоритмов – прямой и двойственный. Прямой алгоритм начинает поиск из допустимой области и движется по пути возрастания целевой функции, пока не найдет крайнюю точку допустимой области. Напротив, двойственный алгоритм ведет поиск в недопустимой области по пути убывания целевой функции, пока не найдет некоторого допустимого решения.

Общая схема прямого алгоритма поиска

1. Полагаем $X_1 = X^0$, $i = 1$.
2. В соответствии с правилом выбираем $X_{i+1} \in O_i(X^0) \cap O_1(X_i) \cap S$. Если таких точек нет, то переходим к п. 3; иначе $i = i + 1$ и повторяем шаг.
3. Принимаем $X_{opt} = X_{i+1}$.

Общая схема двойственного алгоритма поиска

1. Полагаем $X_1 \in O_n(X^0)$, $i = 1$.

2. В соответствии с правилом выбираем $X_{i+1} \in O_{n-i}(X^0) \cap O_1(X_i)$. Если $X_{i+1} \in S$, то переходим к п. 3; иначе $i = i + 1$ и повторяем шаг.

3. Принимаем $X_{opt} = X_{i+1}$.

Как видно из приведенных схем, прямой алгоритм находит крайнюю точку допустимой области, в то время как двойственный алгоритм находит граничную точку, которая может и не являться крайней. Поэтому для задач со связной областью допустимых решений прямой алгоритм в среднем находит лучшее решение, чем двойственный. Если же мы применим прямой алгоритм для задачи с несвязной допустимой областью, то полученное в результате решение может быть далеким от оптимального, так как на пути возрастания целевой функции допустимые и недопустимые решения будут чередоваться. Для таких случаев более пригоден двойственный алгоритм, для которого это чередование не играет никакой роли. Для улучшения решения, полученного двойственным алгоритмом, рекомендуется применять соответствующий прямой алгоритм. Такое улучшение на практике оказывается весьма значительным [211, 212].

Рассматриваемые здесь алгоритмы поиска граничных точек отличаются друг от друга лишь правилом выбора следующей точки на шаге 2 общих схем.

Правило 1. Случайный поиск граничных точек (СПГ)

Точка X_{i+1} выбирается случайным образом. Каждая точка на следующем шаге может быть выбрана с равной вероятностью. При решении практических задач эти вероятности могут быть не равными, а вычисляться на основании специфики задачи до начала поиска.

Правило 2. Гриди алгоритм

Точка X_{i+1} выбирается из условия

$$\lambda(X_{i+1}) = \max_j \lambda(X^j),$$

где $X^j \in O_i(X^0) \cap O_1(X_i) \cap S$ для прямого алгоритма и $X^j \in O_{n-i}(X^0) \cap O_1(X_i)$ для двойственного.

Функция $\lambda(X)$ выбирается исходя из специфики задачи, например:

целевая функция $\lambda(X) = C(X)$,

удельная ценность $\lambda(X) = C(X)/A(X)$ (для одного ограничения) и т.д.

Правило 3. Адаптивный случайный поиск граничных точек (АСПГ)

Точка X_{i+1} выбирается случайным образом в соответствии с вектором вероятностей

$$P^i = (p_1^i, p_2^i, ..., p_J^i),$$

где J – количество точек, из которых производится выбор.

$$p_j^i = \frac{\lambda(X^j)}{\sum_{l=1}^{J} \lambda(X^l)}, \; j = \overline{1, J},$$

где $X^j \in O_i(X^0) \cap O_1(X_i) \cap S$ для прямого алгоритма и $X^j \in O_{n-i}(X^0) \cap O_1(X_i)$ для двойственного.

Правило 4. Модифицированный случайный поиск граничных точек (МСПГ).

Точка X_{i+1} выбирается из условия

$$\lambda(X_{i+1}) = \max_r \lambda(X^r),$$

где X^r – точки, выбранные по правилу 1, $r = \overline{1, R}$; R – задаваемый параметр алгоритма.

Приведенные выше регулярные и стохастические алгоритмы переводят исходную задачу условной псевдобулевой оптимизации, являющуюся *NP*-трудной, в задачу

$$\frac{f(X^*) - f(X')}{f(X^*)} \le \varepsilon,$$

принадлежащую классу P (полиномиально разрешима), где X^* – оптимальное решение, X' – приближенное решение.

Разработанные приближенные алгоритмы (гриди алгоритмы и алгоритмы случайного поиска граничных точек) имеют полиномиальные оценки трудоемкости и поэтому предназначены, прежде всего, для решения задач большой размерности. Наиболее эффективной представляется следующая схема. На первом этапе задача решается гриди алгоритмом, а затем находятся несколько решений с помощью адаптивного случайного поиска; лучшее из найденных принимается за решение задачи.

5.4 Схема метода ветвей и границ

В настоящем разделе описываются алгоритмы оптимизации для решения задач, рассмотренных в предыдущих главах, и предлагается алгоритм условной псевдобулевой оптимизации, основанный на поиске среди граничных точек и методе ветвей и границ. Использование свойств задачи и схемы метода ветвей и границ позволяет быстро исключать из рассмотрения области, в которых нет оптимального решения. Метод ветвей и границ используется разными исследователями для широкого круга задач, например, для задач размещения [213].

5.4.1 Схема метода ветвей и границ для задачи с алгоритмически заданными функциями

Метод ветвей и границ основан на идее разумного перечисления всех допустимых точек комбинаторной задачи оптимизации.

Впервые метод ветвей и границ был предложен Land и Doig [214] в 1960 году для решения общей задачи целочисленного линейного программирования. Интерес к этому методу и фактически его “второе рождение” связано с работой [215], посвященной задаче коммивояжера. Начиная с этого момента, появилось большое число работ, посвященных методу ветвей и границ и различным его модификациям. Столь большой успех объясняется тем, что авторы первыми обратили внимание на широту возможностей метода, отметили важность использования специфики задачи и сами воспользовались спецификой задачи коммивояжера.

В основе метода ветвей и границ лежит идея последовательного разбиения множества допустимых решений на подмножества. На каждом шаге метода элементы разбиения подвергаются проверке для выяснения, содержит данное подмножество оптимальное решение или нет. Проверка осуществляется посредством вычисления оценки снизу для целевой функции на данном подмножестве. Если оценка снизу не лучше *рекорда* — наилучшего из найденных решений, то подмножество может быть отброшено. Проверяемое подмножество может быть отброшено еще и в том случае, когда в нем удается найти наилучшее решение. Если значение целевой функции на найденном решении лучше рекорда, то происходит смена рекорда. По окончанию работы алгоритма рекорд является результатом его работы.

Если удается отбросить все элементы разбиения, то рекорд — оптимальное решение задачи. В противном случае, из неотброшенных подмножеств выбирается наиболее перспективное (например, с наименьшим значением нижней оценки), и оно подвергается разбиению. Новые подмножества вновь подвергаются проверке и т.д.

Очевидно, что именно использование специфических структурных особенностей задачи дает возможность построить работоспособный алгоритм ветвей и границ. Рассматривается применение схемы для решения оптимизационной задачи, в которой все переменные являются булевыми, а целевая функция и ограничение унимодальны и монотонны:

$$\begin{cases} C(X) \to \max\limits_{X \in B_2^n}, \\ A(X) \le H, \end{cases}$$

где $C(X)$ и $A(X)$ - унимодальные и монотонно возрастающие от $X^0 = (x_1^0,...,x_n^0)$ псевдобулевые функции.

Наиболее часто используемый вариант применения схемы метода ветвей и границ для решения задачи псевдобулевой оптимизации заключается в следующем [216, 217]. Решается задача непрерывной оптимизации (как правило, симплекс-алгоритмом), являющаяся ослаблением исходной задачи, и получается решение X^*, которое в общем случае не будет булевым. После этого задачу разбивают на две подзадачи, добавляя два взаимоисключающих и исчерпывающих все возможности ограничения. Пусть, например, компонента x_i' в X^* не булева. Тогда в соответствующих подзадачах появляются ограничения $x_i' = 0$ и $x_i' = 1$. Дальнейшее ветвление происходит подобным образом.

Такой алгоритм может быть довольно эффективным для задач, в которых целевая функция и ограничение определены в явном виде [218]. В реальных задачах очень часто целевая функция задана алгоритмически, т.е. невозможно вычислить значение функции в точке, не являющейся булевой. Поэтому возникла необходимость исследования других вариантов применения схемы.

Далее мы рассмотрим вопрос применения схемы ветвей и границ для задач оптимизации, в которых целевая функция и ограничения заданы алгоритмически. А именно, для задачи (1)-(2), в которой функции C(X) и A(X) монотонно возрастают от точки $X^0 = (x_1^0,...,x_n^0)$.

Простейший алгоритм схемы метода ветвей и границ, основанный на свойствах рассматриваемого класса задач, будет выглядеть следующим образом. Первоначально имеется все множество пространства булевых переменных B_2^n мощностью 2^n. При погружении [219] B_2^n в n-мерное евклидово пространство его

элементам будут соответствовать вершины единичного гиперкуба, одна из которых совпадает с началом координат. На первом этапе ветвления множество B_2^n делится на два равномощных подмножества: $S_1^0 = \{X \in B_2^n : x_1 = 0\}$ и $S_1^1 = \{X \in B_2^n : x_1 = 1\}$ (назовем это ветвлением первого порядка). Показано, что каждое из этих подмножеств является подкубом размерностью $n-1$, мощность подмножеств 2^{n-1}. На следующих этапах ветвления каждый из подкубов разбивается на два подкуба, и т.д. (рисунки 5.10-5.12). Например, S_1^0 делится на $S_2^{00} = \{X \in B_2^n : x_1 = 0, x_2 = 0\}$ и $S_2^{01} = \{X \in B_2^n : x_1 = 0, x_2 = 1\}$. Так, после ветвления *k*-го порядка получаются подкубы, состоящие из 2^{n-k} элементов (векторов).

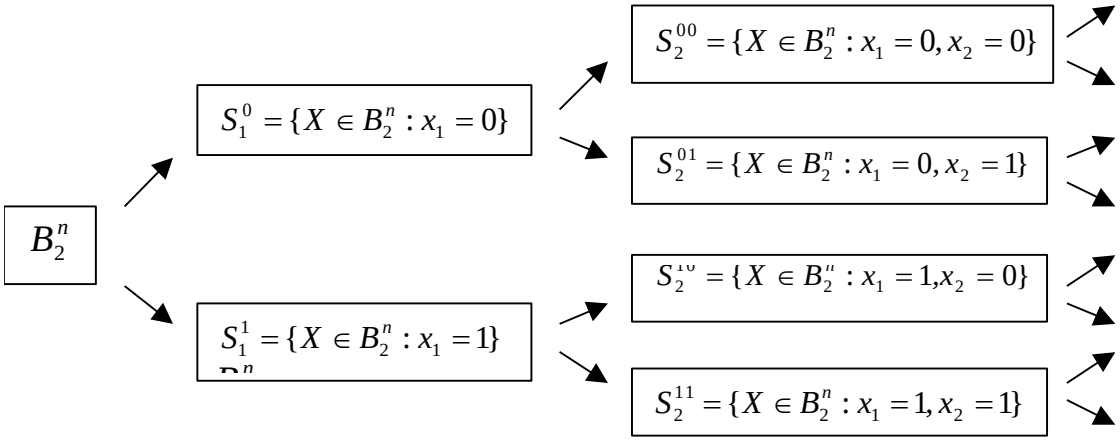


Рисунок 5.10 – Схема ветвления

В подмножестве, полученном после ветвления *k*-го порядка, *k* координат $x_1,\ldots,x_k$ являются постоянными для любого булева вектора из этого подмножества. Назовем *«нижней» точкой* $\underline{X}$ – вектор, в котором переменные координаты равны соответствующим координатам начального вектора X^0, а *«верхней» точкой* $\overline{X}$ подкуба вектор, в котором все переменные координаты противоположны соответствующим координатам X^0:

$$\overline{X} = (x_1,\ldots,x_k,1-x_{k+1}^0,1-x_{k+2}^0,\ldots,1-x_n^0),$$

$$\underline{X} = (x_1,\ldots,x_k, x_{k+1}^0, x_{k+2}^0,\ldots,x_n^0).$$

Целевая функция и функция ограничения монотонно возрастают на B_2^n при выбранной начальной точке $X^0 = (x_1^0,\ldots,x_n^0)$, откуда следует, что в «верхней» точке подкуба они принимают наибольшее значение, а в «нижней» точке – наименьшее.

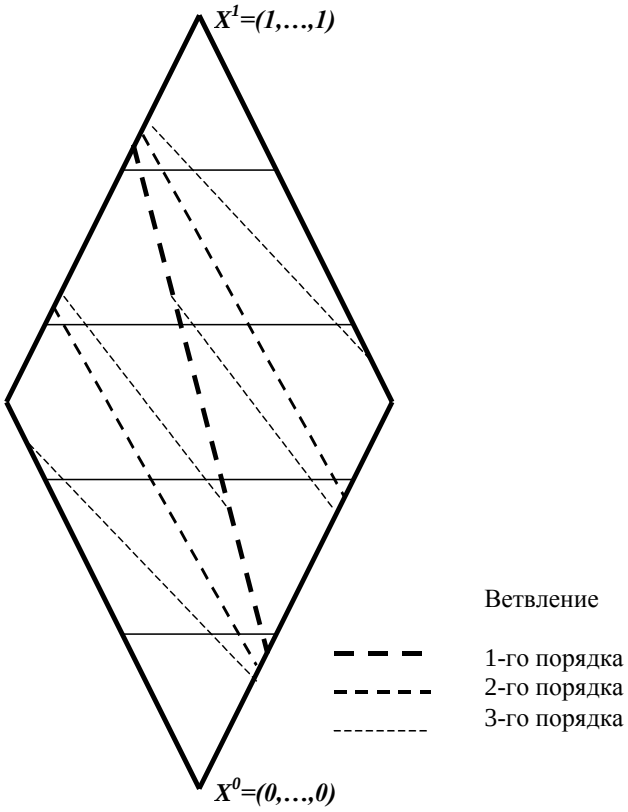


Рисунок 5.11 – Условное представление схемы разбиения пространства B_2^n на подкубы, n=5.

Подмножество (подкуб) исключается из рассмотрения в трех случаях:

1. В точке $\underline{X}$ ограничение не выполняется; в этом случае все решения в подмножестве являются недопустимыми.

2. В точке $\overline{X}$ ограничение выполняется; тогда это решение является наилучшим в подмножестве, и оно сравнивается с рекордом.

3. В точке $\overline{X}$ ограничение не выполняется, но целевая функция в ней принимает значение, которое меньше рекорда.

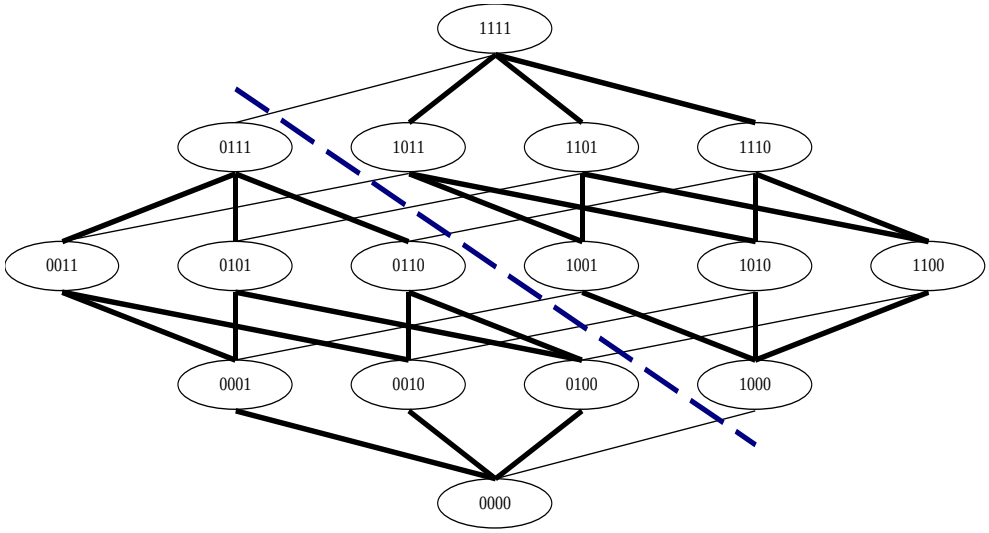


Рисунок 5.12 – Разбиение элементов пространства B_2^n, n=4, на два подкуба.

В противном случае происходит дальнейшее ветвление этого подмножества.

На первом этапе в качестве рекорда можно взять значение целевой функции в любой допустимой точке пространства B_2^n. Если при рассмотрении подмножества $K(\underline{X}, \overline{X})$ точка $\overline{X}$ является допустимой, и значение критериальной функции в ней больше рекорда, то происходит смена рекорда.

Легко показать, что полученное решение будет являться точным. Ограничение $A(X) \le H$ разбивает множество B_2^n на два подмножества, одно из которых удовлетворяет ограничению, а другое нет. Из условия монотонности функций $C(X)$ и $A(X)$ следует, что решением задачи будет точка, принадлежащая подмножеству крайних точек.

В первом случае в подкубе $K(\underline{X}, \overline{X})$ отсутствуют граничные точки. Во втором случае в подкубе лишь точка $\overline{X}$ может быть граничной. В третьем случае в подкубе содержатся граничные точки, но они хуже, чем найденные ранее. Таким образом, эта схема обеспечивает точное решение задачи.

Рассмотренный подход дает возможность быстрого вычисления нижней границы, которая равна значению целевой функции в «верхней» точке подкуба.

К сожалению, не удается теоретически получить эффективную оценку трудоемкости такого алгоритма. Наибольшее число подмножеств, которые могут быть просмотрены в ходе работы алгоритма, является показательной функцией от размерности задачи. Нельзя заранее установить, какая часть просматриваемых подмножеств будет проверена. Тем не менее, с помощью нижней границы, построенной описанным выше способом, удается отсекать большое число подмножеств, что делает этот алгоритм довольно эффективным.

Описанный подход хотя и позволяет существенно сократить число перебираемых точек при нахождении оптимального решения, но всё же достаточно трудоемок, так как может требовать большое число ветвлений.

В следующем разделе описывается алгоритм оптимизации, совмещающий схему метода ветвей и границ и правила отсечения подкубов, изложенные в предыдущем разделе.

5.4.3 Алгоритм оптимизации, основанный на схеме ветвей и границ и поиске крайних точек

Рассмотрим класс задач вида

$$\begin{cases} C(X) \to \max\limits_{X \in B_2^n} \\ A_j(X) \le H_j,\ j = \overline{1, m} \end{cases}$$

где $B_2^n = \{0,1\}^n$ – пространство бинарных переменных, $C(X)$ и $A_j(X)$ – псевдобулевые функции (действительные функции бинарных переменных), в общем случае заданные алгоритмически, H_j – некоторые действительные числа. Рассмотрим класс задач, в котором целевая функция $C(X)$ и функция $A(X)$, определяющая систему ограничений, монотонно возрастают от точки $X^0 = (x_1^0,...,x_n^0)$. К этому классу задач относятся, в частности, задачи поиска оптимальных закономерностей, рассмотренные в третьей главе.

Обозначим $X^1 = (x_1^1,...,x_n^1) = (\overline{x_1^0},...,\overline{x_n^0})$. Всё множество точек пространства B_2^n можно представить в виде подкуба $K(X^0, X^1)$ - это исходный подкуб, в котором выполняется поиск оптимального решения.

5.4.4 Ветвление

Допустим, что найдена некоторая крайняя точка $X' \in B_2^n$. Тогда подкубы $K(X', X^0)$ и $K(X', X^1)$ можно исключить из дальнейшего рассмотрения [220].

Введём вспомогательную переменную

$$z_i = \begin{cases} x_i, & \text{if } x_i^0 = 0, \\ \bar{x}_i, & \text{if } x_i^0 = 1. \end{cases}$$

Тогда подкуб $K(X', X^0)$ можно представить как множество точек, для которых выполняется

$$T^0 = \bigwedge_{i:x_i'=x_i^0} \bar{z}_i.$$

А подкуб $K(X', X^1)$ можно описать выражением

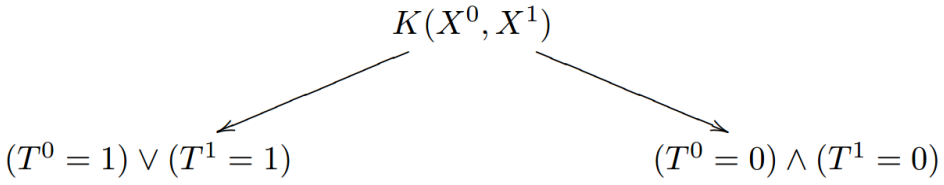
$$T^1 = \bigwedge_{i:x'_i=x_i^1} z_i.$$

Для удобства множество индексов, для которых выполняется $x'_i = x_i^1$ обозначим $A(X') = \{i_1,...,i_k\}$, а множество индексов, для которых выполняется $x'_i = x_i^0$ обозначим $B(X') = \{i_1,...,i_{n-k}\}$. Очевидно, что $|A(X')| = k|$ и $|B(X')| = n-k$, где k - номер уровня, на котором расположена точка $K(X') \in O_k(X_0)$. Тогда можно записать:

$$T^0 = \bigwedge_{i \in B(X')} \bar{z}_i$$

$$T^1 = \bigwedge_{i \in A(X')} z_i$$

Разделим подкуб $K(X^0,X^1)$ на две части.



Левая часть, как было сказано выше, исключается из дальнейшего рассмотрения. Правая часть $(T^0=0)\wedge(T^1=0)$ может быть представлена как множество подкубов.

Рассмотрим условие $(T^1=0)$. Оно выполняется в том случае, когда $z_i = 0$ хотя бы для одного $i \in A(X')$. При $|A(X')|>1$ множество точек, выполняющих условие $T^1 = 0$, не может быть представлено в виде одного подкуба, а только как набор подкубов. Наиболее очевидный способ разбить это множество точек на k подкубов - зафиксировать поочередно значение переменной $z_i = 0$ для $i \in A(X')$. В

этом случае получаем k подкубов размерностью $n-1$. Недостаток этого способа состоит в том, что полученные подкубы существенно пересекаются между собой.

Для того, чтобы этого избежать, используем следующий подход. Первый подкуб K_1^1 получим, зафиксировав одну переменную: $z_{i_1}=0$. Для второго K_2^1 зафиксируем две переменных: $z_{i_1}=1$ и $z_{i_2}=0$. Для третьего K_3^1 три перменных: $z_{i_1}=1$, $z_{i_2}=1$ и $z_{i3}=0$. И так далее. Для k-го подкуба K_k^1: $z_{i_s}=1$, $s=1,...,k-1$, $z_{i_k}=0$.

В результате получаем k подкубов различной размерности. Такой подход гарантирует, что получаемые подкубы не пересекаются.

То же самое проделаем для условия (T^0=0). Соответствующее множество точек следует разбить на ($n-k$) подкубов путем фиксации переменных $j\in B(X')$. Для первого подкуба K_1^0 зафиксируем одну переменную: $z_{j_1}=1$. Для второго K_2^0 зафиксируем две переменных: $z_{j_1}=0$ и $z_{j_2}=1$. Для третьего K_3^0 три перменных: $z_{j_1}=0$, $z_{j_2}=0$, $z_{j_3}=1$. Для ($n-k$)-го подкуба K_{n-k}^0. $z_{j_s}=0$, $s=1,...,n-k-1$, $z_{j_{n-k}}=1$.

В результате получаем два набора подкубов $K_1^1,...,K_k^1$ и $K_1^0,...,K_{n-k}^0$. Множество точек, выполняющих условие (T^0=0)$\wedge$(T^1=0), соответствует объединению всевозможных пересечений пар подкубов, взятых из этих двух наборов:

$$\bigcup_{\substack{i\in A(X')\\ j\in B(X')}}\left(K_i^1\cap K_j^0\right).$$

Итак, найдя в подкубе $K(X^0,X^1)$ некоторую крайнюю точку $X'\in O_k(X^0)$, этот подкуб разбивается на две части, одна из которых отбрасывается (подкубы $K(X', X^0)$ и $K(X', X^1)$), а из другой образуются $k\cdot(n-k)$ новых ветвей. Каждая из этих ветвей является подкубом, с которым может быть произведена такая же процедура ветвления, что описана выше.

5.4.5 Верхняя граница

Обозначим $\overline{X}$ и $\underline{X}$ соответственно верхнюю и нижнюю точки некоторого подкуба. В точке $\underline{X}$ для всех свободных (незафиксированных) переменных выполняется $z_i^{\underline{X}} = 0$, в точке $\overline{X}$ для всех свободных переменных $z_i^{\overline{X}} = 1$. Для зафиксированных переменных, естественно, $z_i^{\underline{X}} = z_i^{\overline{X}}$.

Подкуб $K(\underline{X}, \overline{X})$ может содержать оптимальное решение, только если выполняются условия:

1. В подкубе имеются допустимые решения.
2. Верхняя граница соответствующей ветви выше найденного наилучшего решения.

Так как функция ограничения $A(X)$ монотонно возрастает от точки X^0, то в пределах подкуба $K(\underline{X}, \overline{X})$ функция $A(X)$ монотонно возрастает от точки $\underline{X}$, принимая в ней минимальное значение. Поэтому если точка $\underline{X}$ является недопустимой, то недопустимы все точки этого подкуба.

Целевая функция $C(X)$ в пределах подкуба также монотонно возрастает от точки $\underline{X}$, принимая наибольшее значение в точке $\overline{X}$. Сама эта точка может быть и недопустима, но значение $C(\overline{X})$ может быть использовано как верхняя граница ветви, соответствующей этому подкубу.

Также, если точка $\overline{X}$ является допустимой, то все остальные точки этого подкуба заведомо не лучше, кроме того, в подкубе при этом отсутствуют крайние точки за исключением разве что $\overline{X}$.

Итак, подкуб $K(\underline{X}, \overline{X})$ исключается из дальнейшего поиска при выполнении хотя бы одного из условий:

- Точка $\underline{X}$ является недопустимой.
- Точка $\overline{X}$ является допустимой.

- Значение верхней границы $C(\overline{X})$ не превышает уже найденного наилучшего допустимого значения целевой функции.

Такая проверка, включая вычисление верхней границы, требует просмотра всего двух точек подкуба.

5.4.6 Поиск граничных точек

Для того чтобы произвести ветвление описанным выше способом, необходимо найти некоторую крайнюю точку, принадлежащую рассматриваемому подкубу $K(\underline{X}, \overline{X})$. Это не обязательно должно быть наилучшее решение в данном подкубе. Тем не менее, хорошее решение может повысить рекорд (наилучшее найденное допустимое решение) и улучшает отсев новых ветвей (если их верхняя граница окажется ниже). Для нахождения крайней точки используется жадный алгоритм, также могут использоваться алгоритмы случайного поиска [73, 82, 182].

Простейший алгоритм случайного поиска крайних точек состоит в следующем. Поиск начинается из $\underline{X}$. На каждом шаге алгоритм выбирает допустимую соседнюю точку на следующем уровне, двигаясь, таким образом, по пути возрастания целевой функции до границы допустимой области. При необходимости процедура повторяется несколько раз, и из найденных крайних точек выбирается лучшая.

Задаваемое число L – количество крайних точек, которое планируется найти. Так как $card\{O_1(X_k) \cap O_{k+1}(\underline{X})\} = n_K - k$, где $X_k \in O_k(X)$ (лемма 5.3), n_K - размерность подкуба $K(\underline{X}, \overline{X})$ (число свободных переменных), то из текущей точки поиска X_k алгоритм просматривает не более $n_K - k$ следующих точек. Таким образом, трудоемкость алгоритма

$$T \le L \cdot \sum_{i=0}^{n-1}(n_K - i) = L \cdot \frac{n_K(n_K+1)}{2}.$$

Алгоритм "Случайный поиск"

1. Полагаем $l = 1$.
2. Полагаем $X_1 = \underline{X}$, $i = 1$.
3. Случайным образом выбираем точку $X_{i+1} \in O_1(X_i) \cap O_i(\underline{X}) \cap \{X \in K(\underline{X}, \overline{X}) : A(X) \le H\}$, $i = i + 1$. Если таких нет, то переходим к п. 4, иначе цикл повторяется.
4. $Y_l = X_i$. Если $l < L$, то $l = l + 1$ и переходим к п. 2.
5. Определяем X^* из условия

$$C(X^*) = \max_{l=1,\dots,L} C(Y_l).$$

Альтернативой случайному поиску граничных точек является регулярный алгоритм, использующий гриди эвристику.

Гриди алгоритмы являются интуитивными эвристиками, в которых на каждом шаге принимается решение, являющееся наиболее выгодным в данный момент, без учета того, что происходит на последующих шагах поиска.

Для рассматриваемой задачи гриди алгоритм может иметь следующий вид.

Алгоритм "Гриди"

1. Полагаем $X_1 = \underline{X}$, $i = 1$.
2. Вычисляем $C(X_j)$ и $A(X_j)$ для $X_j \in O_1(X) \cap O_i(\underline{X})$, $j = 1, \dots, n_K - i + 1$.
3. Если нет X_j, для которых $A(X_j) \le H$, то $X^* = X$ - решение задачи.
4. Из тех X_j, для которых $A(X_j) \le H$, находим $X = \arg\max_{X_j} \lambda(X_j)$.
5. $i = i + 1$, переходим к п. 2.

Здесь $\lambda(X_j) = C(X_j)/A(X_j)$ либо $\lambda(X_j) = C(X_j)$.

5.4.7 Схема алгоритма

Описанные выше процедуры являются основными элементами, из которых состоит алгоритм поиска оптимального решения. Теперь рассмотрим сам алгоритм.

Первый шаг - это выбор ветви для ветвления. На первом цикле имеется только одна ветвь, которая соответствует бинарному пространству размерности n. На следующих циклах, когда имеется несколько открытых ветвей, выбирается ветвь с наибольшей верхней оценкой. Если открытых ветвей нет, то алгоритм завершает свою работу.

На втором шаге производится поиск приближенного решения в выбранной ветви, представляющего собой некоторую крайнюю точку в соответствующем подкубе. Если значение целевой функции в этой точке лучше рекорда (наилучшего уже найденного решения), то происходит смена рекорда. Поиск крайней точки можно осуществлять, к примеру, с помощью алгоритма случайного поиска или гриди алгоритма, описанных выше.

На третьем шаге выполняется процедура ветвления выбранной ветви в соответствие с найденной крайней точкой. Производится проверка полученных ветвей: если в ветви имеются допустимые решения и верхняя граница больше рекорда, то эта ветвь добавляется в список открытых ветвей.

После выполнения некоторого количества таких циклов следует сделать прерывание: произвести сортировку ветвей по значению верхней границы и закрыть ветви, верхняя граница которых меньше рекорда.

Остановка поиска осуществляется, если не осталось открытых ветвей. В этом случае можно утверждать, что найдено точное решение задачи (глобальный условный максимум).

При решении задач больших размерностей это может быть недостижимо ввиду чрезмерно огромного времени поиска. Критерием остановки может служить достижение числа образованных ветвей либо числа ветвлений некоторого заданного значения.

Схематично алгоритм изображен на рисунке 5.13.

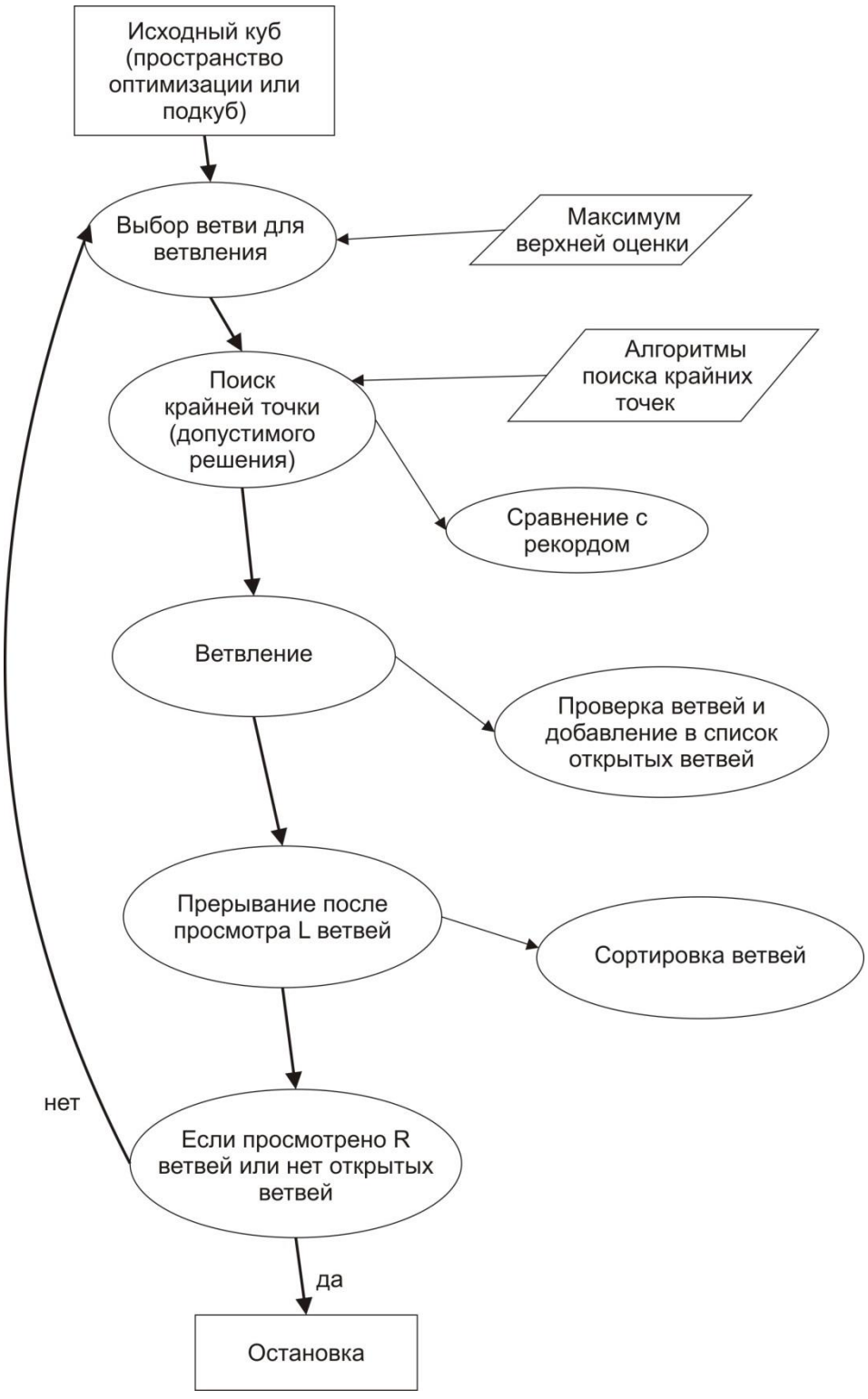


Рисунок 5.13 – Схема алгоритма

5.4.8 Экспериментальные исследования

Здесь приведены результаты экспериментального исследования работы описанного алгоритма на задачах условной псевдобулевой оптимизации, сгенерированных случайным образом. Целевые функции и ограничения имеют следующий вид:

$$C(X) = \sum_{i=1}^{n} c_1^i x_i + \sum_{i=1}^{n-1} c_2^i x_i x_{i+1} + \sum_{i=1}^{n-2} c_3^i x_i x_{i+1} x_{i+2} \to \max,$$

$$A(X) = \sum_{i=1}^{n} a_1^i x_i + \sum_{i=1}^{n-1} a_2^i x_i x_{i+1} + \sum_{i=1}^{n-2} a_3^i x_i x_{i+1} x_{i+2} \le b,$$

$$X \in \{0,1\}^n,$$

где коэффициенты $c_1^i, c_2^i, c_3^i, a_1^i, a_2^i, a_3^i$ - случайные числа, взятые из диапазона [0,20]; $b = A(X^r)$, где X^r - случайно выбранная точка. Так как все коэффициенты неотрицательные числа, то функции $C(X)$ и $A(X)$ являются монотонными функциями с минимумом в точке (0,…,0) и безусловным максимумом в точке (1,…,1).

Эффективность алгоритма будем характеризовать трудоемкостью и достигнутым значением целевой функции (если максимум целевой функции определен неточно или нет доказательств того, что решение является точным). Под трудоемкостью (или временной сложностью) мы будем понимать число вычислительных значений целевой функции (и/или функции ограничения), которые выполнил алгоритм (количество точек, которые алгоритм «просмотрел»).

Сначала рассмотрим, насколько быстро алгоритм оптимизации находит точное решение. Для этого была проведена серия испытаний по задачам малой размерности: $n = 20$. Для каждого значения измерения были решены 500 задач. Для нахождения граничных точек был использован простой алгоритм "случайного поиска" с числом повторений $L = 1$.

На рисунках 5.14 и 5.15 приведены результаты экспериментального исследования работы алгоритма на задачах условной псевдобулевой оптимизации, сгенерированных случайным образом.

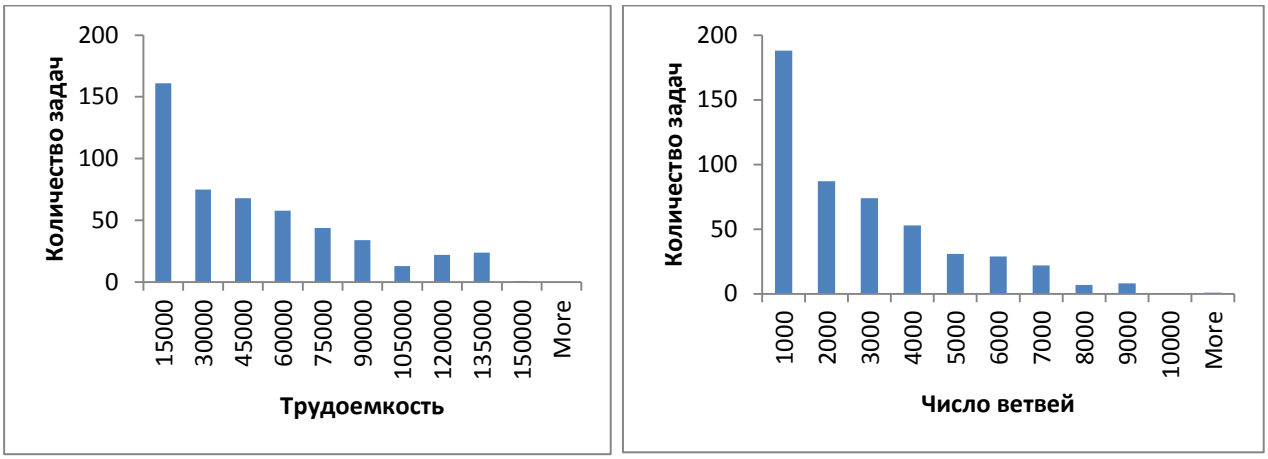


Рисунок 5.14 - Трудоемкость и число ветвей для $n = 20$ (точное решение найдено)

На гистограммах показаны распределения трудоемкости и числа ветвей, полученных в результате точного решения 500 задач размерностью 20. Гистограммы рисунка 5.14 относятся к полному решению задач, то есть не остается открытых ветвей, и, таким образом, точность решения доказана. Гистограммы рисунка 5.15 показывают трудоемкость и число ветвей для момента, когда наилучшее решение уже найдено, но точность еще не доказана, то есть остались открытые ветви.

Как видно из графиков, точное решение обычно найдено задолго до завершения полного поиска.

Следует отметить, что результаты решений задач, сгенерированных таким образом, сильно различаются от задачи к задаче, поэтому не имеет смысла приводить средние значения показателей эффективности. Вместо этого приводятся результаты решения отдельных задач, что в данном случае является более наглядным.

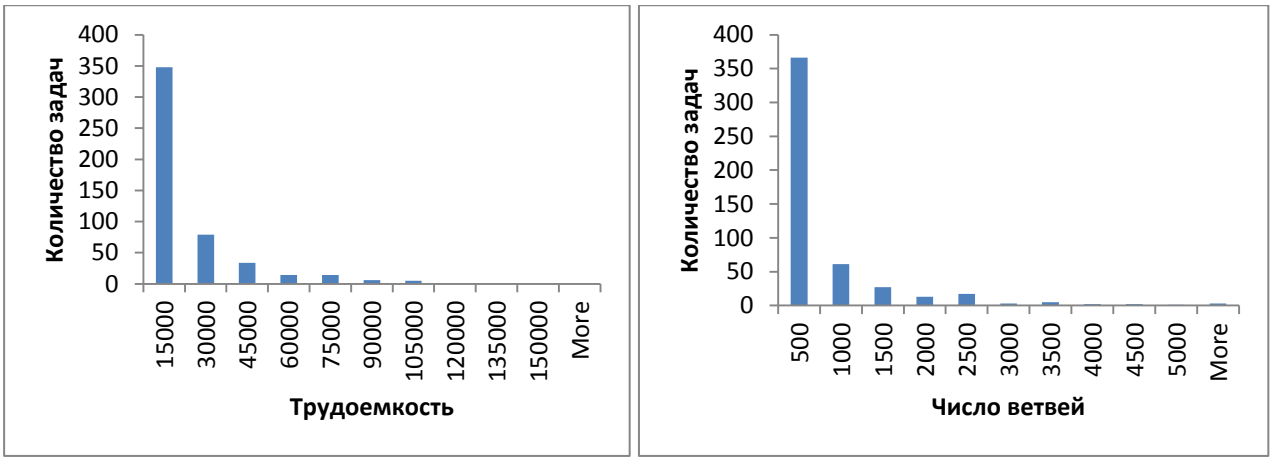


Рисунок 5.15 - Трудоемкость и число ветвей для $n = 20$ (точность не доказана)

В ходе поиска количество открытых ветвей существенно меняется. Первые несколько циклов оно быстро увеличивается и к концу поиска постепенно уменьшается, приближаясь к нулю. Отсутствие открытых ветвей по завершении поиска означает, что решение является точным, то есть лучшего решения в данных условиях не существует.

На рисунке 5.16 графики показывают изменение числа открытых ветвей и значения лучшего найденного решения в процессе поиска. Представлены результаты для отдельно взятых задач размерностью 20 и 100, но это является типичной картиной.

Номер ветвления показан по оси абсцисс, количество открытых ветвей - слева по оси ординат, значение рекорда - справа по оси ординат.

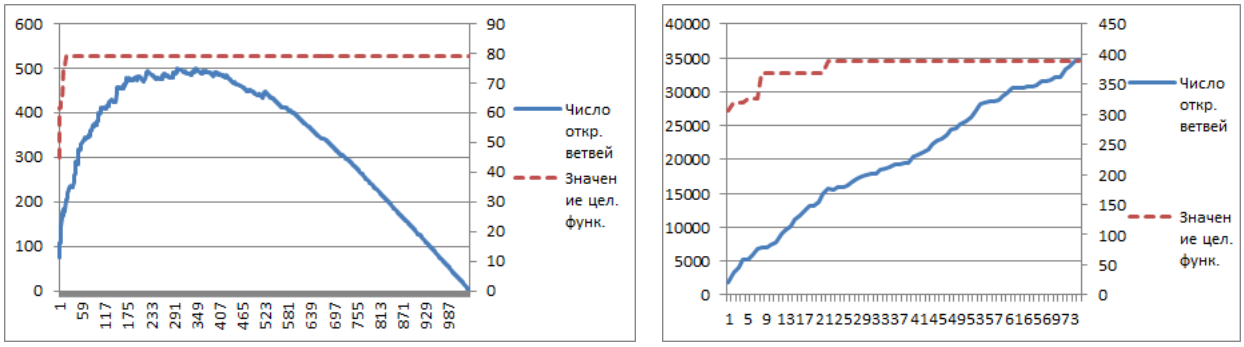


Рисунок 5.16 - Число открытых ветвей и значение рекорда для $n = 20$ и n=100.

Как видно, лучшее найденное значение возрастает на первых ветвлениях, достигая, возможно, оптимального значения, а затем не меняется. Остальная часть поиска нужна лишь для подтверждения оптимальности найденного решения.

Таким образом, из практических соображений, достаточно произвести лишь несколько итераций алгоритма, чтобы значительно улучшить результат жадного алгоритма. Идея улучшения приближенного алгоритма, после чего он способен выдавать решение равное или очень близкое оптимальному, ранее уже применялась, например, в алгоритме локального поиска по обобщенной окрестности для оптимизации псевдобулевых функций [221].

Также был рассмотрен вопрос о том, как предлагаемый алгоритм оптимизации может улучшить решение, полученное индивидуально алгоритмом поиска граничных точек (жадный эвристический или случайный поиск граничных точек).

Для нахождения первого приближенного решения использовался жадный алгоритм оптимизации, описанный выше. Полученное решение использовалось в качестве точки ветвления в соответствии с процедурой, описанной выше для ветвления пространства оптимизации. Для поиска решений в сформированных ветвях также использовался жадный алгоритм. Значения лучших решений, найденных при поиске алгоритмом ветвей и границ, представлены в таблицах 5.2-5.6.

Таблица 5.2 - Жадный алгоритм и алгоритм ветвей и границ, $n = 10$

Число ветвлений	Найденное решение	Трудоемкость
Жадный алгоритм	47	34
1	47	82
2	56	141
4	58	202
7	73	281

Таблица 5.3 - Жадный алгоритм и алгоритм ветвей и границ, $n = 20$

Число ветвлений	Найденное решение	Трудоемкость
Жадный алгоритм	31	90
1	52	216
2	53	341
4	66	492
8	67	862
19	68	1703
32	74	2645
88	78	5403
153	79	8840

Таблица 5.4 - Жадный алгоритм и алгоритм ветвей и границ, $n = 30$

Число ветвлений	Найденное решение	Трудоемкость
Жадный алгоритм	74	165
1	74	407
2	76	707
5	89	1529
12	106	2850
34	108	6161
139	109	19937
183	111	24662
541	118	54243
589	119	58145
1515	123	126962

Таблица 5.5 - Жадный алгоритм и алгоритм ветвей и границ, $n = 100$

Число ветвлений	Найденное решение	Трудоемкость
Жадный алгоритм	342	1810
1	342	4874
3	371	12318
8	374	26684
35	388	88616
67	401	158873
70	406	163781

Таблица 5.6 - Жадный алгоритм и алгоритм ветвей и границ, n = 200

Число ветвлений	Найденное решение	Трудоемкость
Жадный алгоритм	708	7380
1	743	20153
2	784	22181
4	798	35944
7	800	70062
11	803	112484
12	840	114011

Выводы к главе 5

В процессе управления сложными техническими и организационными системами часто возникает необходимость оптимизации по определенным критериям посредством выбора наилучших значений управляемых переменных из области допустимых значений, определяемой рядом ограничений на переменные. Причем эти критерии и ограничения во многих встречающихся на практике случаях не удается записать в виде явных алгебраических выражений, т.е. они заданы алгоритмически, а упрощения моделей, такие как линеаризация, оказываются неприемлемыми для оптимизации при управлении сложными техническими системами. Это делает невозможным применение хорошо изученных методов математического программирования и соответствующих программных приложений.

Следует отметить, что известные поисковые методы не учитывают особенности ограничений, да и целевых функций, наблюдаемые, как правило, в реальных задачах. Адаптивные поисковые алгоритмы типа генетических, к тому

же, требуют подбора некоторого числа параметров, что возможно только при предварительном решении ряда однотипных задач и неприемлемо при единичном решении задачи, и не дают никакой гарантии об исключении полного перебора.

Значительную часть в задачах дискретной оптимизации занимают задачи оптимизации с булевыми переменными. Строгая булевость – свойство, которое часто встречается в практических задачах оптимизации. Задачам оптимизации с булевыми переменными, как с алгоритмическим, так и с алгебраическим заданием функций, многие исследователи уделяют особое внимание. В принципе, любая задача дискретной оптимизации сводится к псевдобулевой.

Проблема повышения эффективности решения задач условной псевдобулевой оптимизации с алгоритмически заданными функциями является весьма актуальной. Наибольшая эффективность может быть достигнута путем выявления топологических свойств пространства булевых переменных и свойств псевдобулевых функций и их реализации при построении математического аппарата решения выделенных классов задач.

Разработанные алгоритмы оптимизации применимы для любых задач условной псевдобулевой оптимизации, возникающих в процессе управления сложными техническими объектами, а также в областях экономики. Описанные алгоритмы использованы в решении задачи оптимизации загрузки технологического оборудования предприятия, представляющей собой задачу условной псевдобулевой оптимизации с несвязным множеством допустимых решений [222].

Главная особенность рассматриваемого класса задач состоит в том, что целевая функция и функции ограничений предполагаются заданными неявно, то есть возможны лишь вычисления функций в точках, но не известна их алгебраическая запись. С одной стороны, такие задачи часто встречаются на практике, например, когда для вычисления функции необходимо обратиться к массиву данных. С другой стороны, даже для тех задач, для которых возможна алгебраическая запись функций, эти функции можно рассматривать как заданные

алгоритмически, что значительно упрощает работу с имеющейся оптимизационной моделью.

Такой класс задач ограничивает перечень доступных для применения алгоритмов оптимизации. Конечно, всегда можно применить алгоритм локального поиска или алгоритмы генетического типа, но они не гарантируют нахождения точного решения, и нельзя сказать, насколько найденное решение близко к оптимальному.

При этом во многих практических задачах целевые функции и ограничения обладают одними и теми же свойствами, такими как унимодальность и монотонность. И эти свойства не учитываются при применении универсальных алгоритмов.

Изложенный в этой работе подход направлен на получение точного решения задачи оптимизации. Реализованный способ ветвления делит ветвь, представляющую собой подкуб пространства бинарных переменных, на большое число ветвей, значительная часть которых сразу же подлежит исключению. Это позволяет быстро уменьшать область, в которой еще может находиться оптимальное решение.

Разработанный алгоритм может применяться и для решения задач больших размерностей. При этом, конечно, не будет доказано, что найденное решение является оптимальным, если еще остаются непросмотренные открытые ветви. В таком случае этот алгоритм можно рассматривать как улучшение приближенных алгоритмов поиска граничных точек, таких как жадный алгоритм и случайный поиск граничных точек. И такое улучшение даже на небольшом числе итераций (ветвлений) позволяет значительно улучшить найденное допустимое решение.

Наиболее интересны результаты применения разработанных алгоритмов оптимизации в методах распознавания, в частности, в логических алгоритмах классификации [223]. Применение эффективных алгоритмов комбинаторной оптимизации позволяет находить информативные закономерности в данных, что ведет к получению эффективных решающих правил.

Излагаемый метод оптимальных логических решающих правил заключается, прежде всего, в выявлении логических закономерностей в данных, которые являются сильными первичными закономерностями, то есть максимальными по отношениям доказательности и простоты.

Новый алгоритм условной псевдобулевой оптимизации на основе схемы метода ветвей и границ и поиска среди граничных точек допустимой области обеспечивает нахождение сильных первичных закономерностей за приемлемое время (жадный алгоритм находит лишь первичные закономерности). Экспериментально показана также эффективность приближенного варианта алгоритма с использованием ранней остановки: достаточно произвести лишь несколько итераций алгоритма, чтобы получить решение, лучшее, чем получаемое жадным алгоритмом.

ГЛАВА 6. ПРАКТИЧЕСКОЕ ПРИМЕНЕНИЕ МЕТОДА ОПТИМАЛЬНЫХ ЛОГИЧЕСКИХ РЕШАЮЩИХ ПРАВИЛ

В настоящей главе рассматривается применение метода оптимальных логических решающих правил на практических задачах. Рассмотрены вопросы применения разработанного метода к задаче классификации электрорадиоизделий космического применения в рамках проектов: «Разработка алгоритмического обеспечения анализа однородности партий электрорадиоизделий для комплектации радиоэлектронной аппаратуры космических аппаратов», «Разработка методологии и методов обеспечения работоспособности электронной компонентной базы в космических аппаратах длительного функционирования», «Отработка программ испытаний, проведение дополнительных отбраковочных испытаний электронной компонентной базы (ЭКБ) с целью прогнозирования работоспособности для космических аппаратов с длительным сроком активного существования». Приведенные результаты, являющиеся примером практического применения метода оптимальных логических решающих правил, внедрены в эксплуатацию в составе системы классификации электрорадиоизделий космического применения в ОАО ИТЦ – НПО ПМ (г.Железногорск, см. Приложение А).

Также описано применение метода к выявлению логических закономерностей в данных историй болезни пациентов, больных инфарктом миокарда, и прогнозированию осложнений заболевания.

Экспериментальные исследования по сравнению точности распознавания на популярных задачах подтверждают конкурентоспособность предлагаемого метода. В таблице 6.1 приведены результаты сравнения точности распознавания на 10 известных наборах данных, взятых из репозитория задач машинного обучения. Приведено сравнения с 5 алгоритмами – логистическая регрессия, деревья решений, случайный лес, искусственная нейронная сеть, метод опорных векторов. Для сравнения с другими методами использовалось программное приложение Weka.

Таблица 6.1 - Сравнение точности на тестовых задачах

Набор данных	Логист. регр.	C4.5	Случ. лес	Нейр. сеть	SVM	ОЛРП
boston	0,87	0,84	0,87	**0,89**	**0,89**	0,87
bupa	0,66	0,63	0,73	0,64	0,70	**0,74**
breast c.w.	0,96	0,94	**0,97**	0,95	0,96	**0,97**
chess (krkp)	0,97	**0,99**	**0,99**	**0,99**	**0,99**	**0,99**
credit ap.	0,87	0,85	**0,88**	0,83	0,86	0,87
heart	0,83	0,80	0,83	0,81	**0,84**	0,83
hepatitis	0,76	0,65	0,72	0,73	0,77	**0,78**
pima	0,73	0,72	0,73	0,73	0,73	**0,75**
sick	0,81	**0,92**	0,83	0,85	0,82	0,83
voting	**0,96**	**0,96**	**0,96**	0,94	**0,96**	**0,96**

6.1 Классификация электрорадиоизделий космического применения

Одно из практических внедрений метода – решение задачи классификации электрорадиоизделий (ЭРИ) по производственным партиям для комплектации радиоэлектронной аппаратуры космических аппаратов.

Основным способом повышения эффективности функционирования космических аппаратов (КА) и снижения затрат на их восполнение является увеличение сроков активного существования (САС) КА. С увеличением САС КА возрастает готовность космических систем (КС), уменьшается число КА, необходимых для восполнения и поддержания пропускной способности КС, снижается потребность в ракетах-носителях для осуществления пусков, упрощается система управления функционирующими КА.

Важность исследований в направлении увеличения сроков активного существования КА до 10 ÷ 15 лет подтверждена в документе «Межведомственный перечень приоритетных направлений развития науки, технологий и техники, критических технологий, реализуемых в ракетно-космической промышленности в

интересах создания перспективных космических средств различного целевого назначения на 2008-2012 годы. Федеральное космическое агентство, Космические войска МО РФ. Москва. 2008» и не потеряла актуальности и в настоящее время.

Эксплуатационные характеристики КА во многом обусловлены техническим уровнем входящей в их состав электронной компонентной базы (ЭКБ) и способностью конструктора обеспечить условия для длительной работоспособности в жестких условиях космического пространства.

В современных КА функционируют 100-200 тысяч штук различных ЭКБ (интегральные схемы, транзисторы, диоды, реле, конденсаторы, резисторы и т.д.), которые должны обеспечивать надежную длительную работоспособность бортовых систем. Обеспечение надежности ЭКБ в течении САС КА является сверхважной научной и практической задачей.

Целью исследования является повышение надежности и срока активного существования КА на основе развития методов прогнозирования работоспособности, обеспечения качества и надежности ЭКБ в КА длительного функционирования. Это позволит решить важную научную и хозяйственную задачу обеспечения аппаратуры военной техники, космического и другого специального назначения ЭКБ с повышенными эксплуатационными характеристиками, а также повысить эффективность её применения.

Выявление однородных партий электрорадиоизделий (ЭРИ), применяемых в узлах космической электроники, является одной из важных задач на пути повышения качества этих узлов и, как следствие, срока активного существования и надежности космической техники. Повышение качества достигается как за счет более согласованной работы радиоэлементов с идентичными характеристиками, так и за счет повышения качества и достоверности результатов разрушающих тестовых испытаний, для которых появляется возможность гарантированно отбирать элементы из каждой производственной партии.

Требования к качеству электронной компонентной базы (ЭКБ), используемой в бортовой аппаратуре, не выполняются для изделий, выпускаемых

российскими предприятиями. Кроме того, поставляемые партии ЭРИ могут быть неоднородными [224].

Тестированием ЭРИ занимается специализированное предприятие – ИТЦ НПО ПМ в Железногорске. Дефекты обнаруживаются проведением сотен неразрушающих тестов и ряда разрушающих. Распространять результаты разрушающих испытаний на всю партию можно только при условии, что эта партия однородна по технологии и сделана из одной партии сырья [224].

В работах Патраева, Орлова, Федосова [225-227] доказано, что однородная партия дает схожие результаты неразрушающих тестов. Изделия с различающимися эксплуатационными характеристиками (фактически различные партии ЭРИ) имеют различия в полученных результатах испытаний. Это дает возможность применения методов анализа данных для осуществления необходимой группировки ЭРИ.

Выявление однородных партий электрорадиоизделий (ЭРИ) является одной из важнейших задач на пути повышения надежности космической техники и увеличения срока активного существования (не менее 15 лет). Однородность партий изделий обеспечивает достоверность результатов разрушающих тестовых испытаний. Также обеспечивается более согласованная работа радиоэлементов с идентичными характеристиками.

Системы поддержки принятия решений, используемые при проведении испытаний, классификации и отборе электрорадиоизделий для комплектации радиоэлектронной аппаратуры космического применения, должны удовлетворять ряду требований, в том числе они должны обеспечивать возможность обосновывать решения и интерпретировать результат. Одной из ключевых задач, возникающих при отборе изделий микроэлектроники, является задача классификации ЭРИ по однородным партиям.

Ранее разработана система, позволяющая производить выявление однородных производственных партий в сборной партии электрорадиоизделий

космического применения (Казаковцев, Масич [228]). Система основана на использовании алгоритма с жадной эвристикой [229].

Целью исследования является создание метода выявления однородных партий в массивах поступающих от производителя электрорадиоизделий (ЭРИ) путем анализа данных, получаемых по результатам проведенных диагностических испытаний, что необходимо для проведения последующих испытаний ЭРИ (разрушающего физического анализа). Алгоритм обработки данных должен обеспечивать стабильные результаты, воспроизводимые при многократных запусках алгоритма.

Исходными данными для анализа при решении задачи являются результаты тестовых воздействий на ЭРИ по контролю вольт-амперных характеристик входных и выходных цепей микросхем. Данные представляют собой таблицу, в строках которой приведены последствия различных электрических воздействий на элементы набора однотипных ЭРИ. Предполагается, что изделия с различающимися эксплуатационными характеристиками (фактически различные партии ЭРИ) будут иметь различия в полученных результатах испытаний. Такое предположение дает возможность применения методов анализа данных для осуществления необходимой группировки ЭРИ.

Исследовалась задача повышения эффективности классификации посредством формирования информативных закономерностей и разработки процедур, позволяющих улучшить интерпретируемость классификатора. Построение подобных классификаторов может быть основано на различных методах, среди которых наиболее перспективными для данной задачи являются методы логической классификации, отличающиеся высокой интерпретируемостью результатов классификации. Интерпретируемость результатов логической классификации в условиях космического производства означает возможность разработки ужесточенных норм параметров ЭРИ.

6.2 Типы испытаний электрорадиоизделий

Комплектация критически важных электронных узлов сложных систем компонентной базой соответствующего качества является одной из важнейших составляющих задачи повышения надежности системы в целом. При этом важно, чтобы однотипные элементы схемы имели очень близкие характеристики для обеспечения их согласованной работы. Наилучшим образом такая однородность характеристик достигается в случае, если элементы изготовлены в рамках одной производственной партии из одной партии сырья [224]. При комплектации критически важных узлов используются элементы, изготовленные отдельными специальными партиями, к которым предъявляются повышенные требования качества [230, 231].

Характеристики партии и ее элементов контролируются с помощью разрушающих и не разрушающих тестов [231]. Данные, полученные с помощью не разрушающих тестов, могут быть использованы для анализа однородности партии. Для разбиения множества исследуемых электронных компонентов по предполагаемым производственным партиям используется наиболее популярный метод автоматической группировки – метод *k*-средних [232].

Зарубежные производители изготавливают изделия специальных классов качества военного (Military) и космического (Space) назначения [233-236]. Отечественные производители не выделяют изделия космического применения в специальный класс. В этой связи классификацию экземпляров ЭРИ с целью выделения потенциально ненадежных в условиях космического полета экземпляров приходится производить в специализированных тестовых центрах.

Конечная цель усовершенствования процесса комплектации ЭКБ космического применения – предотвращение отказов [237-240] [16, 40, 46, 52], которые могут возникать внезапно (при скачкообразном изменении параметров изделия) или постепенно (постепенный дрейф параметров). Следует также выделить специальный вид неисправностей – дефекты [241] – неисправности,

которые непосредственно в момент их обнаружения не приводят к отказу изделия.

Для полупроводниковых приборов [242] причинами отказов являются короткие замыкания, обрывы и изменения параметров. Нарушение технологии производства может приводить к дефектам, способным вызвать отказ любого из перечисленных типов. Но и в случае точного следования предусмотренной технологии случайные вкрапления в материалах, скрытые дефекты оборудования и множество иных причин могут приводить к дефектам, способным в будущем стать причиной отказа. Причиной замыканий может стать теплоэлектрический пробой (вначале – туннельный пробой, приводящий к увеличению тока и сильному разогреву, ведущему к необратимому изменению структуры полупроводника). Вероятность пробоя повышают такие дефекты, как плохое (неполное) соединение кристалла с корпусом, неравномерное распределение тока в структуре, попадание внутрь корпуса ППП токопроводящих частиц, провисания внутренних проводников вследствие их избыточной длины. Обрывы, кроме того, могут возникать из-за механических разрушений соединений кристалла с корпусом или сварных соединений внутренних проводников с другими элементами конструкций при воздействии вибрации, ударов, больших линейных ускорений, химических и электрохимические разрушения соединений и металлических пленок, роста интерметаллической базы в местах соединения разных металлов и расслоения структуры, напряжений в проводниках прибора, залитых пластмассой (температурные коэффициенты линейного расширения металла проводника и пластмассы различны), дефектов металлизации дорожек и контактных площадок. Обрывы зачастую проявляются при достижении некоторой критической температуры.

К нестабильности характеристик прибора часто приводят поверхностные заряды (генерация и перемещение электрических зарядов на поверхности кристалла), что ведет к изменению концентрации подвижных носителей рекомбинации. Причинами таких зарядов могут быть появление ионов на поверхности кристалла (зависит от чистоты окружающей среды и способа

обработки кристаллов), появление заряда, образованного ионами натрия внутри пленки окисла. Для защиты от таких ионов, перемещающихся при температурах 100-200 С в притягивающем электрическом поле к границе кремний-окисел создают тонкую пленку фосфорно-силикатного стекла для захвата этих элементов. Кроме того, возможно появление заряда, образованного избыточными атомами кремния в окисле около границы с кристаллом, которые находятся в избытке при механизме образования окисла кремния, отчего образуется положительный заряд, созданный оставшимися атомами кремния. При очень высокой температуре и наличии сильного поля этот заряд перемещается к внешней поверхности окисла пленки.

При обрыве кристаллической решетки на ее границе появляются атомы, в которых электроны занимают энергетические состояния, лежащие внутри запрещенной зоны энергий для данного материала, из-за чего они не могут проникать вглубь кристалла и остаются только на поверхности. Чем дальше от полупроводника локализован поверхностный энергетический уровень, тем больше время его заполнения или освобождения. Этим может быть вызван низкочастотный шум и нестабильность основных параметров прибора.

Появление каналов проводимости вдоль поверхности кристалла приводит к изменению коэффициента усиления в транзисторах.

Комбинация высокого напряжения с высокой температурой дает наиболее существенные изменения параметров в процессе испытаний. При таких напряжениях в электрическом поле происходит разделение и группировка положительных и отрицательных ионов на поверхности, дрейф этих ионов по направлению к электродам, имеющим соответствующий знак заряда.

Скрытые дефекты снижают качество продукции с точки зрения риска отказов, поэтому многие исследователи отводят решающую роль неразрушающим испытаниям – дефектологии, науке о принципах, методах и средствах обнаружения дефектов.

Первую группу неразрушающих испытаний составляют методы интегральной диагностики, основанные на измерении шумовых характеристик, в

том числе электрических и акустических шумов. Вторую группу составляют методы локальной диагностики. К общим испытания относятся визуальный контроль, испытание давлением, акустическая и магнитная дефектоскопия, радиография и метод вихревых токов. К специфическим методам испытаний относятся рентгеновские, голографические, тепловые, оптические и электрические методы.

Наличие скрытого дефекта, так или иначе, отражается на результате одного или нескольких испытаний. В то же время, влияние дефекта на результат может быть различным и зависит как от типа дефекта, так и от типа изделия. Кроме того, параметры изделий, изготовленных в несколько отличающихся условиях и из разных партий сырья, могут довольно существенно отличаться. Выделение групп изделий, в которых с некоторой повышенной вероятностью содержатся изделия, имеющие дефекты, также выделение групп изделий, изготовленных одновременно из единой партии сырья, является основными задачами настоящего исследования, для чего применяются методы автоматической группировки [224].

Комплектация критически важных электронных узлов сложных систем компонентной базой соответствующего качества является одной из важнейших составляющих задачи повышения надежности системы в целом. При этом важно, чтобы однотипные элементы схемы имели очень близкие характеристики для обеспечения их согласованной работы. Наилучшим образом такая однородность характеристик достигается в случае, если элементы изготовлены в рамках одной производственной партии из одной партии сырья [224]. При комплектации критически важных узлов используются элементы, изготовленные отдельными специальными партиями, к которым предъявляются повышенные требования качества [230].

Характеристики партии и ее элементов контролируются с помощью разрушающих и не разрушающих тестов [231]. Данные, полученные с помощью не разрушающих тестов, могут быть использованы для анализа однородности партии. Для разбиения множества исследуемых электронных компонентов по

предполагаемым производственным партиям используются методы автоматической группировки [232].

Результатом диагностических испытаний, проводимых испытательным центром, является набор параметров каждого экземпляра поступившей с завода-изготовителя партии. Результат каждого из видов испытаний является числовой характеристикой (чаще всего – напряжение или ток в какой-либо цепи в тех или иных условиях). Примеры таких данных приведены в приложении А. Сделать вывод об однородности либо разнородности партии изделий по условиям производства следует на основе анализа этих характеристик. Хотя к диапазону каждой из этих характеристик применяются жесткие требования [243, 244], незначительные на первый взгляд колебания сразу нескольких характеристик позволяют сделать вывод о том, что части партии произведены в разных условиях.

6.3 Исходные данные для классификации ЭРИ

Исходными данными для анализа при решении задачи являются результаты тестовых воздействий на ЭРИ по контролю вольт-амперных характеристик входных и выходных цепей микросхем. Данные представляют собой таблицу, в строках которой приведены последствия различных электрических воздействий на элементы набора однотипных ЭРИ [224].

В ходе экспериментов исследовались следующие наборы исходных данных.

1. Микросхемы 1526ЛЕ5

619 изделий, 120 измерений, указано, что выборка является смесью 3 партий.

2. Полевые транзисторы 2П771А

182 изделия, 18 измерений, указано, что выборка является смесью 2 партий.

3. Арсенид-галиевые диоды 3Д713В

258 изделий, 12 измерений, указано, что выборка состоит из изделий одной партии.

4. Микросхемы OP297GSZ

700 изделий, 18 измерений, число партий не известно.

5. Оптроны транзисторные одноканальные 3ОТ122А

278 изделий, 14 измерений, указано, что выборка является смесью 4 партий.

6. Микросхемы на основе БМК Н5503ХМ5-171

166 изделий, 188 измерений, указано, что выборка состоит из изделий одной партии.

7. Микросхема Н5503ХМ1-289

229 измерений, выборка содержится в 5 файлах, в каждом из которых примерно по 80 изделий.

8. Микросхема 1526ТЛ1

157 измерений, выборка содержится в 13 файлах, в каждом из которых примерно по 100 изделий.

9. Диод 2Д522Б

10 измерений, выборка содержится в 10 файлах, в каждом из которых по 150-500 изделий.

10. Усилители 140УД17АВК и 140УД25АВК, два набора, состоящие из двух партий каждый.

В качестве примера в таблице 6.2 приведен состав исходных данных для микросхемы 1526ЛЕ5.

Таблица 6.2 - Исходные данные для проведения диагностического контроля ИМС 1526ЛЕ5

Контролируемые параметры, единицы измерения	Тесты	Режим измерения
1 Входной ток низкого уровня, IIL(УР), мкА	11-18	Ucc=11 В UIH=11 В
2 Входной ток высокого уровня, IIH(УР), мкА	19-26	Ucc=11 В UIL=0 В
3 Максимальное выходное напряжение низкого уровня, UOLmax(УР), В	27-30	Ucc=5 В UIH=3.5 В
4 Минимальное выходное напряжение высокого уровня, UOHmin(УР), В	31-38	Ucc=5 В UIL=1.5 В
5 Выходной ток низкого уровня, IOL(УР), мА	39-42	Ucc=5 В Uo=0.4 В
6 Выходной ток высокого уровня, IOH(УР), мА	43-46	Ucc=5 В Uo=2.5 В
7 Ток потребления, Icc(УР), мкА	47-55	Ucc=11 В UIH=11 В
8 Прямая ВАХ верхнего диода, U, В	56,59,62,65, 68,71,74,77	II=0.1 мкА
	57,60,63,66, 69,72,75,78	II=100 мкА
	58,61,64,67, 70,73,76,79	II=1 мА

Контролируемые параметры, единицы измерения	Тесты	Режим измерения
9 Прямая ВАХ нижнего диода, U, В	80,83,86,89,92,95,98,101	II=0.1 мкА
	81,84,87,90,93,96,99,102	II=100 мкА
	82,85,88,91,94,97,100,103	II=1 мА
10 Прямая ВАХ диода цепи питания, U, В	104	II=0.1 мкА
	105	II=100 мкА
	106	II=10 мА
11 Выходная ВАХ нижнего транзистора, U, В	107,110,113,116	Io=0.1 мА
	108,111,114,117	Io=0.3 мА
	109,112,115,118	Io=1 мА
12 Выходная ВАХ верхнего транзистора, U, В	119,122,125,128	Io=0.1 мА
	120,123,126,129	Io=0.3 мА

Контролируемые параметры, единицы измерения	Тесты	Режим измерения
	121,124,127 ,130	Io=1 мА

Результаты измерении для токов представлены в миллиамперах, напряжения- в вольтах.

6.4 Преобразование исходных данных для построения логических закономерностей

В этом разделе рассматривается задача преобразования набора данных, представленных в произвольном числовом формате, в соответствующий набор в булевом формате в контексте задачи классификации. Это необходимо для подготовки входных данных для дальнейшего выявления в них логических закономерностей, что предполагает, что все примеры представлены векторами булевых значений.

Рассмотрим описанный подход для определения оптимальных пороговых значений для данных тестовых испытаний ЭРИ.

Пример 1. Усилители 140УД17АВК.

Набор усилителей состоит из 50 экземпляров. Набор состоит из двух производственных партий (24 изделия в первой и 26 во второй). Результаты тестов описываются 46 величинами (признаками): 23 значения – параметры после электротермотренировки, 23 значения – дрейф параметров.

Максимальный набор (наиболшее число порогов, полученных в соответствие с описанным подходом) состоит из 759 порогов. Для каждого численного признака используется от 1 до 29 порогов (рисунок 6.1).

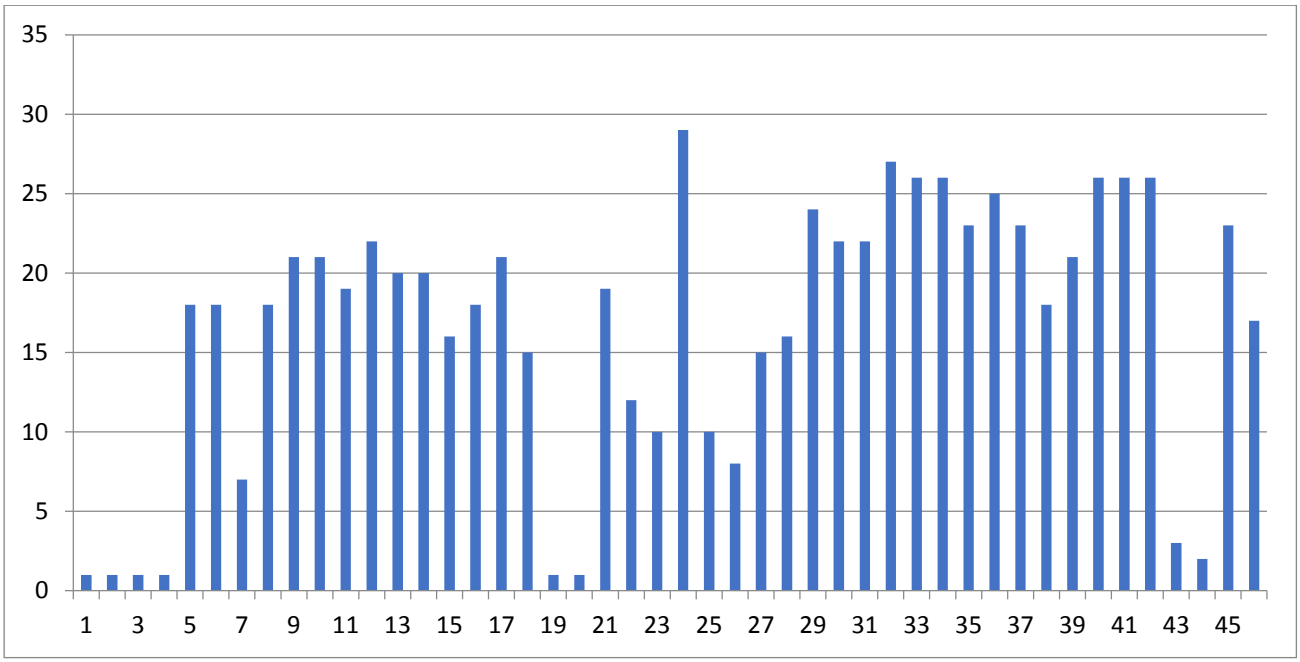


Рисунок 6.1 – Число порогов максимального набора, ось абсцисс – номер количественного признака, ось ординат – число порогов

Задача определения минимального числа порогов решалась как задача целочисленного линейного программирования (задача о покрытии). В данном случае размерность задачи составляет: 759 переменных и 624 ограничения (по числу всевозможных пар изделий из разных групп).

В данном наборе имеются признаки, которые даже использованные по отдельности однозначно разделяют выборку на группы – это признаки с номерами 1, 2, 3, 4, 19, 20, имеющие по одному порогу. (Заметим, однако, что наличие у некоторого признака лишь одного порога не указывает однозначно на высокую разделяющую способность этого признака).

Таким образом, результатом решения задачи 1-согласованности, то есть определения минимального числа порогов, достаточных, чтобы все изделия разных групп различались хотя бы по одному признаку, является один порог (можно использовать любой из порогов выше названных 6 признаков).

При увеличении числа m, то есть при решении задачи m-согласованности (определении минимального числа порогов, достаточных, чтобы все изделия

разных групп различались хотя бы по m порогам) при m>1, растет также число необходимых для использования порогов. Результаты решения задачи m-согласованности для m от 1 до 10 приведены на рисунке 6.2.

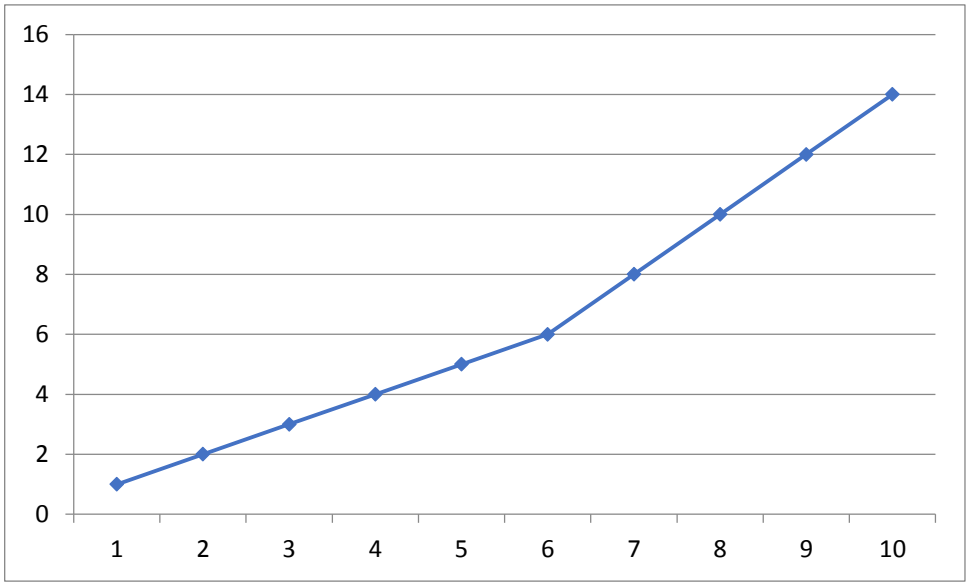


Рисунок 6.2 – Минимальное число порогов для m=1,…,10.

Распределение порогов оптимального решения по признакам при m=10 приведено на рисунке 6.3.

Как видно, пороги лишь одного признака из значений дрейфов (тест 25) учитываются для разделения выборки усилителей.

Указанные выше 6 признаков с номерами 1, 2, 3, 4, 19, 20 даже по отдельности отлично разделяют выборку на группы. Но при их отсутствии для разделения выборки потребовалось бы уже использовать не менее двух порогов, например: x7=4,965E-8 и x9=1,57E-5 (рисунок 6.4).

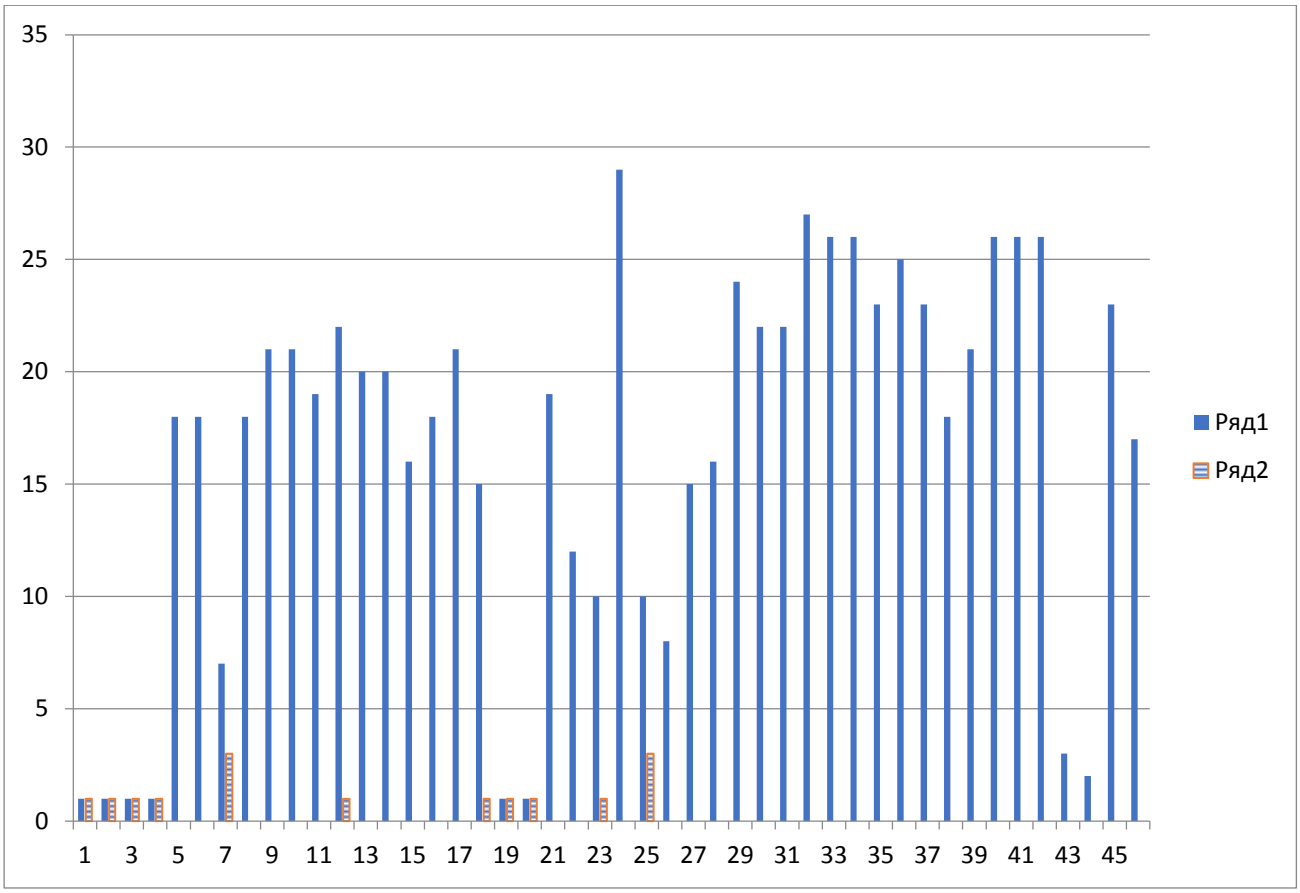


Рисунок 6.3 – Распределение порогов оптимального решения по признакам при m=10.

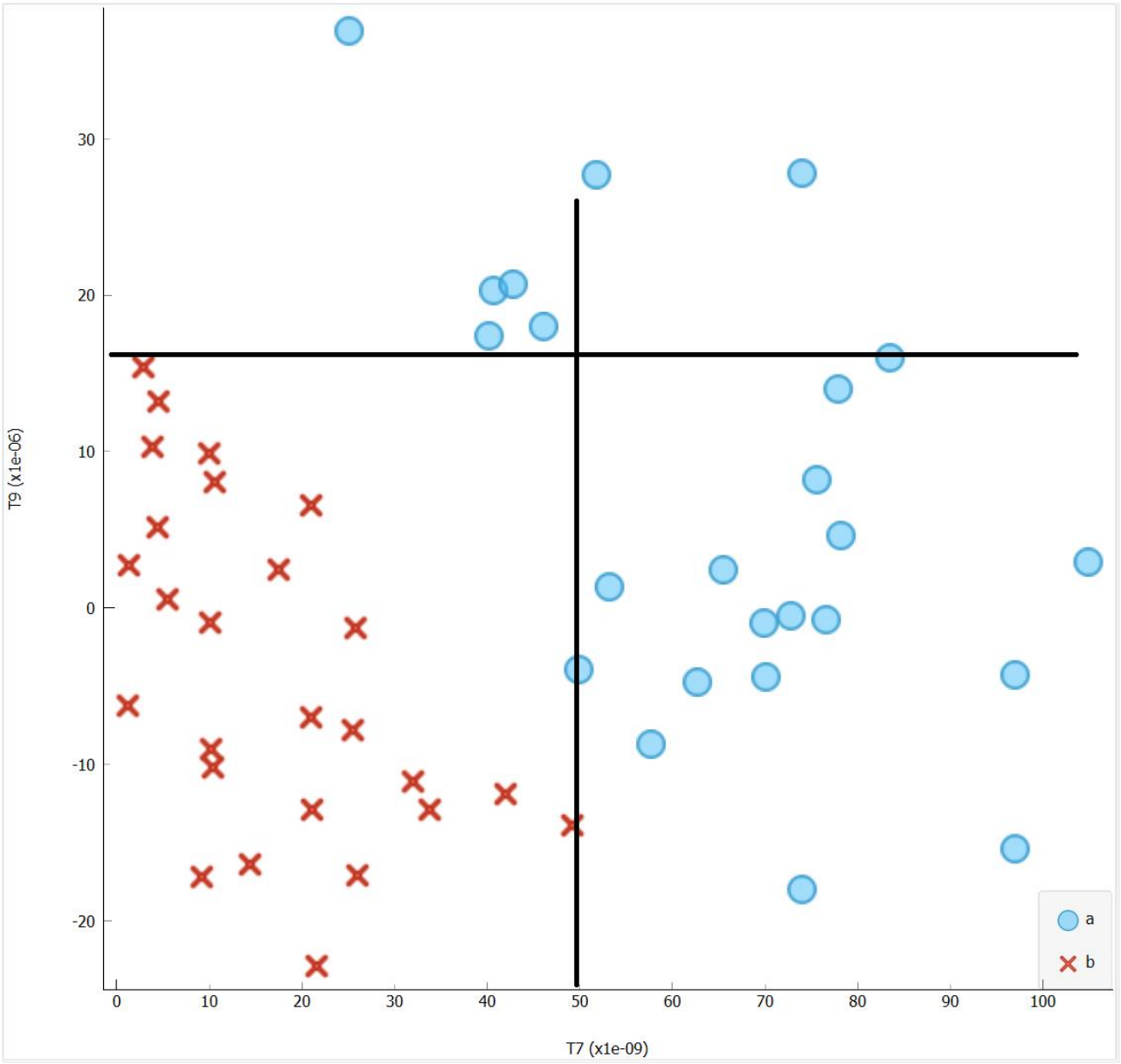


Рисунок 6.4 – Разделение выборки двумя порогами

Пример 2. Усилители 140УД25АВК.

Набор усилителей состоит из 56 экземпляров. Набор состоит из двух производственных партий (30 изделия в первой и 26 во второй). Результаты тестов описываются 42 величинами (признаками): 21 значение – параметры после электротермотренировки, 21 значение – дрейф параметров.

Максимальный набор (наиболшее число порогов, полученных в соответствие с описанным подходом) состоит из 775 порогов. Для каждого численного признака используется от 1 до 34 порогов (рисунок 6.5).

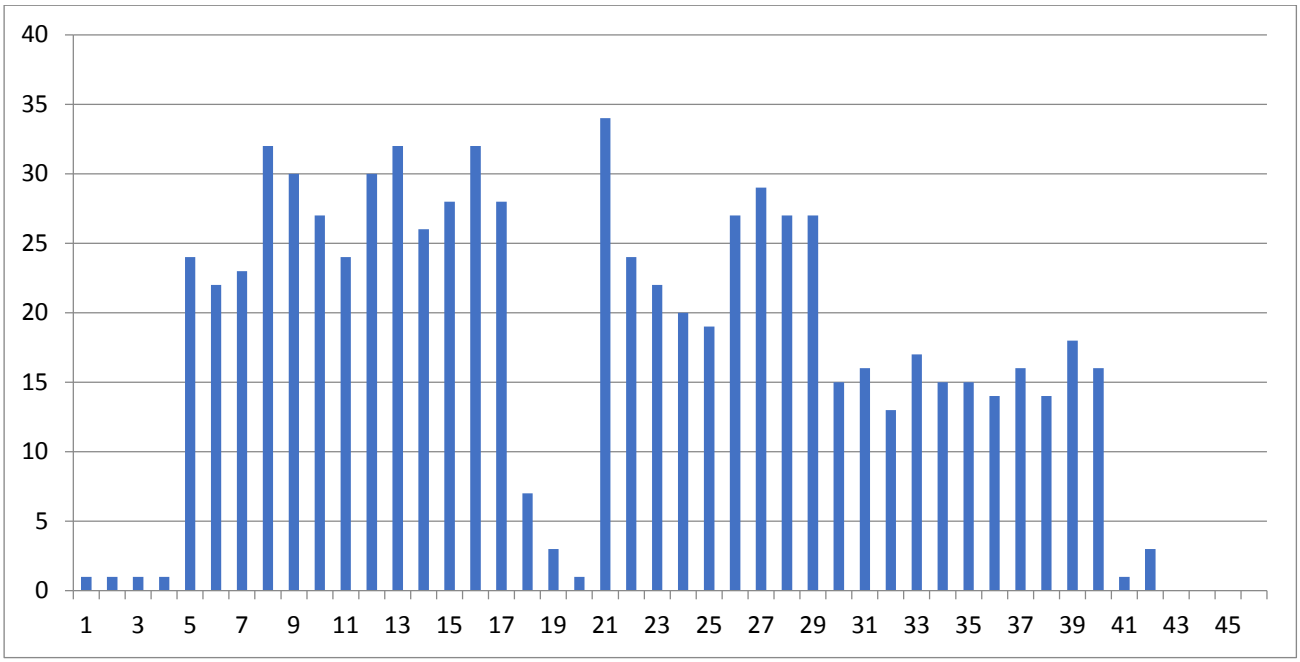


Рисунок 6.5 – Число порогов максимального набора, ось абсцисс – номер количественного признака, ось ординат – число порогов

В данном случае размерность задачи составляет: 775 переменных и 780 ограничений.

Результаты решения задачи m-согласованности для m от 1 до 10 приведены на рисунке 6.6.

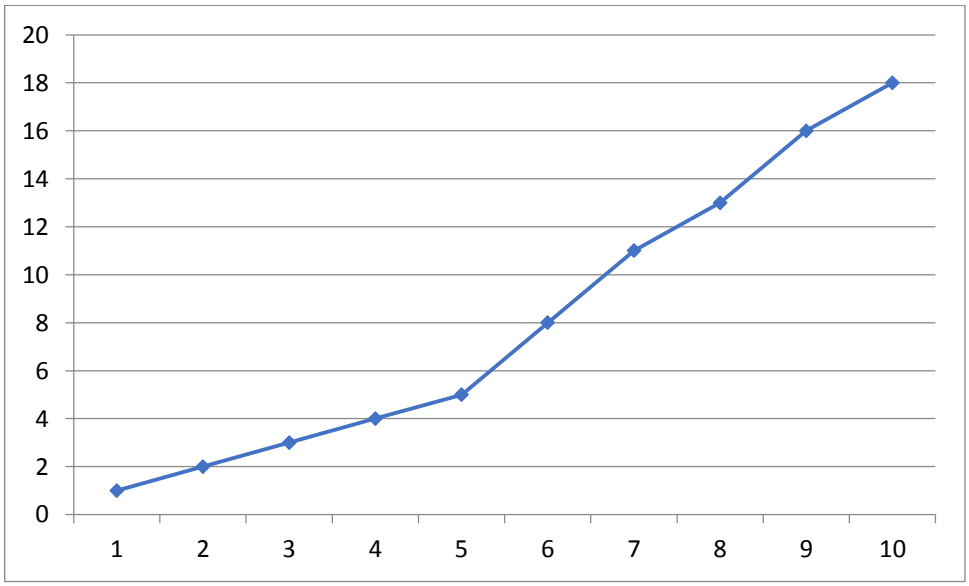


Рисунок 6.6 – Минимальное число порогов для m=1,…,10.

Распределение порогов оптимального решения по признакам при m=10 приведено на рисунке 6.7.

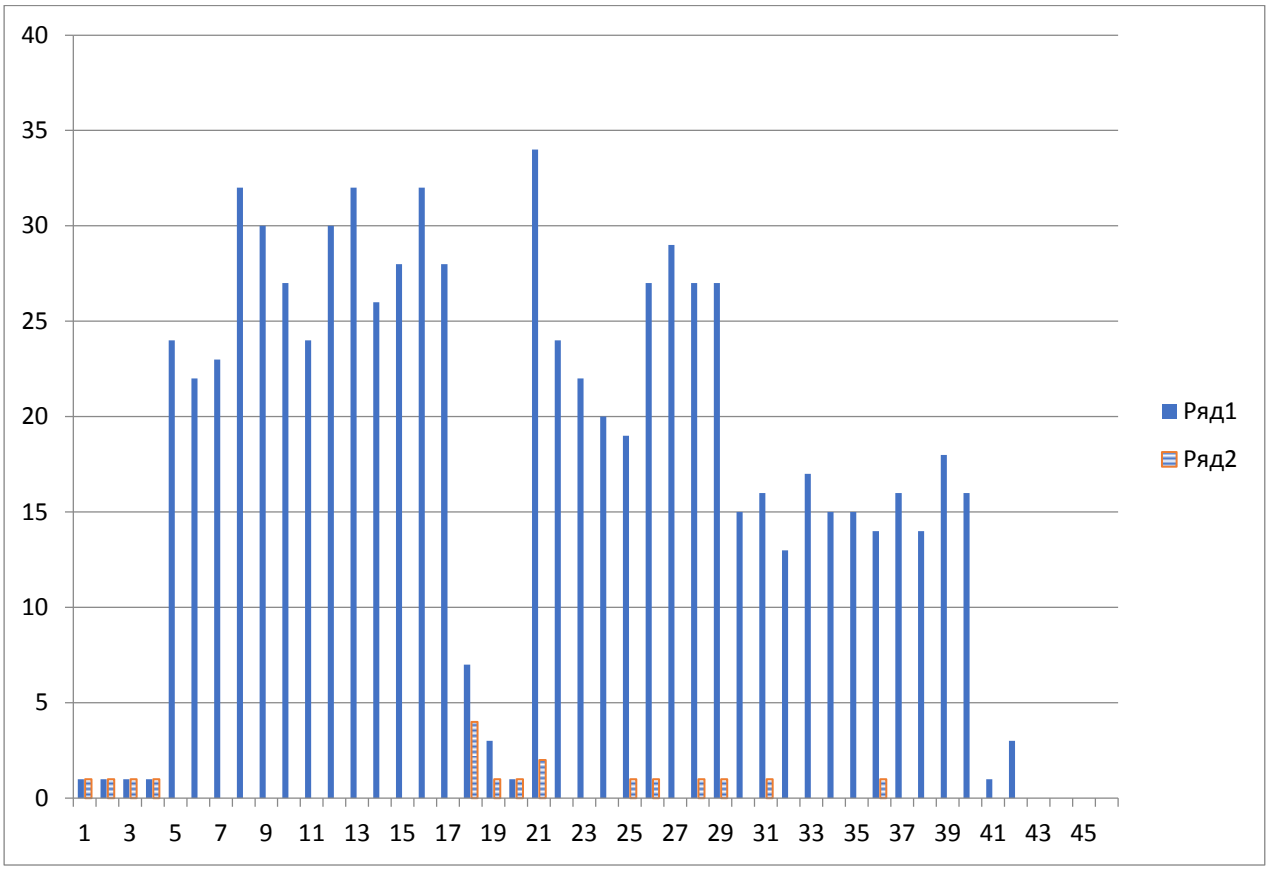


Рисунок 6.7 – Распределение порогов оптимального решения по признакам при m=10.

Следует отметить, что в этом случае треть порогов оптимального решения являются порогами из признаков, описывающих дреф параметров (с номерами больше 21), таким образом, подтверждается высокая полезность использования значений дрейфа параметров для классификации ЭРИ.

Так, для разделения изделий на группы лишь ТОЛЬКО с использованием значений дрейфа параметров достаточно 4 порогов:

x26=0,0425;

x27=-2E-8;

x37=7,05E-6;

x40=107500.

Частичное разделение с помощью двух из этих порогов (x27 и x37) приведено на рисунке 6.8.

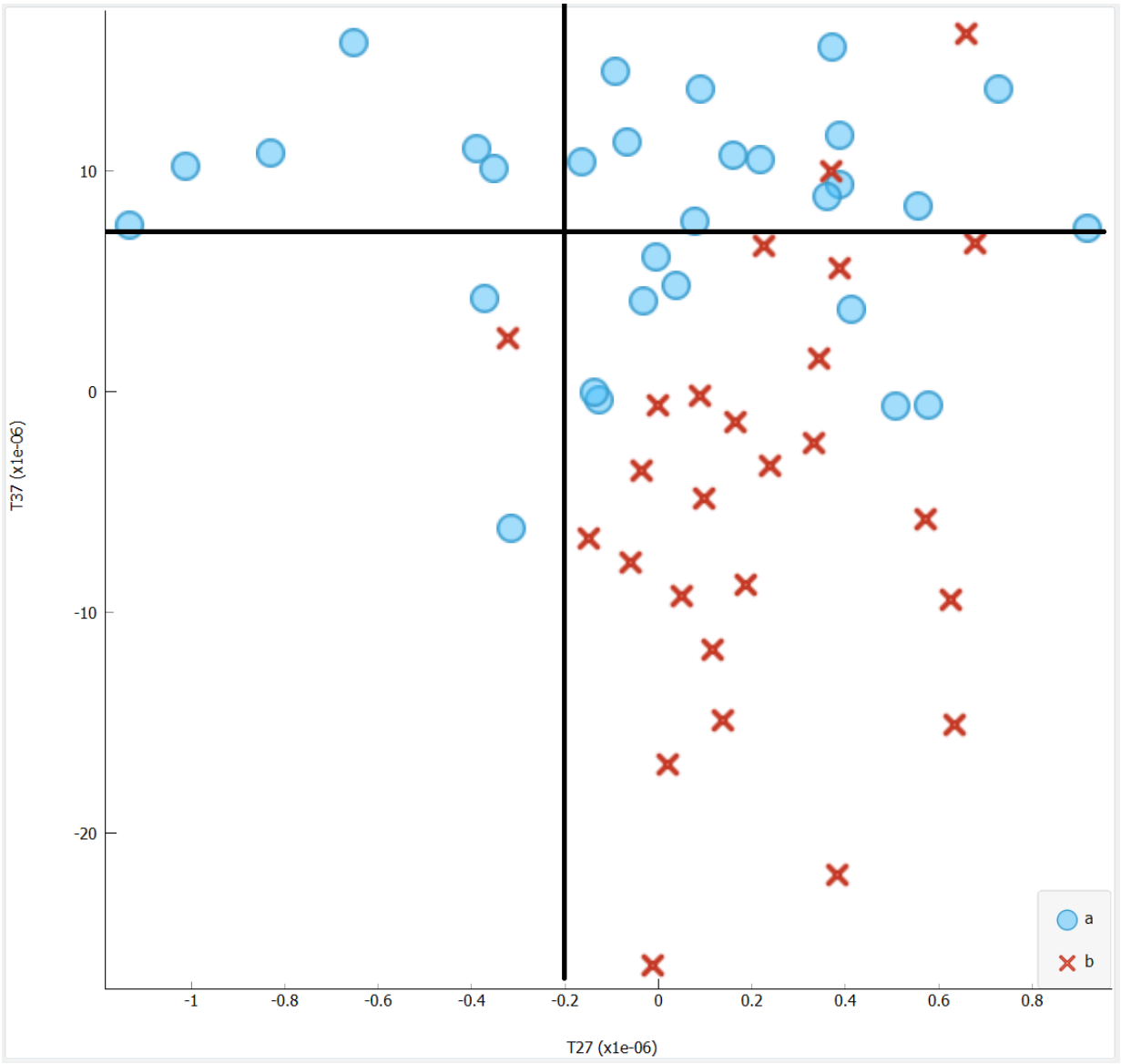


Рисунок 6.8 – Разделение выборки двумя порогами (по дрейфам параметров)

Пример 3. Микросхемы 1526ЛЕ5.

Набор ЭРИ состоит из 619 экземпляров. Набор состоит из трех производственных партий (196 изделий в первой, 205 во второй и 218 в третей). Результаты тестов описываются 120 величинами (признаками), пронумерованными от T11 до T130.

Максимальный набор (наиболшее число порогов, полученных в соответствие с описанным подходом) состоит из 2465 порогов. Для каждого численного признака используется от 1 до 34 порогов (рисунок 1.6).

В данном случае размерность задачи составляет: 2465 переменных и 127598 ограничений (по числу всевозможных пар изделий из разных групп).

Результаты решения задачи m-согласованности для *m* от 1 до 5 приведены на рисунке 6.9.

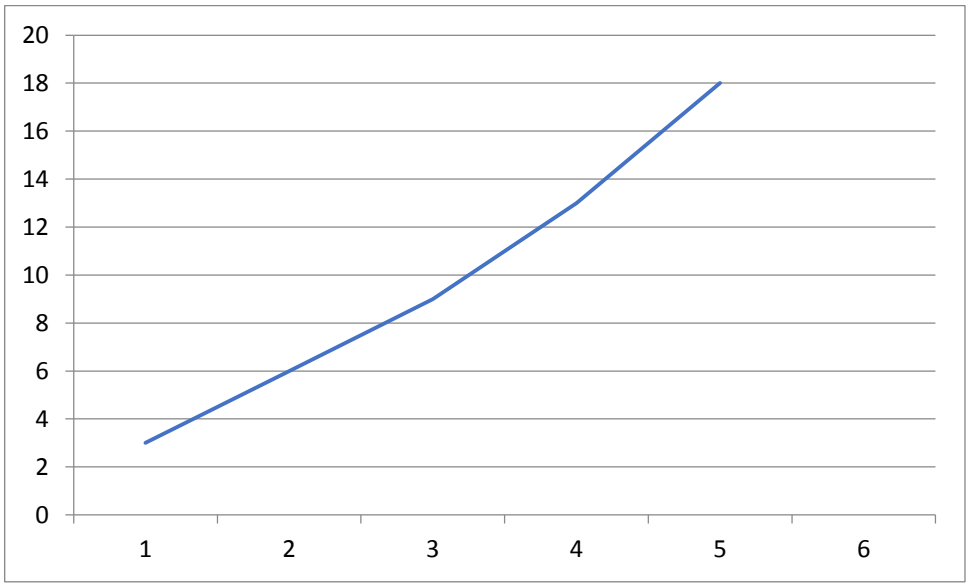


Рисунок 6.9 – Минимальное число порогов для m=1,…,5.

Для простого разделения изделий (m=1) необходимо использовать 3 порога:

T20=5E-7;

T34=4,995;

T127=4,755.

Разделение изделий с помощью этих порогов изображено на рисунках 6.10-6.12.

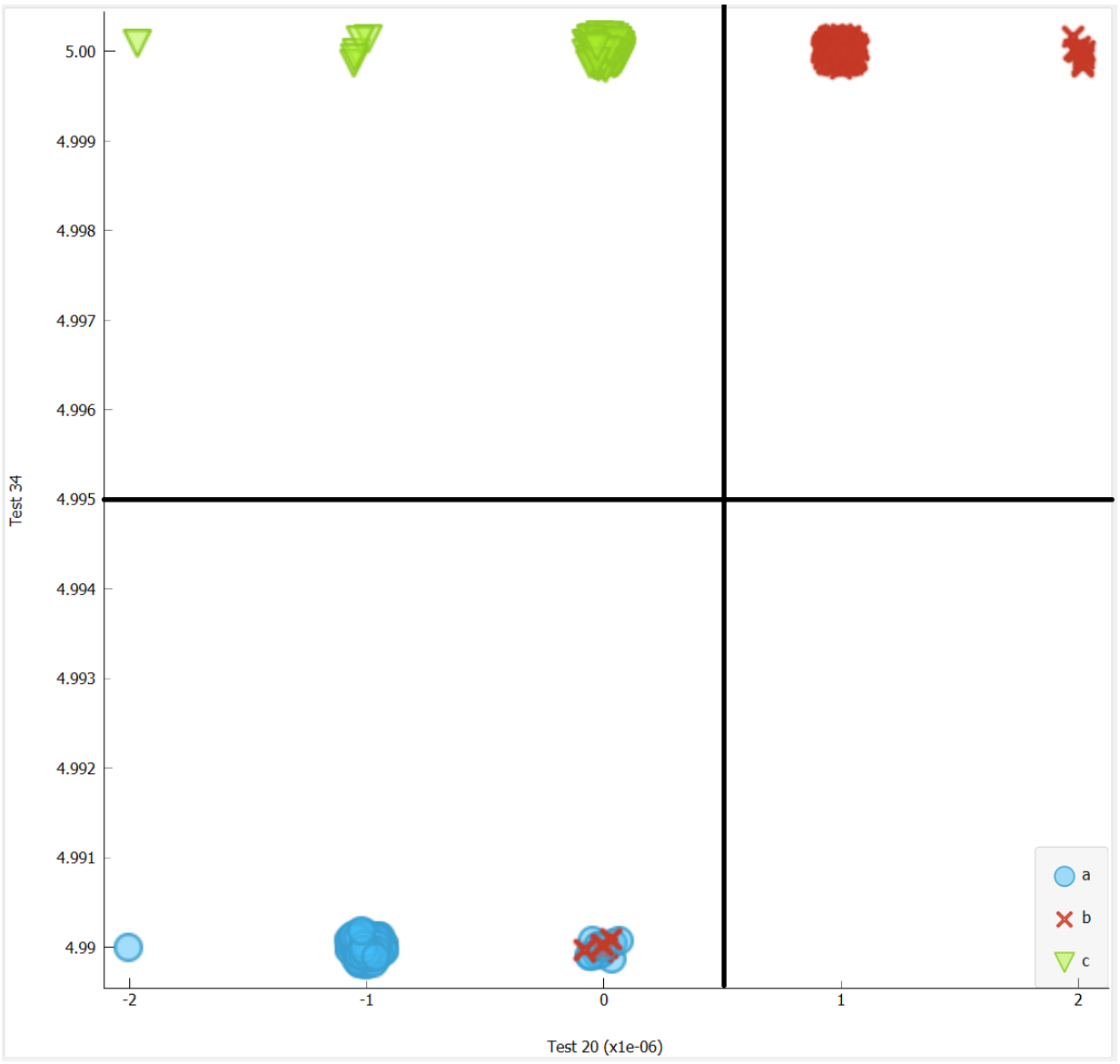


Рисунок 6.10 – Разделение изделий по признакам Т20 и Т34.

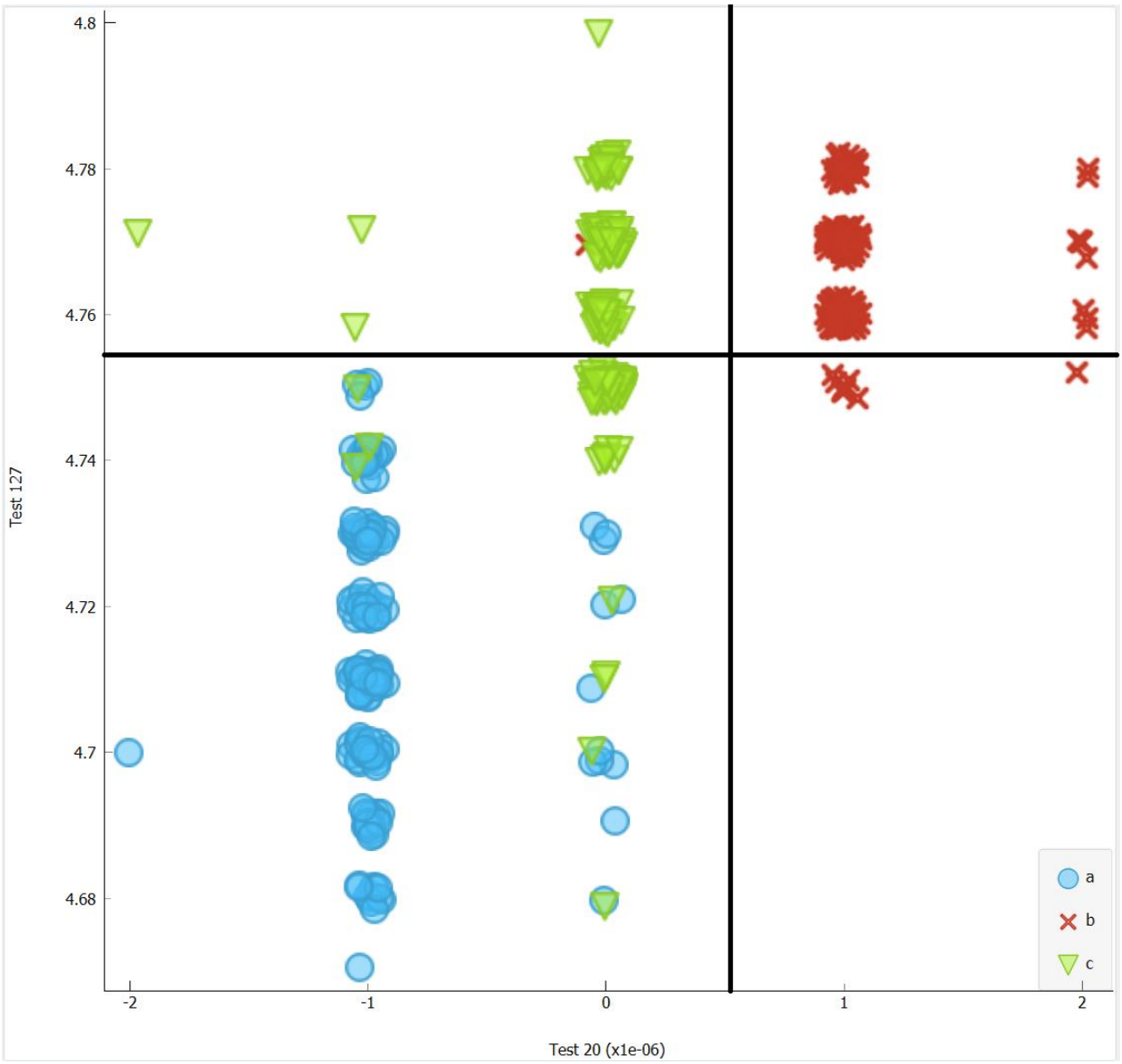


Рисунок 6.11 – Разделение изделий по признакам T20 и T127.

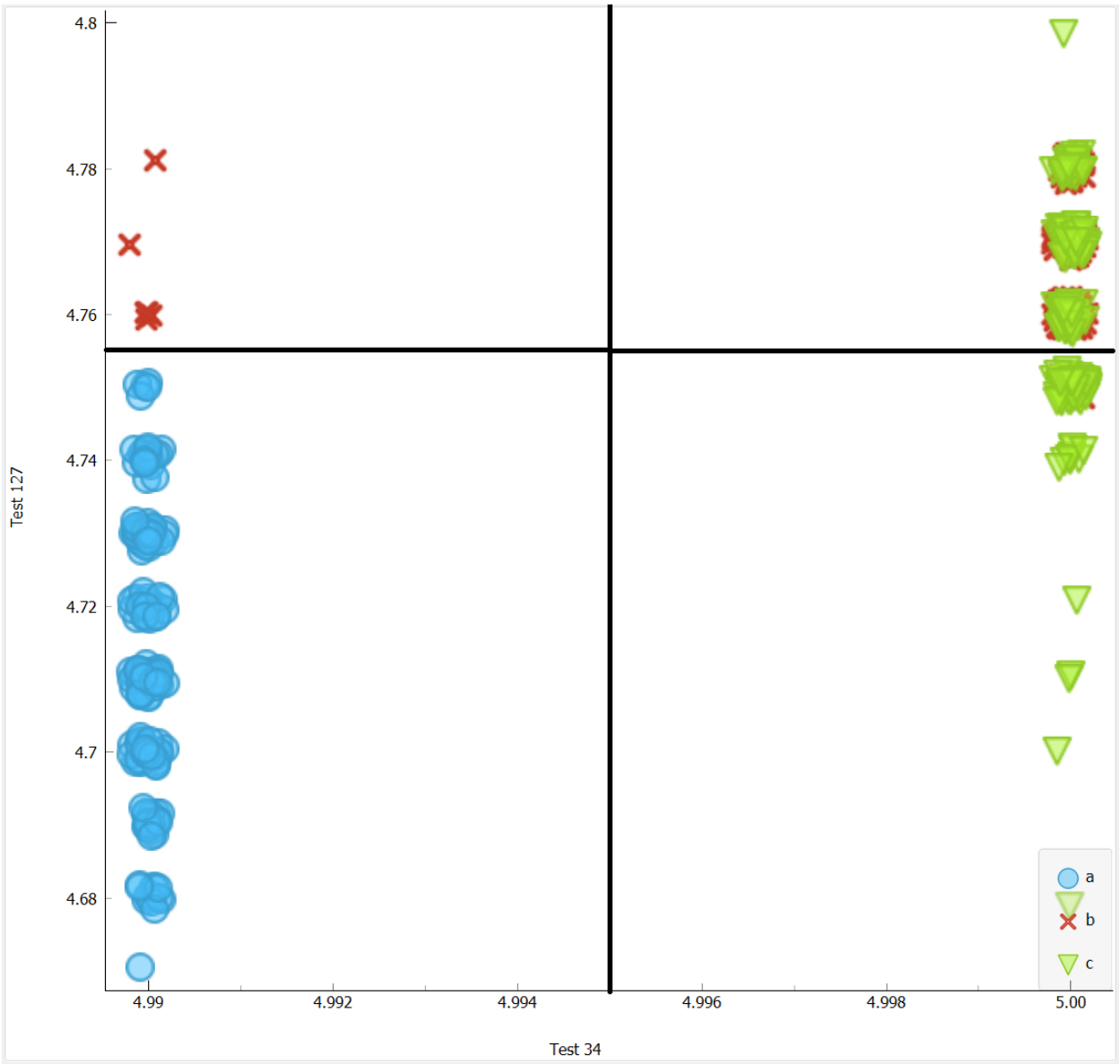


Рисунок 6.12 – Разделение изделий по признакам Т34 и Т127.

Для более надежного разделения изделий по классам необходимо использовать большее число порогов. Для m=5 минимальное число порогов равно 18, для m=10 минимальное число порогов равно 51.

6.5 Выявление информативных закономерностей (логических правил) в данных отбраковочных испытаний

Данный раздел описывает результаты исследования задачи классификации ЭРИ по партиям с помощью алгоритмов классификации, основанных на правилах.

Ранее разработана система, позволяющая производить выявление однородных производственных партий в сборной партии электрорадиоизделий космического применения [228]. Система основана на использовании алгоритма с жадной эвристикой. Работа системы не требует дополнительных испытаний: данные дополнительных отбраковочных испытаний и дополнительного неразрушающего контроля достаточны для выявления однородных производственных партий в сборной партии.

Исходными данными для анализа при решении задачи являются результаты тестовых воздействий на ЭРИ по контролю вольт-амперных характеристик входных и выходных цепей микросхем. Данные представляют собой таблицу, в строках которой приведены последствия различных электрических воздействий на элементы набора однотипных ЭРИ. Предполагается, что изделия с различающимися эксплуатационными характеристиками (фактически различные партии ЭРИ) будут иметь различия в полученных результатах испытаний. Такое предположение дает возможность применения методов анализа данных для осуществления необходимой группировки ЭРИ.

Например, результаты разбиения сборной партии микросхемы 1526ЛЕ5 на предполагаемые производственные партии для k от 1 до 4 показаны на рисунке 6.13. Результаты разбиения показаны в условном двумерном пространстве (результат процедуры MDS). Визуально для микросхемы 1526ЛЕ5 можно различить 3 кластера, что соответствует фактически присутствовавшим в сборной партии (всего 619 единиц) экземплярам трех производственных партий.

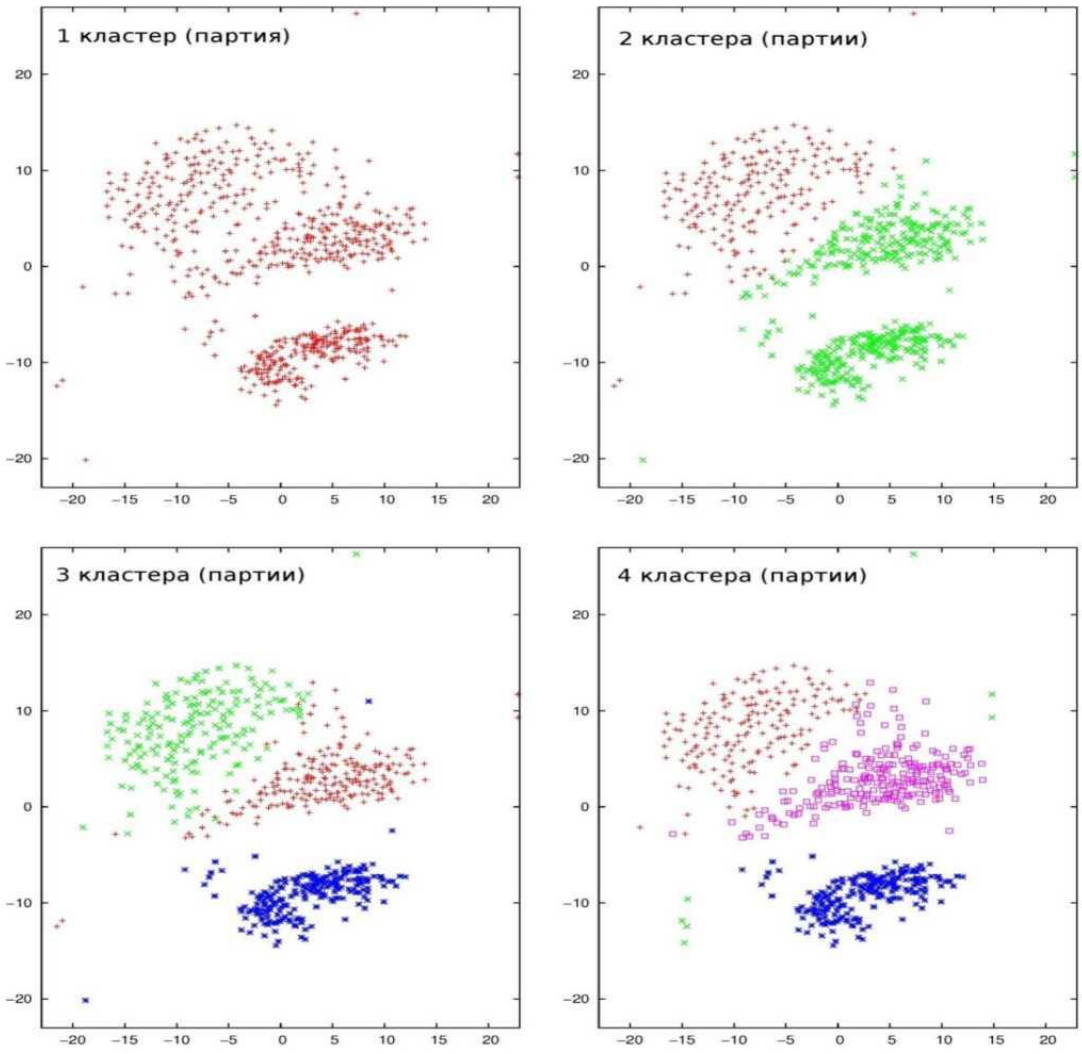


Рисунок 6.13 - Разбиение сборной партии микросхемы 1526ЛЕ5 на предполагаемые производственные партии [224]

Благодаря применению критерия силуэта в совокупности с особым способом нормировки данных, основанном на границах дрейфа, система позволяет определять число производственных партий в сборной партии [228].

На рисунке 6.14 приведены графики зависимости целевой функции и критерия силуэта от числа возможных групп.

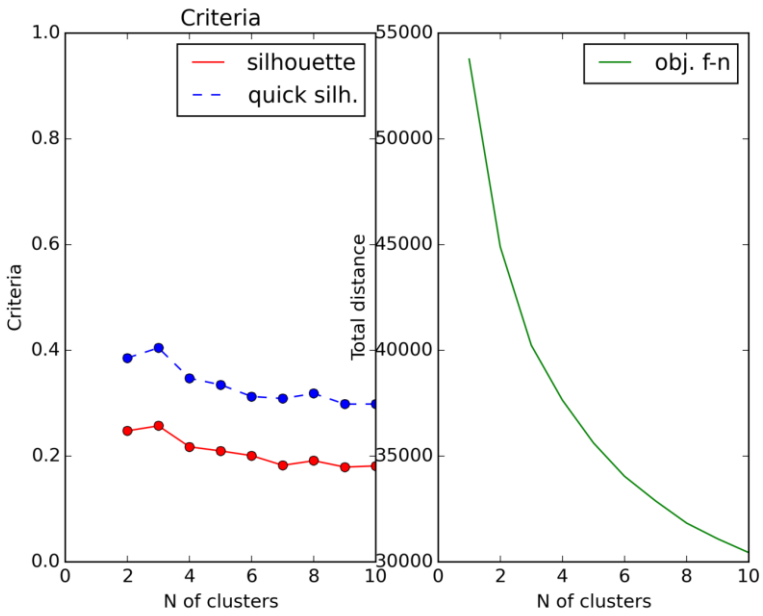


Рисунок 6.14 - График зависимости целевой функции и критерия силуэта (стандартный и ускоренный) от числа групп

Здесь исследовалась задача повышения эффективности классификации посредством формирования информативных закономерностей, базирующихся на различных принципах построения, и разработки процедур, позволяющих улучшить интерпретируемость классификатора, основанного на небольшом числе правил в нем.

Построение подобных классификаторов может быть основано на различных методах, среди которых наиболее перспективными для данной задачи являются методы логической классификации [3, 4], отличающиеся высокой интерпретируемостью результатов классификации. Интерпретируемость результатов логической классификации в условиях космического производства означает возможность разработки ужесточенных норм параметров ЭРИ.

Наиболее известные и применяемые алгоритмы построения правил (списки правил или решающие списки) являются алгоритмами типа «отделяй и властвуй». В основе лежит принцип исключения уже покрытых (каким-либо из сформированных к данному моменту правил) наблюдений из дальнейшего рассмотрения (при построении других правил). Такие алгоритмы быстро

работают, и в результате получается небольшой упорядоченный список правил, который, к тому же, легко интерпретируется. Преимущества и недостатки этих алгоритмов описаны во второй главе.

Такие алгоритмы имеют значимые для рассматриваемой задачи изъяны. Во-первых, получаемые правила (за исключением самого первого) не являются оптимальными по какому-либо критерию, так как основываются лишь на (оставшейся) части обучающей выборки. Во-вторых, может быть мало информации для принятия решения о классификации, так как решение принимается только на основе выполнения лишь одного правила (или невыполнения всех), а не на основе «голосования» правил.

В данной работе рассматриваются возможности применения более совершенного метода выявления в данных закономерностей и их использования для принятия решений о принадлежности к классам надежности, описание которого излагается в предыдущих главах.

Метод оптимальных логических решающих правил обладает рядом преимуществ. Во-первых, все получаемые правила могут быть оптимальными по используемому критерию (простота, избирательность или доказательность, а также их возможные совмещения). Во-вторых, классификатор не просто разделяет области классов, а строит аппроксимацию областей набором правил. Для оценки силы разделения этих областей может быть использовано понятие зазора. В третьих, с помощью голосования правил можно оценить достоверность принадлежности к каждому классу. Кроме того, есть множество других преимуществ: возможность работы с разнотипными признаками, допустимость наличия пропусков в данных и т.д.

Разработка и использование алгоритмов псевдобулевой оптимизации, оптимальных на классе задач, делают возможным применение этого подхода за приемлемое время, при этом позволяя «выжать» из данных всё нужное, что в них содержится.

Таким образом, в работе решается задача оценки надежности БА по результатам дополнительных отбраковочных испытаний электронной

компонентной базы с целью дальнейшего прогнозирования показателей безотказности.

Пример 1. Микросхема 1526ЛЕ5. Выборка состоит из 619 изделий, собрана из 3 производственных партий. Для каждого изделия описано 120 тестов.

На рисунке 6.15 отображены на плоскость изделия выборки с учетом всех тестов с помощью метода многомероного шкалирования.

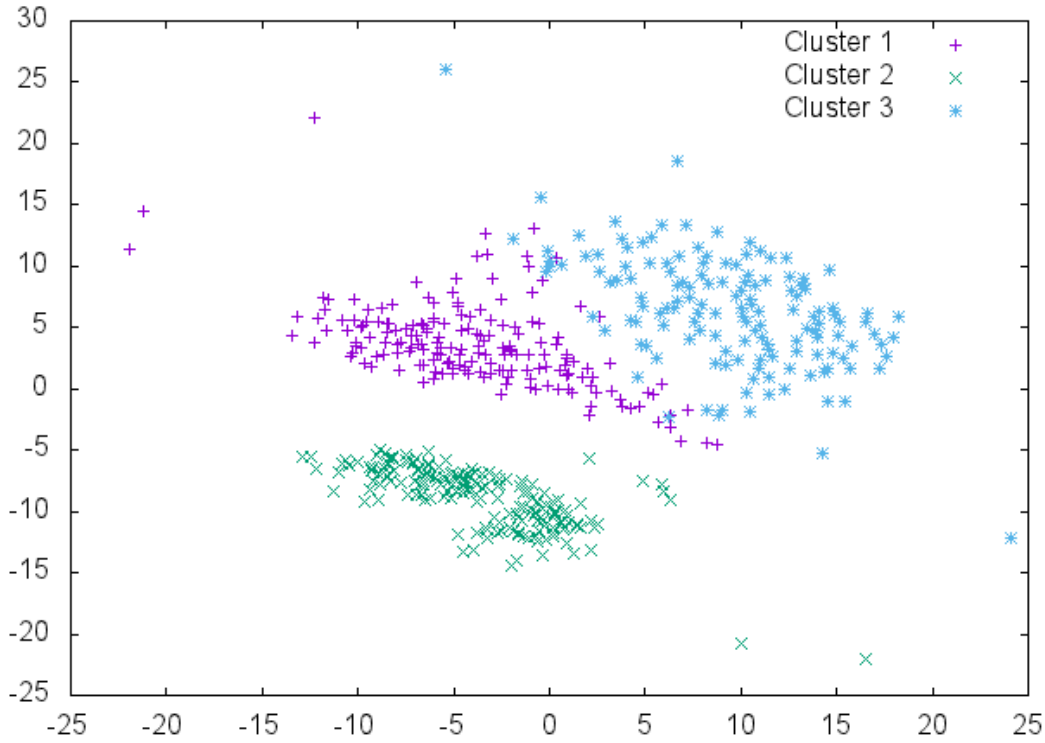


Рисунок 6.15 – Отображение выборки 1526ЛЕ5

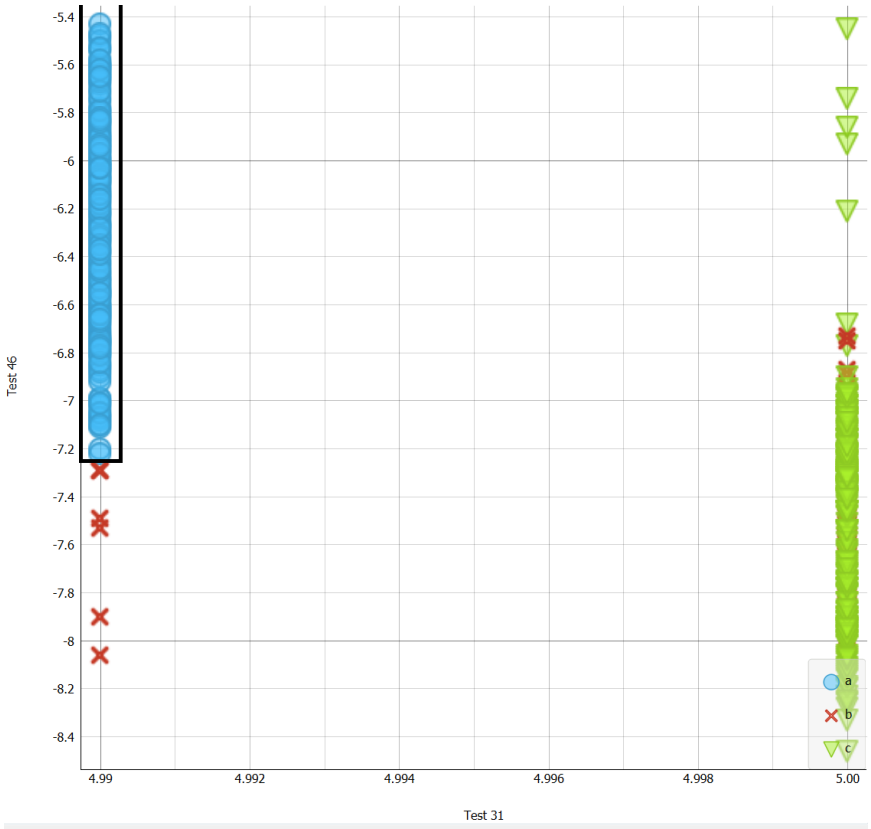
Список логических правил классификации:

(Test 31 <= 4.99) и (Test 46 >= -7.22) => class=a (196.0/0.0)

(Test 58 <= 0.928) => class=c (205.0/0.0)

=> class=b (218.0/0.0)

Полученные правила как условия для тестов наглядно изображены на рисунке 6.16.



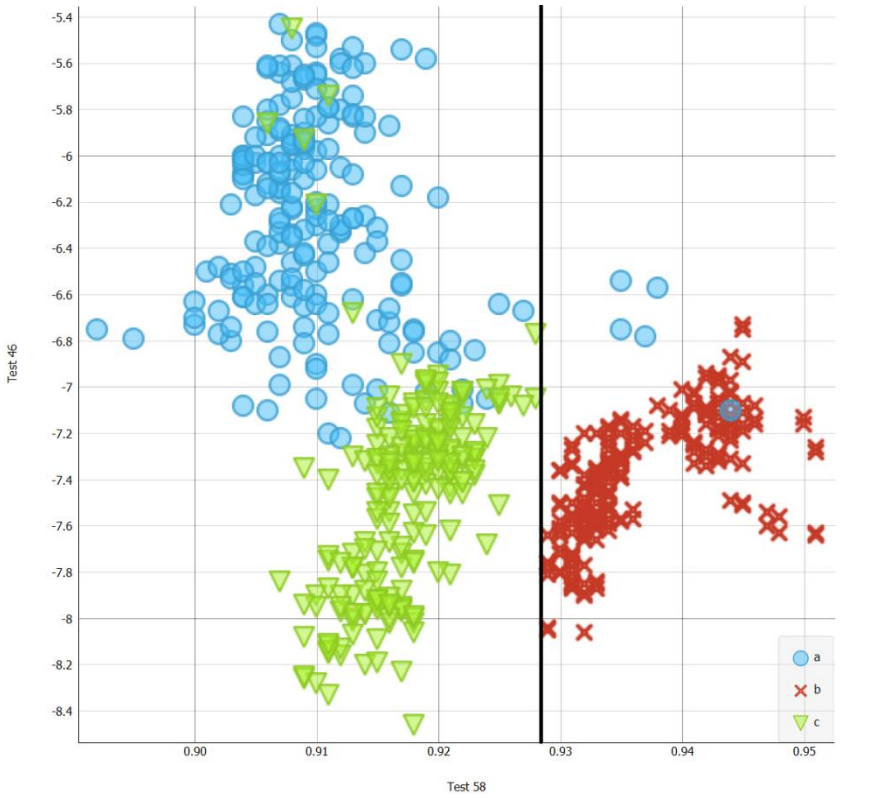


Рисунок 6.16 – Изображение логических правил в пространстве признаков

Пример 2. Полевой транзистор 2П771А. Выборка состоит из 182 изделий, собрана из 2 производственных партий. 12 тестов.

На рисунке 6.17 отображены на плоскость изделия выборки с учетом всех тестов с помощью метода многомероного шкалирования.

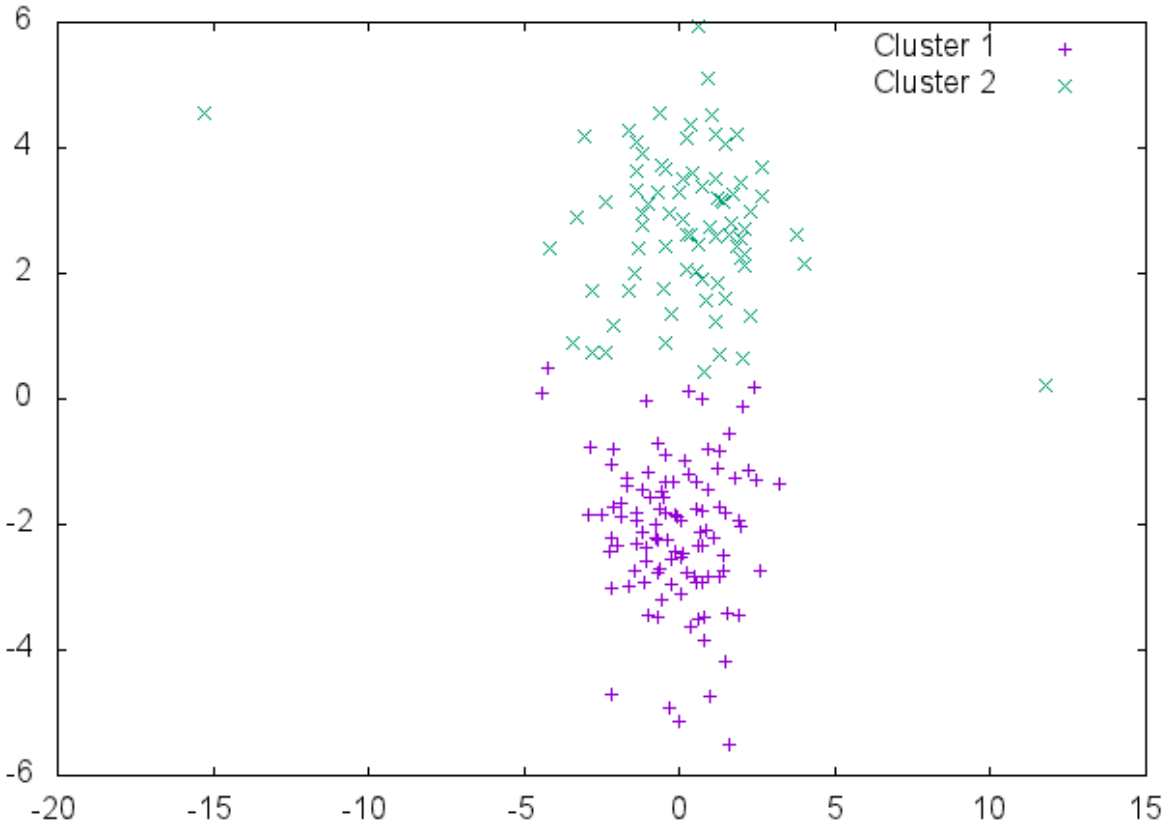


Рисунок 6.17 – Отображение выборки 2П771А

Список логических правил классификации:

(T12 >= 0.0407) и (T6 >= 0.0342) => class=b (74.0/0.0)

(T10 >= 0.699) и (T4 <= 2.92) => class=b (5.0/0.0)

=> class=a (103.0/0.0)

Полученные правила как условия для тестов наглядно изображены на рисунке 6.18.

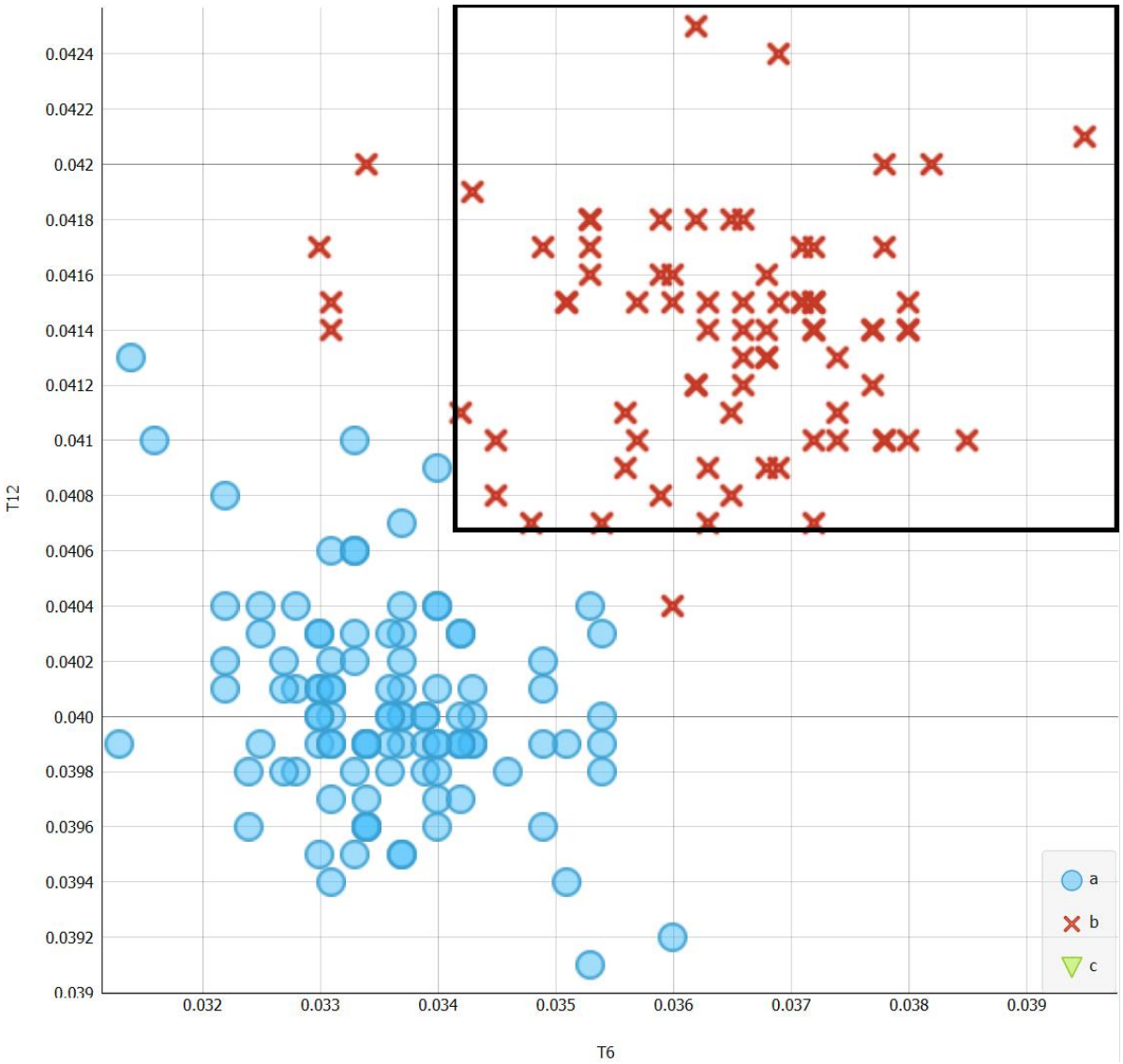


Рисунок 6.18 – Изображение логических правил для классификации 2П771А в пространстве признаков

Пример 3. Арсенид-галиевый диод 3Д713В. Выборка состоит из 258 изделий, собрана из 2 производственных партий. 12 тестов.

На рисунке 6.19 отображены на плоскость изделия выборки с учетом всех тестов с помощью метода многомероного шкалирования.

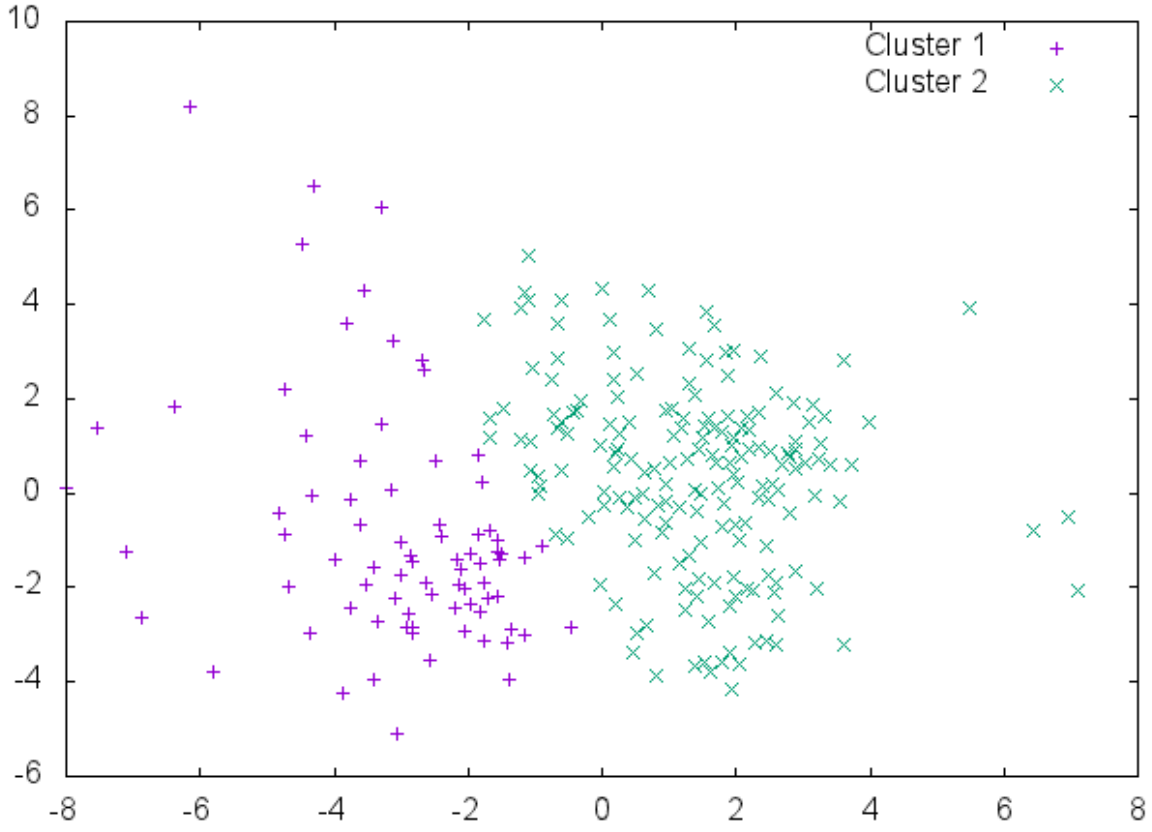


Рисунок 6.19 – Отображение выборки 3Д713В

Список логических правил классификации:

(T2 <= 0.521) и (T11 <= 0.621) => class=b (61.0/0.0)

(T3 >= 0.000151) и (T2 <= 0.528) and (T8 <= 0.209) => class=b (11.0/0.0)

(T10 <= 0.453) и (T2 <= 0.525) => class=b (5.0/0.0)

(T7 >= 0.000159) и (T1 >= 0.000009) => class=b (2.0/0.0)

=> class=a (179.0/0.0)

Полученные правила как условия для тестов наглядно изображены на рисунке 6.20.

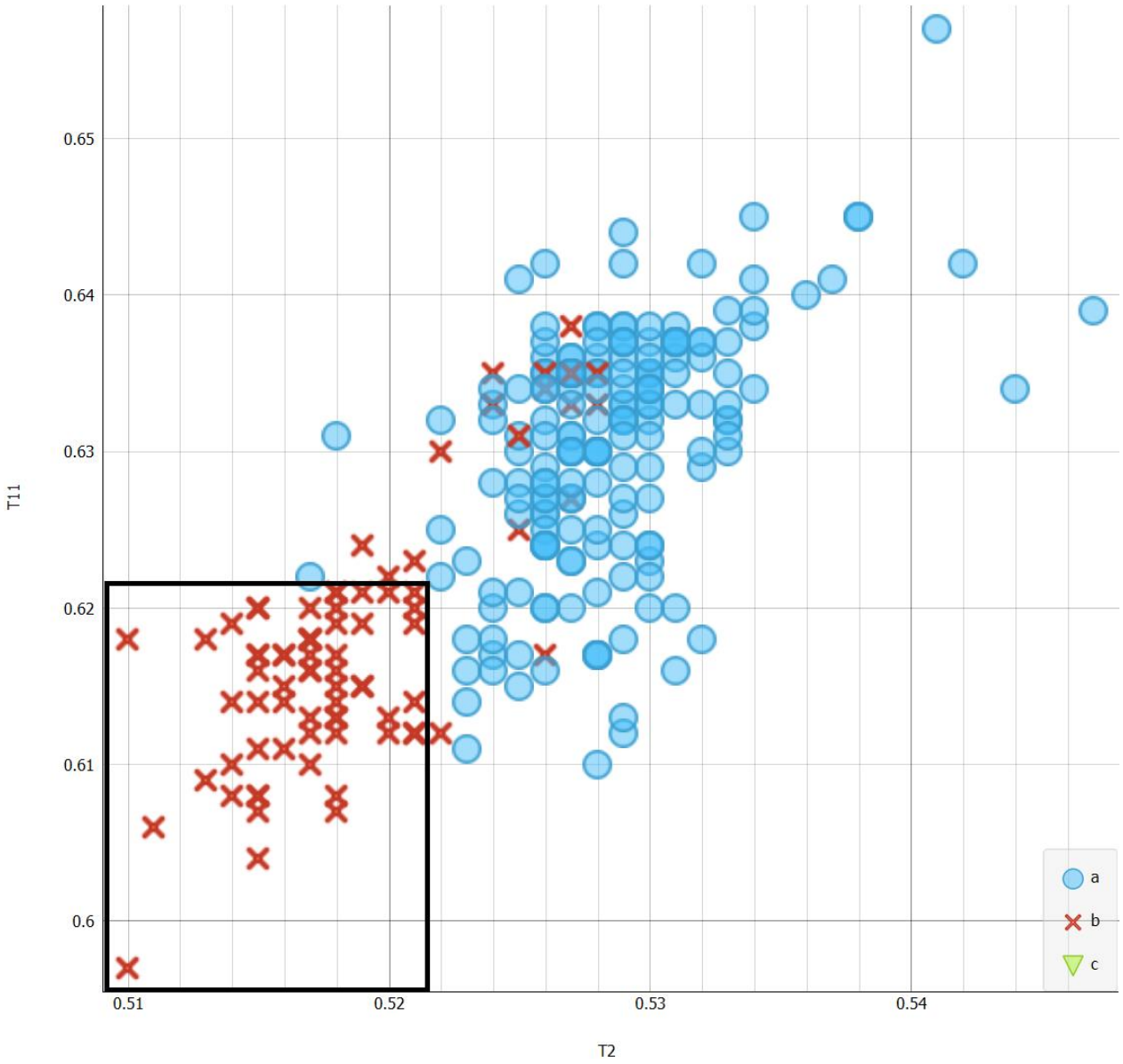


Рисунок 6.20 – Изображение логических правил для классификации 3Д713В в пространстве признаков

Проведенные исследования по использованию методов классификации, основанных на правилах, показывают, что для распознавания группы электрорадиоизделий достаточно использование небольшого числа логических правил (от 1 до 4). Эти правила представляют собой сравнение значения некоторого признака с определенным в процессе построения правил порогом.

Примеры получаемых закономерностей при классификации микросхемы 1526ЛЕ5:

Класс а (196 изделий):

(TEST_34 < 5) ∧ (TEST_127 < 4,76) => класс a (100%)

(TEST_35 < 5) ∧ (TEST_37 < 5) ∧ (TEST_127 < 4,76) => класс a (100%)

Класс b (218 изделий):

(TEST_66 ≥ 0,682) ∧ (TEST_122 ≥ 4,98) => класс b (97,2%)

(TEST_66 ≥ 0,682) => класс b (97,7%)

Класс c (205 изделий):

(TEST_34 ≥ 5) ∧ (TEST_58 < 0,93) => класс c (99%)

6.6 Прогнозирование осложнений инфаркта миокарда

Ещё одной практической задачей, решаемой в рамках данной работы, является задача прогнозирования осложнений инфаркта миокарда (ИМ). Результат болезни пациентов, у которых диагностирован ИМ, может быть различен. Необходимым этапом при назначении лечения является прогнозирование осложнений ИМ, возникновение которых может привести к серьезному ухудшению и даже летальному исходу. На таком прогнозе основывается индивидуальная терапия, которая не может быть одинаковой для всех пациентов. Выявление возможных осложнений является трудной задачей, верно и своевременно решить которую не всегда удается даже опытным специалистам.

В качестве исходных данных используется информация о течении заболевания у 1700 больных ИМ, полученная из историй болезни в Кардиологическом центре городской больницы № 20 г. Красноярска [245]. Таблица содержит сведения о данных анамнеза каждого больного, клинике ИМ, электрокардиографических, лабораторных показателях, лекарственной терапии и особенностях течения заболевания в первые дни ИМ. Признаки принадлежат различным типам, большинство из них бинарные, но имеются также номинальные и количественные. Всего 1700 наблюдений, описанных 124 признаками. В исходных данных много пропущенных значений.

Кратко опишем используемые признаки [245].

Параметры с 4 по 34 отражают наличие или отсутствие различной патологии в организме до развития настоящего ИМ, т.е. в анамнезе пациента. Отражены изменения в тех органах и системах, которые могут оказать влияние на прогноз ИМ. Большее число параметров (с 4 по 26) отражает состояние сердечно-сосудистой системы. Параметры 27-29 характеризуют эндокринную систему, 30-34 - систему органов дыхания. Параметры 35-38 фиксируют артериальное давление пациентов по данным кардиологической бригады и приемного отделения.

Следующие шесть параметров (с 39 по 44) фиксируют осложнения, возникшие в момент транспортировки больного в клинику или в момент госпитализации в отделение реанимации.

Пункты с 45 по 49 характеризуют глубину и локализацию некроза сердечной мышцы. Данные инфарктные изменения относят к трансмуральному (крупноочаговому) ИМ. Однако в имеющейся базе данных эти изменения детализированы, что дает возможность поиска различий течения заболевания в зависимости от формы желудочкового комплекса на ЭКГ.

Параметры 50-75 отражают основной водитель ритма, наличие (отсутствие) аритмий и нарушений проводимости на ЭКГ в момент поступления больного в реанимационное отделение.

Пункты 76-82 отражают вид лекарственного препарата примененного (не примененного) у пациента при проведении фибринолитической терапии - мероприятия, направленного на растворение вызвавшего заболевание тромба в коронарной артерии.

Следующие четыре параметра базы данных (83-86) отражают электролитные сдвиги в крови. Отображены как плавно изменяющиеся параметры - содержание K,Na в крови, так и качественная характеризация сдвигов в электролитном составе крови (гипокалиемия, гипернатриемия). Это интересно не только с точки зрения подтверждения правильности установленных норм, но и выявления того, что более важно для прогноза - плавно изменяющиеся параметры

(количественные показатели) или наличие электролитных сдвигов (качественные характеристики).

Пункты 87-92 содержат информацию о концентрации в крови некоторых ферментов, лейкоцитов, СОЭ, времени госпитализации от момента возникновения ИМ.

Пункты с 93 по 115 отражают течение заболевания в первые дни ИМ (рецидивирование ангинозных болей, температуру тела пациентов) и проводимую на догоспитальном и в первые дни стационарного периодов лекарственную терапию.

Параметры 116-127 отражают возникновение осложнений инфаркта миокарда или исход данного заболевания, т.е. являются ответами при обучении прогностических экспертных систем.

Осложнения: фибрилляция (трепетание) предсердий, суправентрикулярная и желудочковая тахикардии, фибрилляция желудочков, полная атриовентрикулярная (a-v) блокада (пункты 116-120) представляют собой достаточно часто встречающиеся и опасные нарушения ритма и проводимости сердца. Отек легких (пункт 121) - осложнение, заключающееся в недостаточной насосной функции левых отделов сердца. Вышеперечисленные осложнения и летальный исход (пункт 127) возникали у пациентов, внесенных в базу данных, от двух часов с момента поступления в отделение реанимации и до окончания пребывания в стационаре. Это то время, которого достаточно на сбор анамнеза, снятия ЭКГ, выполнения лабораторной диагностики, осуществления прогнозирования и выполнения при необходимости профилактических мероприятий по предупреждению прогнозируемых осложнений. При прогнозировании этих осложнений корректнее не использовать информацию полей 94, 95, 97, 98, 104, 105, 107, 108, поскольку она может быть получена уже после возникновения данных осложнений.

Рецидив инфаркта миокарда и постинфарктная стенокардия (пункты 125-126) возникают спустя три и более суток от начала заболевания и поэтому для их

прогнозирования уместно использовать все (за исключением ответов) поля базы данных.

Хроническая сердечная недостаточность и синдром Дресслера (пункты 123-124) - поздние осложнения инфаркта миокарда. Они возникают, как правило, на третьей неделе заболевания. Для их достаточно точного прогнозирования будет, по всей видимости, недостаточно информации базы данных (т.е. информации первых трех дней заболевания). Но выяснить, какие параметры первых дней инфаркта важны для поздних осложнений, представляется достаточно полезным.

Задача заключается в прогнозировании ряда наиболее значимых осложнений: фибрилляция предсердий (ФП), фибрилляция желудочков (ФЖ), отек легких (ОЛ), разрыв сердца (РС), летальный исход (ЛИ). При этом специалисты имеют потребность в классификаторе, представляющем собой решающее правило в виде логических высказываний [246, 247].

Результаты экспериментов показывают, что точность задачи предлагаемым методом в большинстве случаев превосходит точность решений другими методами, основанными на выявлении и использовании правил (таблица 6.3).

Таблица 6.3 - Точность решения задачи [248]

Задача	RIPPER	CART	C4.5	Random Forest	Adaboost	ОЛРП
ФП	0,66	0,62	0,7	0,7	0,74	0,76
ФЖ	0,867	0,633	0,683	0,833	0,9	0,865
РС	0,786	0,857	0,714	0,857	0,893	0,965
ОЛ	0,692	0,718	0,667	0,769	0,697	0,835
ЛИ	0,74	0,74	0,66	0,76	0,74	0,86

Примеры полученных правил приведены на рисунке 6.21.

```
========================== Отрицательные правила ======================
Правило №1   (63 <  AGE) (INF_IM  < 2) (L_BLOOD  < 12) (TEMPER_3_N  < 2)
Правило №2   (1 <  DLIT_AG) (3 <  TIME_B_S) (NA_R_1_N  < 2) (NOT_NA_3_N  < 2)
Правило №3   (SEX = 0) (STENOK_AN  < 5) (N_R_ECG_P__2 = 0) (NOT_NA_3_N  < 2)
Правило №4   (STENOK_AN  < 3) (IBS_POST  < 2) (SIM_GIPERT = 0) (ZSN_A  < 1) (INF_IM  < 2) (IM_PG_P = 0) (N_P_ECG_P__8 = 0)
Правило №5   (1 <  IBS_POST) (NR_11 = 0) (TEMPER_3_N  < 2) (GEPAR_S_N = 0)
Правило №6   (1 <  FK_STENOK) (NR_11 = 0) (RITM_ECG_P_2 = 0) (N_R_ECG_P__2 = 0) (5 < 18 ROE) (ROE 5 < 18) (TEMPER_3_N  < 3)
Правило №7   (RITM_ECG_P_5 = 0) (NITR_S = 0) (NA_R_1_N  < 1) (NOT_NA_1_N  < 3) (ANT_CA_S_N = 1)
Правило №8   (SEX = 1) (STENOK_AN  < 6) (ZAB_LEG_02 = 0) (130 <  NA_BLOOD) (TIME_B_S  < 9) (NA_R_1_N  < 1)
Правило №9   (2 <  GB) (NR_04 = 0) (ENDOCR_02 = 0) (ZAB_LEG_02 = 0) (INF_IM  < 4) (N_R_ECG_P_ = 0) (NA_R_1_N  < 2)
Правило №10   (STENOK_AN  < 4) (ANT_IM  < 1) (RITM_ECG_P = 1)
========================== Положительные правила ======================
Правило №1   (STENOK_AN  < 4) (ENDOCR_02 = 0) (FIB_G_POST = 0) (1 <  ANT_IM) (LID_S_N = 1) (GEPAR_S_N = 1)
Правило №2   (58 <  AGE) (70 <  D_AD_ORIT) (0,23 <  ALT_BLOOD)
Правило №3   (S_AD_ORIT  < 140) (R_AB_2_N  < 2)
Правило №4   (ENDOCR_02 = 0) (O_L_POST = 0) (1 <  NA_R_1_N) (LID_S_N = 1)
Правило №5   (D_AD_ORIT  < 80)
Правило №6   (D_AD_ORIT  < 90)
Правило №7   (MP_TP_POST = 0) (2 <  ANT_IM) (RITM_ECG_P_2 = 0) (K_BLOOD  < 4,1) (NA_BLOOD  < 140)
Правило №8   (FK_STENOK  < 3) (S_AD_ORIT  < 140)
Правило №9   (D_AD_ORIT  < 90)
Правило №10   (ZSN_A  < 1) (ANT_IM  < 1) (FIBR_TER_0_1 = 0) (8 <  L_BLOOD) (TEMPER_1_N  < 1)
```

Рисунок 6.21 - Примеры закономерностей

Точность решения задачи прогнозирования осложнений методом оптимальных логических решающих правил сопоставима с точностью решения методом нейронных сетей (Горбань А.Н.). При этом предлагаемый метод в явном виде предоставляет правила, по которым принимается решение, что является преимуществом этого метода для специалистов (таблица 6.4).

Таблица 6.4 - Сравнение результатов

Прогнозируемое осложнение	Класс	Точность классификации	
		Нейронные сети	ОЛРП
ФП	чувствительность	0,90	0,78
	специфичность	0,70	0,74
ФЖ	чувствительность	0,76	0,83
	специфичность	0,70	0,90
РС	чувствительность	0,80	0,93
	специфичность	0,70	1,00
ОЛ	чувствительность	0,85	0,89
	специфичность	0,80	0,78
ЛИ	чувствительность	0,86	0,85
	специфичность	0,80	0,87

Выводы к главе 6

Предложен и исследован метод классификации электрорадиоизделий космического применения по однородным партиям, состоящий в формировании логических решающих правил, использующих результаты тестовых воздействий. Предложенный метод позволяет эффективно решать задачу формирования однородных партий при проведении отбраковочных испытаний ЭРИ.

Предлагаемый метод классификации электрорадиоизделий (ЭРИ) по однородным партиям позволяет получать явные правила классификации, и выполнять классификацию на основе части признаков (результатов тестовых

воздействий), которые являются значимыми (значимыми комплексно или комбинаторно, а не только индивидуально).

Использование предлагаемого подхода дает возможность принимать решение о принадлежности изделия партии по данным небольшого числа тестов с использованием простых правил сравнения. Оказывается достаточным использование весьма небольшого числа признаков (выбранных определенным образом в процессе построения правил из всего набора признаков – результатов тестов) для успешной классификации ЭРИ. Таким образом, в работе решена задача выявления закономерностей для классификации ЭРИ по результатам дополнительных отбраковочных испытаний с целью дальнейшего прогнозирования показателей безотказности электронной компонентной базы.

Применение предлагаемого метода для решения задачи прогнозирования осложнений инфаркта миокарда дает значимые преимущества (по сравнению, например, с ранее использованным способом решения задачи с помощью искусственных нейронных сетей), заключающиеся в возможности объяснения и обоснования результатов прогнозирования, с точностью распознавания, превосходящей известные логические алгоритмы классификации.

Повышение точности обеспечивается новыми моделями и алгоритмами оптимизации для поиска логических решающих правил. Результаты диссертации уже получили дальнейшее развитие в работах последователей. На основе метода оптимальных логических решающих правил разрабатываются новые алгоритмы и модели, например [249, 250].

ЗАКЛЮЧЕНИЕ

В диссертации предложен новый метод анализа данных для поддержки принятия решений при распознавании, состоящий в выявлении в данных оптимальных логических закономерностей с помощью алгоритмов псевдобулевой оптимизации, позволяющий успешно решать задачи классификации, предоставляя обоснование и объяснение решений в виде логических правил, подтверждаемых прецедентами.

Цель диссертации достигнута путём решения поставленных задач, а именно:

1. Проведён анализ существующих способов выявления закономерностей в данных и исследованы свойства моделей оптимизации для нахождения закономерностей. Показано, что существующие алгоритмы не гарантируют получения сильных закономерностей с максимальным покрытием. Разработана новая модель оптимизации для нахождения сильных охватывающих закономерностей в данных.

2. Изучение особенностей применения различных типов закономерностей позволило установить, что использование первичных закономерностей уменьшает число нераспознанных наблюдений, а использование сильных охватывающих закономерностей позволяет снизить ошибки в распознавании. Разработан новый подход к поддержке принятия решений при распознавании, заключающийся в совместном использовании закономерностей двух типов – сильных первичных и сильных охватывающих закономерностей.

3. Разработана единая модель оптимизации в рамках метода оптимальных логических решающих правил для поиска пары (первичной и охватывающей) закономерностей.

4. Построен новый алгоритм условной псевдобулевой оптимизации на основе схемы ветвей и границ и поиска среди граничных точек допустимой области, позволяющий находить лучшее решение, чем жадный алгоритм. Алгоритм не требует алгебраического задания функций (целевой и ограничений),

функции могут быть заданы алгоритмически (оптимизация черного ящика), что позволяет его применять для задач с нелинейными функциями, в том числе для задачи поиска логических закономерностей в данных. Экспериментально показана эффективность приближенного варианта алгоритма с использованием ранней остановки: достаточно произвести лишь несколько итераций алгоритма, чтобы получить решение, лучшее, чем получаемое жадным алгоритмом.

5. Разработан алгоритм поиска пары закономерностей (сильной первичной и сильной охватывающей) с использованием нового алгоритма условной псевдобулевой оптимизации.

6. Разработана новая модель оптимизации для нахождения оптимального назначения порогов количественных признаков, которая не просто определяет наименьшее число порогов, достаточных для разделения наблюдений разных классов, а позволяет выбрать такие пороги, которые наилучшим образом разделяют наблюдения разных классов в пространстве булевых признаков.

7. Впервые предложена комплексная процедура ускорения поиска закономерностей, делающая возможным применение метода для случаев большого объема данных. Предлагаемое улучшение состоит в применении нового способа выбора базовых наблюдений для формирования закономерностей, отборе признаков путём решения задачи псевдобулевой оптимизации и применении приближенного варианта нового алгоритма псевдобулевой оптимизации.

8. Разработана процедура повышения интерпретируемости классификатора, основанного на закономерностях, заключающаяся в нахождении закономерностей с лучшей обобщающей способностью, новой схемы использования одновременно двух видов закономерностей (сильной первичной и сильной охватывающей) и отборе достаточного числа закономерностей с ограниченным числом условий на основе решения задачи условной псевдобулевой оптимизации.

9. Решена задача выявления закономерностей для классификации электрорадиоизделий космического применения по результатам дополнительных отбраковочных испытаний с целью дальнейшего прогнозирования показателей безотказности электронной компонентной базы. Использование предлагаемого

подхода дает возможность принимать решение о принадлежности изделия партии по данным небольшого числа тестов с использованием простых правил сравнения.

Совокупность новых моделей и алгоритмов являются содержанием нового метода решения задач классификации, результаты распознавания в которых должны быть обоснованы и интерпретированы в виде логических правил. Новый метод – метод оптимальных логических решающих правил – основан на нахождении оптимальных решений при выявлении логических закономерностей в соответствие с моделями оптимизации, относящимися к классу задач оптимизации монотонных псевдобулевых функций с монотонными ограничениями, при этом функции заданы алгоритмически. Для решения этих задач использован новый алгоритм условной псевдобулевой оптимизации, реализующий свойства данного класса задач и основанного на поиске среди граничных точек допустимой области и схеме ветвей и границ. Применение оптимальных логических решающих правил обеспечивает повышение точности решения задач классификации с выполнением условий доказательности и интерпретируемости.

Основные результаты работы опубликованы в следующих статьях автора.

А) **Статьи в российских периодических изданиях**, рекомендованных ВАК:

1. Масич И.С. Метод оптимальных логических решающих правил для задач распознавания и прогнозирования. // Системы управления и информационные технологии. 2019. Т. 75. № 1. С. 31-37.

2. Кузьмич Р.И., Масич И.С., Ступина А.А. Модели формирования закономерностей в методе логического анализа данных. // Системы управления и информационные технологии. 2017. Т. 67. № 1. С. 33-37.

3. Федосов В.В., Казаковцев Л.А., Масич И.С. Метод нормировки исходных данных испытаний электрорадиоизделий космического применения для алгоритма автоматической группировки. // Системы управления и информационные технологии. 2016. Т. 65. № 3. С. 92-96.

4. Орлов В.И., Казаковцев Л.А., Масич И.С. Применение критерия силуэта в алгоритме автоматической группировки электрорадиоизделий космического применения. // Вестник Сибирского государственного аэрокосмического университета им. академика М.Ф. Решетнева. 2016. Т. 17. № 4. С. 883-890.

5. Антамошкин А.Н., Масич И.С. Поисковые алгоритмы условной псевдобулевой оптимизации. // Системы управления, связи и безопасности. 2016. № 1. С. 103-145.

6. Казаковцев Л.А., Масич И.С., Орлов В.И., Федосов В.В. Быстрый детерминированный алгоритм для классификации электронной компонентной базы по критерию равнонадежности. // Системы управления и информационные технологии. 2015. Т. 62. № 4. С. 39-44.

7. Антамошкин А.Н., Масич И.С. Обнаружение закономерностей в данных для распознавания объектов как задача условной псевдобулевой оптимизации. // Вестник Сибирского государственного аэрокосмического университета им. академика М.Ф. Решетнева. 2015. Т. 16. № 1. С. 16-21.

8. Казаковцев Л.А., Масич И.С., Орлов В.И., Федосов В.В. Детерминированный алгоритм для задач классификации электрорадиоизделий. // Информационные технологии моделирования и управления. 2015. Т. 96. № 6. С. 519-525.

9. Кузьмич Р.И., Масич И.С. Модификация целевой функции при построении паттернов для увеличения различности правил в модели классификации. Системы управления и информационные технологии. 2014. Т. 56. № 2.

10. Казаковцев Л.А., Орлов В.И., Ступина А.А., Масич И.С. Задача классификации электронной компонентной базы. // Вестник Сибирского государственного аэрокосмического университета им. академика М.Ф. Решетнева. 2014. № 4 (56). С. 55-61.

11. Антамошкин А.Н., Масич И.С. Выбор логических закономерностей для построения решающего правила распознавания. // Вестник Сибирского

государственного аэрокосмического университета им. академика М.Ф. Решетнева. 2014. № 5 (57). С. 20-25.

12. Масич И.С., Краева Е.М. Отбор закономерностей для построения решающего правила в логических алгоритмах распознавания. // Системы управления и информационные технологии. 2013. Т. 51. № 1.1. С. 170-173.

13. Кузьмич Р.И., Масич И.С. Построение модели классификации как композиции информативных паттернов. // Системы управления и информационные технологии. 2012. Т. 48. № 2. С. 18-22.

14. Антамошкин А.Н., Масич И.С. Исследование свойств задач оптимизации при поиске логических закономерностей в данных. // Системы управления и информационные технологии. 2011. N4.1(46). С. 111-115.

15. Масич И.С., Краева Е.М., Кузьмич Р.И., Гулакова Т.К. Сравнительный анализ методов классификации данных на практических задачах прогнозирования и диагностики. // Системы управления и информационные технологии. 2011. N1(43). С. 20-25.

16. Головенкин С.Е., Гулакова Т.К., Кузьмич Р.И., Масич И.С., Шульман В.А. Модель логического анализа для решения задачи прогнозирования осложнений инфаркта миокарда. // Вестник СибГАУ, выпуск 4(30). 2010. С. 68-73.

17. Antamoshkin A.N., Masich I.S. Combinatorial optimization and rule search in logical algorithms of machine learning. // Engineering & automation problems (Проблемы машиностроения и автоматизации). V.7. N.1. 2010. P.52-57.

18. Масич И.С. Модель логического анализа для прогнозирования осложнений инфаркта миокарда. // Информатика и системы управления. №3 (25). 2010. С.48-56.

19. Masich I.S. Combinatorial optimization in foundry production planning. // Вестник СибГАУ, выпуск 2(23). 2009. С.40-44.

20. Масич И.С. Комбинаторная оптимизация в задаче классификации. // Системы управления и информационные технологии. 2009. №1.2(35). С. 283-288.

21. Antamoshkin A.N., Masich I.S. Unimprovable algorithm for monotone pseudo-Boolean function conditional optimization. // Engineering & automation problems (Проблемы машиностроения и автоматизации). V. 6. N. 1. 2008. P. 71-75.

22. Antamoshkin A.N., Masich I.S. Heuristic search algorithms for monotone pseudo-boolean function conditional optimization. // Engineering & automation problems (Проблемы машиностроения и автоматизации). № 3. 2007. P. 41-45.

23. Antamoshkin A.N., Masich I.S. Identification of pseudo-Boolean function properties. // Engineering & automation problems (Проблемы машиностроения и автоматизации). 2/2007. P. 66-69.

24. Масич И.С., Шарыпова К.В. Оптимизация загрузки производственных мощностей литейного производства. // Системы управления и информационные технологии. №3(29). 2007. С. 76-80.

25. Масич И.С. Приближенные алгоритмы поиска граничных точек для задачи условной псевдобулевой оптимизации. // Вестник СибГАУ, 8470, 1(8). 2006. С. 39-43.

Б) **статьи в зарубежных изданиях**, включенных в международные базы цитирования, рекомендованные ВАК:

26. Kazakovtsev L.A., Masich I.S. A branch-and-bound algorithm for a pseudo-boolean optimization problem with black-box functions. // Facta Universitatis, Series Mathematics and Informatics. Vol. 33. No 2 (2018). P. 337–360. (WoS)

27. Kuzmich, R., Masich, I., Stupina, A., Kazakovtsev, L. Algorithmic procedure for constructing the truncated basic set of characteristics in the method of logical analysis of data // Proceedings of the 30th International Business Information Management Association Conference. IBIMA 2017. P. 5592-5597. (SCOPUS)

28. Masich I.S., Kazakovtsev L.A., Stupina A.A. Optimization Models for Detection of Patterns in Data. // Optimization Problems and their Applications. Proceedings of the School-Seminar on Optimization Problems and their Applications (OPTA-SCL 2018). Omsk, Russia, July 8-14, 2018. P. 264-275. (SCOPUS)

29. Kazakovtsev L.A., Orlov V.I., Stashkov D.V., Antamoshkin A.N., Masich I. S. Improved model for detection of homogeneous production batches of electronic

components. // IOP Conference Series: Materials Science and Engineering 255. 2017. 012004. (SCOPUS)

30. Kraeva E.M., Masich I.S. Analysis and improvement of calculation procedure of high-speed centrifugal pumps. // IOP Conference Series: Materials Science and Engineering 255. 2017. 012005. (SCOPUS)

31. Antamoshkin A.N., Masich I.S. Combinatorial optimization in foundry practice. // IOP Conference Series: Materials Science and Engineering. 2016. C. 012001. (SCOPUS)

32. Antamoshkin A.N., Masich I.S. Selection of logical patterns for constructing a decision rule of recognition. // IOP Conference Series: Materials Science and Engineering. 2016. C. 012002. (SCOPUS)

33. Kraeva E.M., Masich I.S. Calculation and optimization of parameters in low-flow pumps. // IOP Conference Series: Materials Science and Engineering. 2016. C. 012019. (SCOPUS)

34. Antamoshkin A.N., Masich I.S., Kuzmich R.I. Heuristics and criteria for constructing logical patterns in data. // IOP Conference Series: Materials Science and Engineering. 2015. C. 012003. (SCOPUS)

35. Kazakovtsev L.A., Antamoshkin A.N., Masich I.S. Fast deterministic algorithm for eee components classification. // IOP Conference Series: Materials Science and Engineering. 2015. C. 012015. (SCOPUS)

36. Kazakovtsev L.A., Stupina A.A., Orlov V.I., Karaseva M.V., Masich I.S. Clustering Methods for Classification of Electronic Devices by Production Batched and Quality Classes. // FACTA UNIVERSITATIS. Ser. Math. Inform. Vol. 30, No 5, 2015, pp. 567-581. (Math. Reviews)

37. Antamoshkin A., Masich I. Pseudo-Boolean Optimization in Case of an Unconnected Feasible Set. // Models and Algorithms for Global Optimization. Springer Optimization and Its Applications. Vol. 4. 2007. P. 111-122. (SCOPUS)

Монографии, препринты и главы в книгах:

1. Кузьмич Р.И., Масич И.С. Модификации метода логического анализа данных для задач классификации : монография – Красноярск: Сиб. федер. ун-т, 2018. – 180 с.

2. Орлов В.И., Федосов В.В., Казаковцев Л.А., Масич И.С., Проценко В.В., Сташков Д.В. Алгоритмическое обеспечение поддержки принятия решений по отбору изделий микроэлектроники для космического приборостроения : монография; СибГУ им. М. Ф. Решетнева. – Красноярск, 2017. – 228 с.

3. Казаковцев Л. А., Масич И. С., Орлов В. И., Проценко В. В., Федосов В. В. Разработка алгоритмического обеспечения анализа однородности партий электрорадиоизделий для комплектации РЭА КА : монография; Сиб. гос. аэрокосмич. ун-т. – Красноярск, 2016. – 192 с.

4. Антамошкин А.Н., Масич И.С., Кузьмич Р.И. Комбинаторная оптимизация при логической классификации : монография. – Красноярск: КрасГАУ. 2015. – 130 с.

5. Масич И.С. Поисковые алгоритмы условной псевдобулевой оптимизации : монография. – Красноярск: СибГАУ. 2013. – 160 с.

6. Масич И.С., Крушенко Г.Г. Алгоритмы случайного поиска в практических задачах условной оптимизации. Препринт № 1-12. – Красноярск: ИВМ СО РАН; СибГАУ. 2012. – 38 с.

7. Antamoshkin A., Masich I. Pseudo-Boolean Optimization in Case of an Unconnected Feasible Set. // In: “Models and Algorithms for Global Optimization”. Springer Optimization and Its Applications. Vol. 4. 2007. P. 111-122. (SCOPUS) (глава в книге)

Программы для ЭВМ, зарегистрированные в Роспатенте:

1. Орлов В.И., Федосов В.В., Казаковцев Л.А., Масич И.С. Система интеллектуального анализа данных результатов тестовых испытаний электрорадиоизделий космического применения. // Свидетельство о государственной регистрации программы для ЭВМ №2017617555 от 6.07.2017.

2. Антамошкин А.Н., Кузьмич Р.И., Масич И.С. Модифицированный метод логического анализа данных. // Свидетельство о государственной регистрации программы для ЭВМ №2016619162 от 15.08.2016.

3. Казаковцев Л.А., Масич И.С., Орлов В.И., Федосов В.В. Программа автоматической группировки электрорадиоизделий. // Свидетельство о государственной регистрации программы для ЭВМ №2016616924, 22.06.2016.

4. Казаковцев Л.А., Масич И.С., Орлов В.И., Федосов В.В. Система автоматизированного формирования и контроля специальных партий электрорадиоизделий. // Свидетельство о государственной регистрации программы для ЭВМ №2016611353, 1.02.2016.

5. Масич И.С., Кузьмич Р.И., Краева Е.М. Логический анализ данных в задачах классификации. // Свидетельство о государственной регистрации программы для ЭВМ № 2011612265, 2011.

6. Масич И.С., Краева Е.М. Алгоритмы условной псевдобулевой оптимизации для решения задач рюкзачного типа. // Свидетельство о государственной регистрации программы для ЭВМ № 2011613446, 2011.

СПИСОК ИСПОЛЬЗОВАННЫХ ИСТОЧНИКОВ

1. Moreira L. The use of Boolean concepts in general classification contexts. Lausanne. // École Polytechnique Fédérale De Lausanne, 2000.
2. Fayyad U. M., Piatetsky-Shapiro G., Smyth P., Uthurusamy R. Advances in Knowledge Discovery and Datti Mining. AAAI Press. // The MIT Press, 1996.
3. Jain A. Iv., Dubes R. C. Algorithms for Clustering Data. // Prentice Hall, Engelwood Clifts, 1988.
4. Gersho A., Gray R. M. Vector Quantization and Signal Compression. // Ivluwer Academic Publishers. Boston, 1992.
5. Kohonen T. Self-Organizing Maps. // Springer-Verlag. Berlin, 1995.
6. Cheeseman P., Stutz J. Bayesian classification (autoclass): Theory and results. In Fayyad U. M., Piatetsky-Shapiro G., Smyth P., Uthurusamy R., editors. Advances in Knowledge Discovery and Data Mining, chapter 6. P. 153 - 180. AAAI Press, The MIT Press, 1996.
7. Mitchell T. Machine Learning. // McGraw Hill, 1997.
8. Vapnik V. N. The Nature of Statistical Learning Theory. // Springer. New York, 1995.
9. Bishop C. M. Neural Networks for Pattern Recognition. // Clarendon Press. Oxford, 1995.
10. Furnkranz J., Gamberger D., Lavrac N. Foundations of Rule Learning. // Springer-Verlag, 2012.
11. Kl¨osgen, W. Explora. A multipattern and multistrategy discovery assistant. // In Fayyad U.M. Piatetsky-Shapiro G., Smyth P., Uthurusamy R., editors Advances in Knowledge Discovery and Data Mining, chap. 10. AAAI Press, 1996. P. 249 - 271.
12. Wrobel S. An algorithm for multi-relational discovery of subgroups. In Proceedings of the 1st European Symposium on Principles of Data Mining and Knowledge Discovery (PKDD-97). // Springer-Verlag, Berlin, 1997. P. 78–87.

13. Bay S.D., Pazzani M.J. Detecting group differences. Mining contrast sets. Data Mining and Knowledge Discovery 5(3), 2001. P. 213 – 246.
14. Morishita S., Sese, J. Traversing itemset lattice with statistical metric pruning. // In Proceedings of the 19th ACM SIGMOD-SIGACT-SIGART Symposium on Principles of Database Systems (PODS-00), 2000. P. 226 – 236.
15. Dong G., Li J. Efficient mining of emerging patterns. Discovering trends and differences. // In Proceedings of the 5th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining (KDD-99) CA, San Diego. 1999. P. 43 – 52.
16. Kralj Novak P., Lavraˇc N., Webb G.I. Supervised descriptive rule discovery A unifying survey of contrast set, emerging pattern and subgroup mining // Journal of Machine Learning Research 10. 2009. P. 377 – 403.
17. Zimmermann A., De Raedt L. Cluster grouping: From subgroup discovery to clustering. // Machine Learning 77(1). 2009. P. 125 – 159.
18. Lindgren T., Bostr¨om H. Resolving rule conflicts with double induction. // Intelligent Data Analysis 8(5). 2004. P. 457 – 468.
19. Hhn J., Hllermeier E. Furia: an algorithm for unordered fuzzy rule induction. // Data Mining and Knowledge Discovery 19(3). 2009. P. 293 – 319.
20. Eineborg M., Bostr¨om H. Classifying uncovered examples by rule stretching. // Proceedings of the Eleventh International Conference on Inductive Logic Programming (ILP-01). In Rouveirol C., Sebag M. (eds.), Springer Verlag, Strasbourg, France. 2001. P. 41 – 50.
21. Dˇzeroski S., Lavraˇc N. Relational Data Mining: Inductive Logic Programming for Knowledge Discovery in Databases. Springer-Verlag. 2001.
22. De Raedt L. Logical and Relational Learning. Springer-Verlag. 2008.
23. Fürnkranz J. Separate-and-Conquer Rule Learning. // Artificial Intelligence Review 13(1). 1999. P. 3-54.
24. Mitchell T. M. 1980. The Need for Biases in Learning Generalizations. Tech. rep., Computer Science Department, Rutgers University, New Brunswick, MA.

Reprinted in Shavlik, J. W. & Dietterich, T. G. (eds.) Readings in Machine Learning. Morgan Kaufmann. 1991.

25. Michalski, R. S. On the Quasi-Minimal Solution of the Covering Problem. // In Proceedings of the 5th International Symposium on Information Processing (FCIP-69), Vol. A3 (Switching Circuits). Bled. Yugoslavia. 1969. P. 125-128.
26. Pagallo G., Haussler D. Boolean Feature Discovery in Empirical Learning. // Machine Learning 5: 1990. P. 71-99.
27. Clark P., Niblett T. The CN2 induction algorithm. // Machine Learning 3(4). 1989. P. 26-283.
28. Clark P., Boswell R. Rule induction with CN2. // Some recent improvements. In Proceedings of the 5th European Working Session on Learning (EWSL-91). Springer-Verlag. Porto. Portugal. 1991. P. 151-163.
29. Quinlan J.R. Learning logical definitions from relations. // Machine Learning 5. 1990. P. 239-266.
30. F¨urnkranz J. Pruning algorithms for rule learning. // Machine Learning 27(2). 1997. P. 139-171.
31. Witten I.H., Frank E. Data Mining - Practical Machine Learning Tools and Techniques with Java Implementations. Morgan Kaufmann Publishers, 2nd edn. 2005.
32. Webb G.I. OPUS. An efficient admissible algorithm for unordered search. // Journal of Artificial Intelligence Research 5. 1995. P. 431-465.
33. Cendrowska J. PRISM. An Algorithm for Inducing Modular Rules. // International Journal of Man-Machine Studies 27. 1987. P. 349-370.
34. Mooney R. J. Encouraging Experimental Results on Learning CNF. // Machine Learning 19. 1995. P. 79-92.
35. Cohen W. W. Efficient Pruning Methods for Separate-and-Conquer Rule Learning Systems. // In Proceedings of the 13th International Joint Conference on Artificial Intelligence (IJCAI-93). Morgan Kaufmann. Chambery, France.1993. P. 988-994.

36. Cohen W. W. Fast Effective Rule Induction. // In Prieditis A. Russell S. (eds.) Proceedings of the 12th International Conference on Machine Learning (ML-95). Morgan Kaufmann. Lake Tahoe. CA. 1995. P. 115-123.
37. Лбов Г.С., Котюков В.И., Манохин А.Н. Об одном алгоритме распознавания в пространстве разнотипных признаков // Тр. ИМ СО АН СССР «Вычислительные системы». 1973. Вып. 55. С. 98-107.
38. Yang J., Tiyyagura A., Chen F., Honavar V. Feature Subset Selection for Rule Induction Using RIPPER. // In Proc. of the Genetic and Evolutionary Computation Conference. Oriando. Florida. USA. 1999.
39. Asadi S., Shahrabi J. RipMC. RIPPER for multiclass classification. // Neurocomputing, 191. 2016. P. 19-33.
40. Govada V. S., Thomas I., Samal, S. K. Sahay. Distributed Multi-class Rule Based Classification Using RIPPER. // IEEE International Conference on Computer and Information Technology (CIT). Nadi. 2016. P. 303-309.
41. De Raedt, L., Van Laer, W. Inductive Constraint Logic. // In Proceedings of the 5th Workshop on Algorithmic Learning Theory (ALT-95). Springer-Verlag, 1995. P.80-94.
42. Clark P., Boswell R. Rule Induction with CN2. // Some Recent Improvements. In Proceedings of the 5th European Working Session on Learning (EWSL-91). Springer-Verlag: Porto. Portugal, 1991. P. 151-163.
43. Mooney R. J., Califf M. E. Induction of First-Order Decision Lists: Results on Learning the Past Tense of English Verbs. // Journal of Artificial Intelligence Research 3, 1995. P. 1-24.
44. Лбов Г.С. Методы обработки разнотипных экспериментальных данных. Новосибирск: Наука, 1981. 160 с.
45. Asadi S., Shahrabi J. ACORI: a novel ACO algorithm for Rule Induction. // Knowledge-Based Systems 97. P. 175-187.
46. Michalski R. S. Pattern Recognition and Rule-Guided Inference. // IEEE Transactions on Pattern Analysis and Machine Intelligence 2, 3, 1980. P.49-361.

47. Quinlan J. R., Cameron-Jones R. M. Induction of Logic Programs. // FOIL and Related Systems. New Generation Computing 13 (3,4): Special Issue on Inductive Logic Programming, 1995. P. 287-312.
48. Furnkranz J., Widmer G. Incremental Reduced Error Pruning. // In Cohen W., Hirsh, H. (eds.) Proceedings of the 11th International Conference on Machine Learning (ML-94), Morgan Kaufmann: New Brunswick, NJ, 1994. P. 70-77.
49. Muggleton S. H. Inverse Entailment and Progol. // New Generation Computing 13(3,4), Special Issue on Inductive Logic Programming. 1995. P. 245-286.
50. De Raedt L., Thon I. Probabilistic rule learning. // In Paolo Frasconi, Francesa Alessandra Lisi, International Conference on Inductive Logic Programming (ILP), volume 6489 of LNCS, (Lecture Notes in Computer Science). Springer. Berlin/Heidelberg, 2011. P. 47-58.
51. De Raedt L., Dries A., Thon I., Van den Broeck G., Verbeke M. Inducing Probabilistic Relational Rules from Probabilistic Examples. // In Proceedings of 24th International Joint Conference on Artificial Intelligence (IJCAI), AAAI Press, 2015. P. 1835–1843.
52. Widmer G. Combining Knowledge-Based and Instance-Based Learning to Exploit Quantitative Knowledge. // Informatica 17. Special Issue on Multistrategy Learning, 1993. P. 371-385.
53. Weiss S. M., Indurkhya N. Rule-Based Regression. // In Bajcsy, R. (ed.) Proceedings of the 13th International Joint Conference on Artificial Intelligence (IJCAI-93), 1993. P.1072-1078.
54. Weiss S. M., Indurkhya N. Rule-Based Machine Learning Methods for Functional Prediction. // Journal of Artificial Intelligence Research 3, 1995. P. 383-403.
55. Karalič A., Bratko I. First Order Regression. // Machine Learning, 26, 1997. P. 147-176.
56. Zenkoˇ et al., Zenko B., Zeroski S. Dˇ, Struyf J.ˇ Learning predictive clustering rules. // In Proc. 4th International Workshop on Knowledge Discovery in Inductive Databases. Springer, 2005. P. 234-250.

57. Sikora M., Wróbel L., Gudyś A., Guide R. A guided separate-and-conquer rule learning in classification, regression, and survival settings. // Knowledge-Based Systems 173, 2019. P. 1-14.
58. Quinlan J. R. Induction of Decision Trees. // Machine Learning 1, 1986. P. 81-106.
59. Bostrom H. Covering vs. Divide-and-Conquer for Top-Down Induction of Logic Programs. // In Proceedings of the 14th International Joint Conference on Artificial Intelligence (IJCAI-95), 1995. P. 1194-1200.
60. Quinlan J. R. C 4.5. Programs for Machine Learning. Morgan Kaufmann. San Mateo, CA, 1993.
61. Rivest R. L. Learning Decision Lists. // Machine Learning 2, 1987. P. 229-246.
62. Дюкова Е. В. , Журавлёв Ю. И., Прокофьев П. А. Логические корректоры в задаче классификации по прецедентам. // Ж. вычисл. матем. и матем. физ., **57**:11, 2017. С. 1906-1927.
63. Boros E., Hammer P.L., Ibaraki T., Kogan A., Mayoraz E., Muchnik I. // An Implementation of logical analysis of data. IEEE Trans. Knowl. Data Eng. **12**(2), 2000. P. 292-306
64. Bonates T. O. Large margin rule-based classifiers. // In J. J. Cochran (ed.), Wiley encyclopedia of operations research and management science. New York. Wiley, 2010. P. 1–12.
65. Boros E., Ibaraki T., Makino K. Extensions of partially defined boolean functions with missing data. // RUTCOR Research Report RRR 06-96, RUTCOR, Rutgers University, 1996.
66. Chikalov et al. Three Approaches to Data Analysis. // Test Theory, Rough Sets and Logical Analysis of Data. Intelligent Systems Reference Library 41. Springer-Verlag. Berlin. Heidelberg, 2013.
67. Hammer P.L. Partially defined boolean functions and cause-effect relationships. // In: International Conference on Multi-attribute Decision Making Via OR-based Expert Systems. University of Passau. Passau. Germany, April, 1986.

68. Crama Y., Hammer P.L., Ibaraki T. Cause-effect relationships and partially defined Boolean functions. // Annals Oper Res. 16, 1988. P. 299–325.
69. Boros E., Hammer P. L., Kogan A., Mayoraz E., Mnehnik I. Logical Analysis of Data Overview. // RTR 1-94, RUTCOR Rutgers University Center For Operations Research, 1994.
70. Boros E., Hammer P. L., Ibaraki T., Kogan A., Mayoraz E., Muchnik I. An implementation of Logical Analysis of Data. // RRR 22-96. RUTCOR Rutgers University- Center For Operations Research, 1996.
71. Boros E., Hammer P., Ibaraki T., Kogan A. Logical analysis of numerical data. // Mathematical Programming 79, 1997. P. 163–190.
72. Alexe G., Alexe S., Bonates T.O. et al. Logical analysis of data – the vision of Peter L. // Hammer Annals of Mathematics and Artificial Intelligence 49, 2007. P. 265-312.
73. Растригин Л.А., Фрейманис Э.Э. Алгоритмы случайного поиска. // Проблемы случайного поиска, 11, 1988. С.9-25.
74. Масич И.С. Комбинаторная оптимизация в задаче классификации. // Системы управления и информационные технологии, 2009, №1.2(35). С. 283-288.
75. Антамошкин А.Н., Масич И.С. Обнаружение закономерностей в данных для распознавания объектов как задача условной псевдобулевой оптимизации. Вестник Сибирского государственного аэрокосмического университета им. академика М.Ф. Решетнева, 2015. Т. 16. № 1. С. 16-21.
76. Hammer P.L., Kogan A., Simeone B., Szedmak S. Pareto-optimal patterns in logical analysis of data. // Discrete Appl. Math. 144, 2004. P. 79–102.
77. Alexe G., Hammer P.L. Spanned patterns for the logical analysis of data. // Discrete Appl. Math. 154(7), 2006. P. 1039–1049.
78. Alexe S., Hammer,P.L. Accelerated algorithm for pattern detection in logical analysis of data. // Discrete Appl. Math. 154(7), 2006. P. 1050–1063.
79. Bonates T.O., Hammer P.L., Kogan A. Maximum patterns in datasets. // Discrete Appl. Math. doi:10.1016/j.dam.2007.06.004.

80. Eckstein J., Hammer P.L., Liu Y., Nediak M., Simeone B. The maximum box problem and its application to data analysis. // Comput. Optim Appl. 23(3), 2002. P. 285–298.
81. Antamoshkin A.N., Masich I.S. Combinatorial optimization and rule search in logical algorithms of machine learning. // Engineering & automation problems (Проблемы машиностроения и автоматизации), V.7, N.1, 2010. P. 52-57.
82. Антамошкин А.Н. Оптимизация функционалов с булевыми переменными. Томск: Изд-во Томского ун-та, 1987. 218 с.
83. Масич И.С. Поисковые алгоритмы условной псевдобулевой оптимизации: монография. Красноярск: СибГАУ, 2013. 160 с.
84. Антамошкин А.Н., Масич И.С. Исследование свойств задач оптимизации при поиске логических закономерностей в данных. // Системы управления и информационные технологии, 2011. N4.1(46). С. 111-115.
85. Масич И.С. Метод оптимальных логических решающих правил для задач распознавания и прогнозирования. // Системы управления и информационные технологии. 2019. Т. 75. № 1. С. 31-37.
86. Alexe G., Alexe S., Hammer P.L., Kogan A. Comprehensive vs. comprehensible classifiers in logical analysis of data. // Discrete Applied Mathematics, 2007.
87. Banfield R.E., Hall L.O., Bowyer K.W., Kegelmeyer W.P. A comparison of decision tree ensemble creation techniques. // IEEE Transactions on Pattern Analysis and Machine Intelligence, 29, 2007. P. 173-180.
88. Kuncheva L.I., Diversity in multiple classifier systems. // Information Fusion, 6, 2005. P. 3-4.
89. Alexe S., Hammer P.L. Pattern-based discriminants in the logical analysis of data. // Data Mining in Biomedicine. In: Pardalos P.M., Boginski V.L., Vazacopoulos A. (eds.). Springer, 2007.
90. Dua D., Graff C. UCI Machine Learning Repository [http://archive.ics.uci.edu/ml]. Irvine, CA: University of California, School of Information and Computer Science, 2019.

91. Lim T.S., Loh W.Y., Shin Y.S. A comparison of prediction accuracy, complexity, and training time of thirty-three old and new classification algorithms. // Mach. Learn. 40, 2000. P. 203-229.
92. Hammer P.L., Bonates T.O. Logical analysis of data - An overview: From combinatorial optimization to medical applications. // Annals of Operations Research 148, 2006.
93. Кузьмич Р.И., Масич И.С., Ступина А.А. Модели формирования закономерностей в методе логического анализа данных. // Системы управления и информационные технологии. 2017. Т. 67. № 1. С. 33-37.
94. Антамошкин А.Н., Масич И.С. Выбор логических закономерностей для построения решающего правила распознавания. // Вестник Сибирского государственного аэрокосмического университета им. академика М.Ф. Решетнева. 2014. № 5 (57). С. 20-25.
95. Осложнения инфаркта миокарда: база данных для апробации систем распознавания и прогноза / С.Е. Головенкин, А.Н. Горбань, В.А. Шульман и др. // Красноярск: Вычислительный центр СО РАН: Препринт №6, 1997.
96. Waikato Environment for Knowledge Analysis. Weka 3: Data Mining Software in Java. http://www.cs.waikato.ac.nz/ml/weka.
97. Масич И.С., Краева Е.М. Отбор закономерностей для построения решающего правила в логических алгоритмах распознавания. // Системы управления и информационные технологии, 2013. Т. 51. № 1.1. С. 170-173.
98. Crama Y., Hammer P.L. Boolean Functions. Theory, Algorithms, and Applications. New York: Cambridge University Press, 2011. 687 p.
99. Казаковцев Л.А. Подходы к автоматизации задач планирования ассортимента на торговых предприятиях // Вестник НИИ Систем управления, волновых процессов и технологий. 2000. Вып. 5.
100. Имануилов П.А., Пуртиков В.А. Решение задачи формирования кредитного портфеля банка методом Мивер. // Вестник НИИ Систем управления, волновых процессов и технологий. 2000. Вып. 5.

101. Ashrafi N., Berman O., Cutler M. Optimal Design of Large Software Systems Using N-Version Programming // In IEEE Transactions on Reliability. Vol. 43. №. 2, 1994. P. 344-350.
102. Avizienis A. The methodology of N-version programming // Software fault tolerance. Vol. 5, 1995. P. 23-47.
103. Антамошкин А.Н. Оптимизация функционалов с булевыми переменными. Томск: Изд-во Томского ун-та, 1987. 218 с.
104. Масич И.С. Метод ветвей и границ в задаче оптимизации систем отказоустойчивого программного обеспечения // Научная сессия ТУСУР. Материалы докладов межрегиональной научно-технической конференции. Томск, 2002. С. 31-34.
105. Masich I.S. Multi-version methods of software reliability growth in intelligence systems. Intelligent Systems // Proceeding of the Fifth International Symposium. Moscow: BMSTU. 2002. P. 74-76.
106. Системный анализ: проектирование, оптимизация и приложения / под общей редакцией А. Н. Антамошкина. Красноярск: САА и СО РИА, 1996. Т.1. 334 с.; Т.2. 290 с.
107. Юдин Д.Б., Горяшко А. П., Немировский А.С. Математические методы оптимизации устройств и алгоритмов АСУ. М.: Радио и связь. 1982. 288 с.
108. Barahona F., Grotschel M., Junger M., Reinelt G. An application of combinatorial optimization to statistical physics and circuit layout design // Operation Research. 1988. №. 36. P. 493-513.
109. System Analysis, Design and Optimization. An Introduction / General Editing by A. Žilinskas. Krasnoyarsk, 1993. 203 p.
110. Масич И.С. Поисковые алгоритмы псевдобулевой оптимизации в задачах диагностики и прогнозирования // Высокие технологии, исследования, промышленность. Сборник трудов 9 Международной научно-практической конференции «Исследование, разработка и применение высоких технологий в промышленности». Том 3. Санкт-Петербург, 2010. С. 89-90.

111. Karp R.M., Miller R.G., Thatcher J.W. Reducibility among combinatorial problems. // Journal of Symbolic Logic. 1975. № 40 (4). P. 618-619.
112. Ranyard R.H. An algorithm for maximum likelihood ranking and Slater's i from paired comparisions // British Journal of Mathematical and Statistical Psychology. 1976. № 29. P. 242-248.
113. Rao M.R. Cluster analysis and mathematical programming // Journal of the American Statistical Association. 1971. Vol. 66. P. 622-626.
114. Масич И.С. Поисковые алгоритмы комбинаторной оптимизации в задачах классификации данных // Высокие технологии, образование, промышленность. Т. 3: сборник статей XI Международной научно-практической конференции «Фундаментальные и прикладные исследования, разработка и применение высоких технологий в промышленности». Санкт-Петербург. 2011. С. 91-92.
115. Hammer P.L., Shliffer E. Applications of pseudo-Boolean methods to economic problems // Theory and decision. 1971. № 1. P. 296-308.
116. Масич И.С., Краева Е.М. Логические алгоритмы классификации в задачах управления сельскохозяйственной деятельностью // Инновационные тенденции развития российской науки: материалы VI международной научно-практической конференции. Красноярск, 2013. С. 123-125.
117. Hillier F.S. The evaluation of risky interrelated investments. Amsterdam: North-Holland Publishing, 1969. 113 p.
118. Laughhunn D.J. Quadratic binary programming with applications to capital budgeting problems // Operations Research, 1970. № 18. P. 454-461.
119. Laughhunn D.J., Peterson D.E. Computational experience with capital expenditure programming models under risk // Business Finance, 1971. № 3. P. 43-48.
120. Масич И.С. Комбинаторные алгоритмы оптимизации в задаче распределения земли при производстве полевых культур // Инновационные тенденции развития российской науки: материалы VI международной научно-практической конференции. Красноярск, 2013. С. 125-126.

121. Масич И.С., Краева Е.М. Логический анализ данных и алгоритмы распознавания в задачах управления сельскохозяйственной деятельностью // Наука и образование: опыт, проблемы, перспективы развития: материалы международной научно-практической конференции. Красноярск, 2013. С. 301-303.
122. Weingartner H.M. Capital budgeting of interrelated projects: survey and synthesis // Management Science. 1966. № 12. P. 485-516.
123. Масич И.С. Модель и алгоритмы оптимизации для задачи распределения земли при производстве полевых культур // Наука и образование: опыт, проблемы, перспективы развития: материалы международной научно-практической конференции. Красноярск, 2013. С. 303-304.
124. Hammer P.L. Pseudo-Boolean remarks on balanced graphs // International Series of Numerical Mathematics, 1977. № 36. P. 69-78.
125. Ebenegger Ch., Hammer P.L., de Werra D. Pseudo-Boolean functions and stability of graphs // Annals of Discrete Mathematics, 1984. № 19. P. 83-97.
126. Масич И.С. Алгоритмы комбинаторной оптимизации для планирования загрузки литейного производства // Материалы VII Международной научно-практической конференции «Татищевские чтения: актуальные проблемы науки и практики». Тольятти, 2010. С. 3-10.
127. Crama Y. Combinatorial optimization models for production scheduling in automated manufacturing systems // European Journal of Operational Research, 1997. № 99. P. 136-153.
128. Антамошкин А. Н., Дегтерев Д.А., Масич И.С. Регулярный алгоритм оптимизации загрузки оборудования // Труды конф. «Информационные недра Кузбасса». Кемерово: КемГУ, 2003. С. 59-61.
129. Bilbao J.M. Cooperative Games on Combinatorial Structures. Norwell, Massachusetts: Kluwer Academic Publishers, 2000.
130. Hammer P.L., Holzman R. Approximations of pseudo-Boolean functions: Applications to game theory // Methods and Models of Operations Research, 1992. №. 36. P. 3-21.

131. Crama Y., Hammer P.L. Bimatroidal independence systems // Mathematical Methods of Operations Research, 1989. Vol. 33. № 3. P. 149-165.
132. Welsh D.J.A. Matroid theory. London: Academic Press, 1976.
133. Fraenkel A.S., Hammer P.L. Pseudo-Boolean functions and their graphs // Annals of Discrete Mathematics,1984. № 20. P. 137-146.
134. Hammer P.L., Hansen P., Simeone B. Upper planes of quadratic 0-1 functions and stability in graphs // Nonlinear Programming. № 4, 1981. P. 395-414.
135. Nieminen J. A linear pseudo-Boolean viewpoint on matching and other central concepts in graph theory // Zastosowania Matematyki, 1974. № 14. P. 365-369.
136. Масич И.С. Отбор существенных факторов в виде задачи псевдобулевой оптимизации // Решетневские чтения: Тез. докл. VII Всерос. науч. конф. Красноярск: СибГАУ, 2003. С. 234-235.
137. Масич И.С. Локальная оптимизация для задачи выбора информативных факторов // Наука. Технологии. Инновации. Материалы докладов всероссийской научной конференции. Часть 1. Новосибирск: Изд-во НГТУ, 2003. С. 50-51.
138. Масич И.С. Повышение качества распознавания логических алгоритмов классификации путем отбора информативных закономерностей // Информационные системы и технологии: Материалы Международной научно-технической конференции. Красноярск: Изд. Научно-инновационный центр, 2012. С. 149-153.
139. Масич И.С. Задачи оптимизации и их свойства в логических алгоритмах распознавания // Проблемы оптимизации и экономические приложения: материалы V Всероссийской конференции. Омск: Изд-во Ом. гос. ун-та, 2012.
140. Масич И.С. Комбинаторная оптимизация и логические алгоритмы классификации в задаче прогнозирования осложнений инфаркта миокарда // Инновационные тенденции развития российской науки: материалы III Международной научно-практической конференции. Красноярск, 2010. С. 276-280.

141. Антамошкин А.Н. Регулярная оптимизация псевдобулевых функций. Красноярск: Изд-во Красноярского ун-та, 1989. 284 с.
142. Hammer P.L., Rudeanu S. Boolean Methods in Operations Research and Related Areas. Berlin: Springer-Verlag; New York: Heidelberg, 1968. 310 p.
143. Boros E., Hammer P.L. Pseudo-Boolean Optimization // Discrete Applied Mathematics, 2002. №. 123(1-3). P. 155-225.
144. Береснев В.Л., Агеев А.А. Алгоритмы минимизации для некоторых классов полиномов от булевых переменных // Модели и методы оптимизации: сб. науч. тр. Том 10. –Новосибирск: Наука, 1988. С. 5-17.
145. Гришухин В.П. Полиномиальность в простейшей задаче размещения: препринт. М.: ЦЭМИ АН СССР, 1987. 64 с.
146. Ковалев М.М. Дискретная оптимизация (целочисленное программирование). Минск: Изд-во БГУ, 1977. 191 с.
147. Корбут А.А., Финкельштейн Ю.Ю. Дискретное программирование. М.: Наука. Гл. ред. физ.-мат. Лит, 1969. 389 с.
148. Allemand K., Fukuda K., Liebling T.M., Steiner E. A polynomial case of unconstrained zero-one quadratic optimization // Mathematical Programming. Ser. A, 2001. № 91. P. 49-52.
149. Barth P. A Davis-Putnam Based Enumeration Algorithm for Linear Pseudo-Boolean Optimization // Technical Report MPI-I-95-2-003. Saarbrücken: Max-Planck-Institut für Informatik, 1995. 19 p.
150. Crama Y. Recognition problems for special classes of polynomials in 0-1 variables // Mathematical Programming, 1989. № 44. P. 139-155.
151. Crama Y., Hansen P., Jaumard B. The basic algorithm for pseudo-Boolean programming revisited // Discrete Applied Mathematics, 1990. № 29. P. 171-185.
152. Werra D. De, Hammer P.L., Liebling T., Simeone B. From linear separability to unimodality: A hierarchy of pseudo-Boolean functions // SIAM Journal on Discrete Mathematics, 1998. № 1. P. 174-184.

153. Hammer P.L., Hansen P., Pardalos P.M., Rader D.J. Maximizing the product of two linear functions in 0-1 variables // Research Report 2-1997. Piscataway, NJ: Rutgers University Center for Operations Research, 1997.
154. Hammer P.L., Peled U.N. On the Maximization of a Pseudo-Boolean Function // Journal of the Association for Computing Machinery, 1972. Vol. 19, № 2. P. 265-282.
155. Hansen P., Jaumard B., Mathon V. Constrained nonlinear 0-1 programming // Journal on Computing, 1993. № 5. P. 97-119.
156. Martello S., Toth P. The 0-1 Knapsack Problem. // Combinatorial Optimization, 1979. P. 237-279.
157. Padberg M.W. A Note on zero-one programming // Operations Research, 1975. № 23. P. 833-837.
158. Padberg M.W. Covering, Packing and Knapsack Problems // Annals of Discrete Mathematics, 1979. № 4. P. 265-287.
159. Wegener I., Witt C. On the Optimization of Monotone Polynomials by Simple Randomized Search Heuristics // Procs. of GECCO 2003, 2003. P. 622-633.
160. Adams W.P., Sherali H.D. A tight linearization and an algorithm for zero-one quadratic programming problems // Management Science, 1986. Vol. 32. № 10. P. 1274-1290.
161. Boros E., Hammer P.L. A max-flow approach to improved roof duality in quadratic 0-1 minimization // Research Report RRR 15-1989, Piscataway, NJ: Rutgers University Center for Operations Research, 1989.
162. Glover F., Woolsey R.E. Further reduction of zero-one polynomial programs to zero-one linear programming problems // Operations Research, 1973. № 21. P. 156-161.
163. Hansen P., Lu S.H., Simeone B. On the equivalence of paved duality and standard linearization in nonlinear 0-1 optimization // Discrete Applied Mathematics, 1990. № 29. P. 187-193.
164. Watters L.G. Reduction of integer polynomial problems to zero-one linear programming problems // Operations Research, 1967. № 15. P. 1171-1174.

165. Boros E., Crama Y., Hammer P. L. Chvátal cuts and odd cycle inequalities in quadratic 0-1 optimization // SIAM Journal on Discrete Mathematics, 1992. № 5. P. 163-177.
166. Boros E., Crama Y., Hammer P.L. Upper bounds for quadratic 0-1 maximization // Operation Research Letters, 1990. № 9. P. 73-79.
167. Boros E., Hammer P.L. A max-flow approach to improved roof duality in quadratic 0-1 minimization // Research Report RRR 15-1989, Piscataway, NJ: Rutgers University Center for Operations Research, 1989.
168. Boros E., Hammer P.L. Cut-Polytopes, Boolean quadratic polytopes and nonnegative quadratic pseudo-Boolean functions // Mathematics of Operations Research, 1993. № 18. P. 245-253.
169. Hammer P.L., Hansen P., Simeone B. Roof duality, complementation and persistency in quadratic 0-1 optimization // Mathematical Programming, 1984. № 28. P.121-155.
170. Hammer P.L., Simeone B. Quadratic functions of binary variables // Combinatorial Optimization: Lecture Notes in Mathematics. Vol. 1403. Berlin: Springer-Verlag; New York: Heidelberg, 1989.
171. Papadimitriou C.H., Steiglitz K. Combinatorial Optimization. New York: Dover Publications, 1998.
172. Масич И.С. Поисковые алгоритмы условной псевдобулевой оптимизации: монография. Красноярск: СибГАУ, 2013. 160 с.
173. Гринченко С.Н. Метод «проб и ошибок» и поисковая оптимизация: анализ, классификация, трактовка понятия «естественный отбор» // Исследовано в России, 2003. № 104. С. 1228-1271.
174. Растригин Л.А. Адаптация сложных систем. Рига: Зинатне, 1981. 375 с.
175. Гринченко С.Н. Иерархическая оптимизация в природных и социальных системах: селекция вариантов приспособительного поведения и эволюции систем "достаточно высокой сложности" на основе адаптивных алгоритмов случайного поиска // Исследовано в России, 2000. № 108. С. 1421-1440.

176. Гринченко С.Н. Случайный поиск, адаптация и эволюция: от моделей биосистем - к языку представления о мире (части 1 и 2) // Исследовано в России, 1999. № 10 и № 11.
177. Goldberg D.E. Genetic algorithms in search, optimization, and machine learning. Boston: Addison-Wesley, 1989.
178. Schwefel H.P. Evolution and Optimum Seeking. No.Y.: Whiley, 1995. 612p.
179. Björklund H., Petersson V., Vorobjov S. Experiments with Iterative Improvement Algorithms on Completely Unimodal Hypercubes // Technical report 2001-017. Uppsala, Sweden: Uppsala University, 2001. 30 p.
180. Björklund H., Sandberg S., Vorobjov S. Optimization on Completely Unimodal Hypercubes // Technical report 2002-018. Uppsala, Sweden: Uppsala University, 2002. 39 p.
181. Крутиков В.Н., Самойленко Н.С., Мешечкин В.В. О свойствах метода минимизации выпуклых функций, релаксационного по расстоянию до экстремума. // Автоматика и телемеханика. 2019. № 1. С. 126-137.
182. Крутиков В.Н. Численные методы безусловной оптимизации с итеративным обучением и их применение Диссертация на соискание ученой степени доктора технических наук / Томский государственный университет. Кемерово, 2005
183. Антамошкин А.Н. Регулярная оптимизация псевдобулевых функций. Красноярск: Изд-во Красноярского ун-та, 1989. 284 с.
184. Антамошкин А.Н., Рассохин В.С. Оптимизация полимодальных псевдобулевых функций // М.: ОФАП Минвуза РСФСР. Инв. № 85119, 1985. 43 с.
185. Antamoshkin A. Regular optimization of pseudoboolean functions // System Modelling and Optimization. Budapest, 1985. P. 17-18.
186. Масич И.С. Регулярный алгоритм поиска граничных точек в задаче условной оптимизации монотонных псевдобулевых функций // «Совершенствование технологий поиска и разведки, добычи и переработки

полезных ископаемых»: Сб. материалов Всероссийской научно-технической конференции. Красноярск: КрасГАЦиЗ, 2003. С. 170-172.

187. Масич И.С. Решение задач условной псевдобулевой оптимизации посредством обобщенных штрафных функций // Вестник молодых ученых, 2004. № 4. С. 63-69.
188. Масич И.С. Приближенные алгоритмы поиска граничных точек для задачи условной псевдобулевой оптимизации // Вестник СибГАУ, 2006. № 1(8). С. 39-43.
189. Garey M.R., Johnson D.S. Computers and Intractability: a Guide to the Theory of NP-completeness. San Francisco: W. H. Freeman and Company,1979.
190. Kreinovich V., Lakeyev A., Rohn J., Kahl P. Computational Complexity and Feasibility of Data Processing and Interval Computations // Norwell, Massachusetts: Kluwer Academic Publishers, 1998. 460 p.
191. Lenstra J.K., Rinnooy Kan A.H.G. Computational Complexity of Discrete Optimization Problems // Annals of Discrete Mathematics, 1979. № 4. P.1 21-140.
192. Генс Г.В., Левнер Е.В. Дискретные оптимизационные задачи и эффективные приближенные алгоритмы (обзор) // Известия АН СССР. Техническая кибернетика, 1976. № 6. С. 84-92.
193. Сергиенко И.В., Лебедева Т.Т., Рощин В.А. Приближенные методы решения дискретных задач оптимизации. Киев: Наукова думка, 1981. 272 с.
194. Финкельштейн Ю.Ю. Приближенные методы и прикладные задачи дискретного программирования. М.: Наука, 1976. 264 с.
195. Хачатуров В.Р. Аппроксимационно-комбинаторный метод и некоторые его приложения // Журнал вычислительной математики и математической физики, 1974. Т. 14. № 6. С. 1464-1487.
196. Лбов Г.С. Выбор эффективной системы зависимых признаков. // В кн.: Вычислительные системы. Новосибирск: ИМ СО АН СССР, 1965. Вып. 19. С. 21-34.

197. Лбов Г.С. Программа "Случайный поиск с адаптацией для выбора эффективной системы зависимых признаков" // В кн.: Геология и математика. Новосибирск: Наука, 1970. С. 204-219.
198. Foldes S., Hammer P.L. Disjunctive and conjunctive normal forms of pseudo-Boolean functions // Research Report RRR 1-2000. Piscataway, NJ: Rutgers University Center for Operations Research, 2000.
199. Foldes S., Hammer P.L., Monotone. Horn and Quadratic Pseudo-Boolean Functions // Journal of Universal Computer Science, 2000. № 6(1). P. 97-104.
200. Antamoshkin A.N., Masich I.S. Identification of pseudo-Boolean function properties. // Engineering & automation problems, 2007. № 2. P. 66-69.
201. Семенкин Е.С., Семенкина О.Э., Коробейников С.П. Оптимизация технических систем. Красноярск: СИБУП, 1996. 358 с.
202. Boros E., Hammer P.L. Pseudo-Boolean Optimization // Discrete Applied Mathematics, 2002. № 123(1-3). P. 155-225.
203. Iwata S., Fleischer L., Fujishige S. A combinatorial, strongly polynomial-time algorithm for minimizing submodular functions // Proceedings of the 32nd ACM Symposium on Theory of Computing. 2000. P.97-106.
204. Simeone B. Quadratic 0-1 programming. Boolean functions and graphs // Ph.D. thesis. Ontario, Canada: University of Waterloo, 1979.
205. Antamoshkin A., Masich I. Pseudo-Boolean Optimization in Case of an Unconnected Feasible Set. // In: «Models and Algorithms for Global Optimization». Springer Optimization and Its Applications, Vol. 4, 2007. P. 111-122.
206. Антамошкин А.Н., Масич И.С. Поисковые алгоритмы условной псевдобулевой оптимизации. // Системы управления, связи и безопасности, 2016. № 1. С. 103-145.
207. Antamoshkin A.N., Masich I.S. Heuristic search algorithms for monotone pseudo-boolean function conditional optimization. // Engineering & automation problems (Проблемы машиностроения и автоматизации), 2007. № 3. С. 41-45.

208. Масич И.С. Приближенные алгоритмы поиска граничных точек для задачи условной псевдобулевой оптимизации. // Вестник СибГАУ, 8470, 1(8), Красноярск, 2006. С. 39-43.
209. Feo T., Resende M.G. Greedy Randomized Adaptive Search Procedures. // Journal of Global Optimization, 1995. № 6. P. 109-133.
210. Масич И.С., Крушенко Г.Г. Алгоритмы случайного поиска в практических задачах условной оптимизации: препринт № 1-12. Красноярск: ИВМ СО РАН; СибГАУ, 2012. 38 с.
211. Masich I.S. Combinatorial optimization in foundry production planning. // Вестник СибГАУ. № 2(23), 2009. С.40-44.
212. Масич И.С., Шарыпова К.В. Оптимизация загрузки производственных мощностей литейного производства. // Системы управления и информационные технологии, №3(29), 2007. С. 76-80.
213. Береснев В.Л., Мельников А.А. Алгоритм ветвей и границ для задачи конкурентного размещения предприятий с предписанным выбором поставщиков. // Дискретн. анализ и исслед. опер., № **21**:2, 2014. P. 3-23.
214. Land A.H., Doig A.G. An automatic method of solving discrete programming problems // Econometrica, 1960. Vol. 28. P. 497-520.
215. An algorithm for the traveling salesman problem / J.D.C. Little, K.G. Murty, D.W. Sweeney and C. Karel. // Operations Research, 1963. Vol. 11. P. 972-989.
216. Сигал И.Х., Иванова А.П. Введение в прикладное дискретное программирование: модели и вычислительные алгоритмы. М.: ФИЗМАТЛИТ, 2002. 240 с.
217. Garey M.R., Johnson D.S. Computers and Intractability: a Guide to the Theory of NP-completeness. San Francisco : W.H. Freeman and Company, 1979.
218. Лесков Д.О. Тестирование алгоритма решения задач рюкзачного типа на основе метода ветвей и границ с релаксацией // Вестник НИИ СУВПТ: Адаптивные системы моделирования и адаптации: Сб. научн. трудов. Красноярск: НИИ СУВПТ, 2000. № 5. С. 59-61.

219. Стоян Ю.Г. Об одном отображении комбинаторных множеств в евклидово пространство: препринт. Харьков: ИПМаш АН УССР, 1982. 33 с.
220. Kazakovtsev L.A., Masich I.S. A branch-and-bound algorithm for a pseudo-boolean optimization problem with black-box functions. // Facta Universitatis, Series Mathematics and Informatics. Vol. 33, № 2 (2018). P. 337–360.
221. Береснев В. Л., Гончаров Е. Н., Мельников А. А. Локальный поиск по обобщённой окрестности для задачи оптимизации псевдобулевых функций // Дискретн. анализ и исслед. опер., 18:4, 2011. С. 3–16.
222. Antamoshkin A.N., Masich I.S. Combinatorial optimization in foundry practice. // IOP Conference Series: Materials Science and Engineering, 2016. С. 012001.
223. Masich I.S., Kazakovtsev L.A., Stupina A.A. Optimization Models for Detection of Patterns in Data. // Optimization Problems and their Applications. Proceedings of the School-Seminar on Optimization Problems and their Applications (OPTA-SCL 2018). Omsk. Russia. July 8-14, 2018. P. 264-275.
224. Казаковцев Л. А. Метод жадных эвристик для систем автоматической группировки объектов: диссертация ... доктора технических наук: 05.13.01. Сибирский федеральный университет.- Красноярск, 2016.
225. Орлов В.И., Федосов В.В. Качество электронной компонентной базы – залог длительной работоспособности космических аппаратов. // Решетневские чтения, 2013. Т. 1, № 17. С. 238-241.
226. Федосов В.В., Патраев В.Е. Повышение надежности радиоэлектронной аппаратуры космических аппаратов при применении электрорадиоизделий, прошедших дополнительные отбраковочные испытания в специализированных испытательных технических центрах. // Авиакосмическое приборостроение, 2006. № 10. С.50-55.
227. Патраев В.Е. Методы обеспечения и оценки надежности космических аппаратов с длительным сроком активного существования: монография. Красноярск: Издательство СибГАУ, 2010. 136 с.
228. Казаковцев Л.А., Масич И.С., Орлов В.И., Проценко В.В., Федосов В.В. Разработка алгоритмического обеспечения анализа однородности партий

электрорадиоизделий для комплектации РЭА КА : монография. Красноярск: Сиб. гос. аэрокосмич. ун-т., 2016. 192 с.

229. Казаковцев Л.А., Масич И.С., Орлов В.И., Федосов В.В. Быстрый детерминированный алгоритм для классификации электронной компонентной базы по критерию равнонадежности. // Системы управления и информационные технологии, 2015. Т. 62. № 4. С. 39-44.
230. Беляева Т.П. Достаточность и реализуемость требований к электронной компонентной базе // Моделирование систем и процессов, 2010. № 3-4. С. 10-12.
231. Федосов В.В., Орлов В.И. Минимально необходимый объем испытаний изделий микроэлектроники на этапе входного контроля // Изв. ВУЗов. Приборостроение, 2011. № 54 (4). С.68-62.
232. Казаковцев Л.А., Орлов В.И., Ступина А.А., Масич И.С. Задача классификации электронной компонентной базы. // Вестник Сибирского государственного аэрокосмического университета им. академика М.Ф. Решетнева, 2014. № 4 (56). С. 55-61.
233. Hamiter L. The History of Space Quality EEE Parts in the United States. // ESA Electronic Components Conference. Noordwijk, The Netherlands: ESTEC, 1990, Nov 12-16. P. 503-508.
234. Kirkconnell C.S., Luong T.T., Shaw L.S. [et al.]. High Efficiency Digital Cooler Electronics for Aerospace Applications // Infrared Technology and Applications, June 24, 2014. doi:10.1117/12.2053075
235. MIL-PRF-38535 Performance Specification: Integrated Circuits (Micricircuit) Manufacturing, General Specifications for. United States of America: Department of Defence, 2007. 188 p.
236. Scott A.J., Angel D.P. The US Semiconductor Industry: a Locational Analysis. Environment and Planning, series A, 1987. Vol.19(7). P. 875-912.
237. Данилин Н.С. Диагностика и контроль качества изделий цифровой микроэлектроники. М.: Из-во стандартов, 1991. 176 с.

238. Никифоров А.Ю., Скоробогатов П.К., Стриханов М.Н. [и др.]. Развитие базовой технологии прогнозирования, оценки и контроля радиационной стойкости изделий микроэлектроники // Известия высших учебных заведений. Электроника, 2012. № 5 (97). С. 18-23.

239. Орлов В.И., Федосов В.В. Фирменный стиль: надежность и качество // Петербургский журнал электроники. № 1(62), 2010. С. 55-62.

240. Прогнозирование надежности узлов и блоков радиотехнических устройств космического на основе моделирования напряженно-деформируемых состояний. / С. Б. Сунцов [и др.]. Томск: Издательство назначения Томского государственного университета систем управления и радиоэлектроники, 2012. 114 с.

241. Kwon Y., Omitaomu O.A., Wang G.-N. Data mining approaches for modeling complex electronic circuitdesign activities. //Computer & Industrial Engineering, 2008. Vol. 54. P. 229-241.

242. Identifying Systematic Failures on Semiconductor Wafers Using ADCAS. / Ooi M. P.- L. [et al.] // Design &Test, IEEE, 2013. Vol.30 (5). P. 44-53.

243. Субботин В., Стешенко В. Проблемы обеспечения бортовой космической аппаратуры космических аппаратов электронной компонентной базой // Компоненты и технологии, 2011. №11. С. 10-12.

244. Харченко В.С., Юрченко Ю.Б. Анализ структур отказоустойчивых бортовых комплексов при использовании компонент Industry // Технология и конструирование в электронной аппаратуре, 2003. № 2. С.3-10.

245. Осложнения инфаркта миокарда: база данных для апробации систем распознавания и прогноза: препринт № 6 / С. Е. Головенкин, А. Н. Горбань, В. А. Шульман и др. Красноярск: Вычислительный центр СО РАН, 1997.

246. Масич И.С. Модель логического анализа для прогнозирования осложнений инфаркта миокарда. // Информатика и системы управления, 2010, №3 (25). С. 48-56.

247. Головенкин С.Е., Гулакова Т.К., Кузьмич Р.И., Масич И.С., Шульман В.А. Модель логического анализа для решения задачи прогнозирования

осложнений инфаркта миокарда. // Вестник СибГАУ. № 4(30), 2010. С. 68-73.

248. Кузьмич Р.И., Масич И.С. Модификации метода логического анализа данных для задач классификации: монография. Красноярск: Сиб. федер. ун-т, 2018. 180 с.
249. Kuzmich R. et al. The Modified Method of Logical Analysis Used for Solving Classification Problems. // INFORMATICA, 2018, Vol. 29, № 3. P. 467–486.
250. Kuzmich R. I. et al. Application of informative patterns in the classifier for a logical data analysis method development. // IOP Conf. Series: materials Science and Engineering 450, 2018. 052005.

ПРИЛОЖЕНИЕ А. АКТЫ ОБ ИСПОЛЬЗОВАНИИ РЕЗУЛЬТАТОВ

Открытое акционерное общество
«Испытательный технический центр-НПО ПМ»
ул. Молодежная, 20, г. Железногорск, ЗАТО Железногорск, Красноярский край, Россия, 662970
Тел. / факс (391-9)-74-97-45, e-mail: itcnpopm@atomlink.ru, ttc@krasmail.ru, http:/www.ttc-npopm.ru
ОКПО 51431138, ОГРН 1032401224272, ИНН 2452027379, ОКВЭД 73.10

УТВЕРЖДАЮ
Директор ОАО «ИТЦ-НПО - ПМ»
В.И. Орлов

АКТ

о внедрении результатов диссертационного исследования
Масича Игоря Сергеевича.

Настоящим актом подтверждается, что система автоматизированного формирования и контроля специальных партий электрорадиоизделий космического применения, использующая алгоритмы классификации данных, в частности – логические алгоритмы анализа и классификации данных и поисковые алгоритмы условной псевдобулевой оптимизации, и созданный с их применением метод классификации электрорадиоизделий по производственным партиям с выявлением однородных групп на основе данных входного контроля, дополнительных отбраковочных и диагностических испытаний внедрена в деятельности ОАО «Испытательный технический центр – НПО ПМ».

Внедрение данной системы, разработанной в рамках диссертационного исследования доцента ФГБОУ ВПО «Сибирский государственный аэрокосмический университет имени академика М.Ф. Решетнева» Масича Игоря Сергеевича, позволило регламентировать и автоматизировать процесс отбора электрорадиоизделий для формирования специальных партий изделий, предназначенных для космического приборостроения, в результате чего появилась возможность реализации технологического процесса комплектования узлов космических аппаратов продукцией более высокого уровня качества, приближенного к продукции иностранного производства категории качества «SPACE».

Заместитель директора
ОАО «ИТЦ-НПО ПМ»
В.В. Федосов

15.10.2016.

ОАО «ИТЦ – НПО ПМ»

Открытое акционерное общество
"Испытательный технический центр – НПО ПМ"
Россия, Красноярский край, ЗАТО Железногорск, г. Железногорск,
ул. Молодежная, 20,
662970, т.(391-97) 4-97-40, факс: (391-97) 4-97-45, *E-mail:* ttc@krasmail.ru

УТВЕРЖДАЮ
Директор ОАО «ИТЦ – НПО ПМ»
В. И. Орлов

АКТ

о внедрении программы для ЭВМ

Настоящим актом подтверждается, что программа для ЭВМ «Модифицированный метод логического анализа данных» (свидетельство о государственной регистрации программы для ЭВМ № 2016619162 от 15.08.2016, авторы: Антамошкин Александр Николаевич, Кузьмич Роман Иванович, Масич Игорь Сергеевич) внедрена и успешно используется в деятельности ОАО «Испытательный технический центр – НПО ПМ».

Программа реализует алгоритмы классификации данных, в частности – логические алгоритмы анализа и классификации данных и поисковые алгоритмы условной псевдобулевой оптимизации, и предназначена для модификации метода классификации электрорадиоизделий по классам качества и производственным партиям на основе данных входного контроля и разрушающего физического анализа.

Заместитель директора
ОАО «ИТЦ – НПО ПМ»
29.09.2016.

В. В. Федосов