Федеральное государственное автономное образовательное
учреждение высшего образования
«СИБИРСКИЙ ФЕДЕРАЛЬНЫЙ УНИВЕРСИТЕТ»

На правах рукописи

Монгуш Чодураа Михайловна

РАЗРАБОТКА МЕТОДА И СРЕДСТВ ФРАГМЕНТАЦИИ И ДЕФРАГМЕНТАЦИИ ФОРМАЛЬНЫХ КОНТЕКСТОВ

Специальность 05.13.17 — Теоретические основы информатики

Диссертация на соискание ученой степени
кандидата физико-математических наук

Научный руководитель
кандидат физико-математических наук,
доцент Семенова Дарья Владиславовна

Красноярск 2019

ОГЛАВЛЕНИЕ

Введение

Актуальность темы исследования. Во многих задачах интеллектуального анализа данных, в том числе в анализе естественно-языковых текстов, изучаемая предметная область часто описывается в виде объектно-признаковой таблицы, в которой каждый столбец соответствует некоторому признаку, а каждая строка определяет признаковое описание отдельного объекта [1, 5, 6, 29–32, 39].

Появление размеченных корпусов естественных языков как элементов информационных систем позволяет получать структурированную информацию и представлять ее в виде объектно-признаковой таблицы. Разметка — главная характеристика корпуса, отличающая корпус от простых электронных коллекций текстов [33, 72]. Она отражает лингвистическую и экстралингвистическую информацию хранимых текстов в корпусе. Чем больше набор признаков, характеризующий каждый текст, тем шире возможности корпуса по поиску текстов для решения различных филологических и лингвистических задач. В рамках корпусов решаются различные прикладные задачи, учитывающие семантическую составляющую анализируемых текстов [23, 74]. Эти задачи в основном сводятся к задачам концептуального моделирования коллекции текстов [43].

Существует формализованный подход, известный в литературе как анализ формальных понятий (АФП, англ. Formal Concept Analysis), который позволяет построить концептуальную модель исходя из объектно-признаковой таблицы на основе алгебраической теории решеток Г. Биркгофа [9,10]. Построенная концептуальная модель позволяет решать различные прикладные задачи, связанные с анализом текстов на естественном языке. В рамках АФП объектно-признаковая таблица представляется формальным контекстом и моделируется 0,1-матрицей, отражающей отсутствие или наличие признаков, характерных для исследуемого множества текстов. Основные идеи АФП были сформулированы в работах Р. Вилле и Б. Гантера в начале 80-х годов XX века и развиты в исследованиях российских ученых С. О. Кузнецова, К. А. Найденовой, С. А. Объедкова, С. И. Гурова, Д. И. Игнатова [20, 34, 35, 91–93, 110]. Каждое формальное понятие в АФП определяется с использованием соответствий Галуа и представляет

собой пары замкнутых множеств, интерпретируемых как объем и содержание этого понятия. В матричной форме формальному понятию соответствует некоторая максимально полная подматрица 0,1-матрицы, представляющей формальный контекст. Главным достоинством этого определения является полное совпадение традиционной трактовке термина «понятие», применяемого в гуманитарных науках [125]. С применением методов АФП решаются типовые задачи анализа данных, связанные с классификацией и кластеризацией данных, выявлением зависимостей между данными [7, 8, 11, 21, 35, 46, 88, 113, 117]. В них формальные понятия трактуются как перекрестные ассоциации, кластеры или бикластеры. В рамках АФП решение указанных задач сводится к нахождению всех формальных понятий исходного формального контекста с последующим связыванием их в решетку. Полученная решетка служит концептуальной моделью исследуемой предметной области и основой для решения прикладных задач.

При всей привлекательности методов АФП их практическое применение ограничивается высокой трудоемкостью процесса извлечения всех формальных понятий из исходного контекста. В задаче нахождения всех формальных понятий требуется найти множество всех формальных понятий для заданного формального контекста. Данная задача относится к комбинаторным перечислительным задачам и является $\#P$-полной [103]. Высокая вычислительная сложность задачи состоит в том, что число формальных понятий в общем случае экспоненциально зависит от размера исходного формального контекста. Рассматриваемая задача эквивалентна задаче определения всех максимально полных подматриц 0,1-матрицы и может встречаться в различных задачах комбинаторной оптимизации [24–26, 79, 82, 97, 105, 109, 115, 118, 124, 127].

Степень разработанности темы исследования.

На сегодняшний день для нахождения множества всех формальных понятий и построения решетки разработано много алгоритмов и программных средств [45, 80, 84, 89, 94, 95, 99–101, 106, 107, 112]. Традиционно данные алгоритмы разделяют на две группы: пакетные алгоритмы (Bordat [84], NextClosure [94], Close-by-One [101], Lindig [106]), которые строят решетку понятий из ранее найденных формальных понятий; инкрементные алгоритмы (Nourine [112], Godin

[95], Dowling [89], Norris [99]), которые достраивают решетку посредством постепенного добавления объектов и пересечения с имеющимися формальными понятиями. Подробное описание и сравнение практической производительности этих алгоритмов представлено в работе [101]. Известно, что время выполнения указанных алгоритмов в худшем случае составляет $O(|FC| \cdot |G|^2 \cdot |M|)$, где $|FC|$ — число найденных формальных понятий, $|G|$ — количество объектов, $|M|$ — количество признаков исходного формального контекста. Поскольку величина $|FC|$ экспоненциально зависит от $|G|$ и $|M|$, то время выполнения данных алгоритмов также может быть экспоненциальным.

Наиболее известными программными системами являются Concept Explorer, ToscanaJ, Galicia, Lattice Minner, OpenFCA, FCART [27, 81, 86, 104, 111, 116, 123]. Многие из них находятся в открытом доступе. Программа Concept Explorer позволяет обрабатывать формальные контексты, шкалировать многозначные признаки формального контекста и визуализировать решетки формальных понятий [27]. Автоматическое формирование исходного контекста на основе реляционных баз данных реализовано в системе ToscanaJ [81]. Программа Galicia имеет широкие возможности по визуализации решеток формальных понятий в трехмерном пространстве, а основная цель OpenFCA — отображение решеток через веб-приложения [86,123]. Все эти программные средства являются специализированными продуктами, т. е. направлены на решение конкретной задачи. Создатели программы FCART объединили полный цикл исследований на основе методов АФП в одну универсальную интегрированную среду [111]. Однако, вычислительная сложность задачи нахождения всех формальных понятий формального контекста большой размерности и связывание их в решетку остается открытой.

В настоящее время актуальны исследования по снижению вычислительной сложности задачи нахождения всех формальных понятий. Первое направление исследований связано с разработкой новых алгоритмов отбора информативных, релевантных формальных понятий при построении решетки [25, 35, 76–78]. Такой подход к решению рассматриваемой задачи позволяет уменьшить ее выход. Второе направление исследований рассматривает уменьшение размерно-

сти входа, а значит, повышение производительности существующих алгоритмов нахождения множества всех формальных понятий и родственных с ней задач, путем «неискажающего» разложения формального контекста — декомпозиции исходного контекста с сохранением всех искомых формальных понятий [83,90,91,119,121]. Такое направление исследований является более универсальным и рассматривается в настоящей диссертационной работе.

Цель и задачи исследования. Целью диссертационной работы является повышение производительности существующих алгоритмов решения задачи нахождения всех формальных понятий путем декомпозиции формального контекста на фрагменты без потери искомых формальных понятий и разработка на их основе математического и программного обеспечения.

Для достижения цели были поставлены и решены следующие задачи.

1. Разработать и теоретически обосновать метод «неискажающего» разложения формального контекста на фрагменты. Исследовать структуру фрагментов и найти оценку числа фрагментов, получаемых на каждой итерации разложения, определить правила остановки процесса разложения формального контекста на фрагменты без потери формальных понятий.

2. Разработать алгоритмы формирования для заданного формального контекста системы фрагментов, восстановления искомого решения исходя из решений, полученных для подзадач, и реализации возможных запросов на извлечение знаний из решетки формальных понятий.

3. Разработать алгоритмы предобработки формального контекста без потери формальных понятий путем удаления единичных, нулевых и кратных строк и столбцов этого контекста.

4. Создать комплекс программ, реализующий разработанные метод и алгоритмы, для проверки их результативности на случайных формальных контекстах и на реальных данных применительно к корпусу тувинского языка.

Научная новизна.

1. Разработан новый метод декомпозиции формального контекста на фрагменты без потери формальных понятий. В отличие от существующих методов АФП, предложенный метод позволяет уменьшить размерность формального кон-

текста с сохранением всех искомых формальных понятий и тем самым повысить производительность известных алгоритмов нахождения всех формальных понятий формального контекста.

2. Впервые разработан алгоритм реализации предложенного метода «неискажающей» декомпозиции формального контекста. Алгоритм отличается от ранее существующих алгоритмов тем, что разлагает исходный формальный контекст без потери формальных понятий и восстанавливает решение поставленной задачи исходя из решений, полученных для подзадач.

Методы исследования. Для решения поставленных в диссертационной работе задач использовались современные методы АФП, теории графов и методы объектно-ориентированного программирования.

Теоретическая значимость работы. Предложенный в работе метод «неискажающей» декомпозиции формального контекста может быть использован для развития АФП и комбинаторной оптимизации при решении задач определения всех максимально полных подматриц 0,1-матрицы.

Практическая значимость работы. Применение результатов диссертационной работы при исследовании объектно-признаковых описаний предметных областей позволяет на семантическом уровне решать различные задачи анализа данных, включая классификацию, кластеризацию, обнаружение закономерностей в данных и извлечение знаний из решетки формальных понятий. Исследование естественно-языковых текстов тувинского фольклора в научно-образовательном центре «Тюркология» Тувинского государственного университета с применением предложенных в работе метода и алгоритмов позволяет эффективно решать филологические и лингвистические задачи в рамках корпуса тувинского языка.

Соответствие паспорту специальности. Диссертационная работа соответствует области исследования специальности 05.13.17 — Теоретические основы информатики по п. 5 «Разработка и исследование моделей и алгоритмов анализа данных, обнаружения закономерностей в данных и их извлечениях, разработка и исследование методов и алгоритмов анализа текста, устной речи и изображений» (пункты 1, 2 научной новизны).

Положения, выносимые на защиту.

1. Доказательство корректности метода декомпозиции формального контекста на фрагменты без потери формальных понятий, позволяющего уменьшить размерность формального контекста и тем самым повысить производительность известных алгоритмов нахождения всех формальных понятий формального контекста.

2. Алгоритм реализации предложенного метода «неискажающей» декомпозиции формального контекста и оценки его сложности, а также рекомендации по практическому применению этого алгоритма при анализе данных.

3. Комплекс программ для проверки результативности предложенных метода и алгоритмов на случайных формальных контекстах и на реальных данных применительно к корпусу тувинского языка.

Степень достоверности и апробация результатов работы. Достоверность результатов работы подтверждается строгими математическими доказательствами основных положений, экспериментальной проверкой результатов, численных расчетов на реальных текстовых данных и практической эффективности программных реализаций.

Основные результаты работы докладывались и обсуждались на III Международной научно-практической конференции молодых ученых, аспирантов и студентов «Актуальные проблемы исследования этноэкологических и этнокультурных традиций народов Саяно-Алтая» (Кызыл, 2015), Международной конференции студентов, аспирантов и молодых ученых «Молодежь и наука: Проспект Свободный — 2016» (Красноярск, 2016), Международной конференции «Актуальные проблемы прикладной математики и информационных технологий — Аль-Хорезми 2016» (Ташкент, 2016), Международной конференции имени А. Ф. Терпугова «Информационные технологии и математическое моделирование» (Томск, 2016, 2018), VI Международной конференции «Математика, ее приложения и математическое образование» (Улан-Удэ, 2017), Всероссийской научно-практической конференции преподавателей, сотрудников и аспирантов Тувинского государственного университета (Кызыл, 2017), Всероссийской конференции «Компьютерная безопасность и криптография» — SIBECRYPT’19

(Томск, 2019), VII Международной конференции «Знания — Онтологии — Теории» (Новосибирск, 2019), научных семинарах кафедры высшей и прикладной математики Сибирского федерального университета и кафедры информатики и ИКТ Тувинского государственного университета.

Результаты диссертационного исследования переданы в научно-образовательный центр «Тюркология» для использования в научных исследованиях и на кафедру информатики и ИКТ Тувинского государственного университета для внедрения в учебный процесс при подготовке бакалавров по направлению «Фундаментальная информатика и информационные технологии», а также успешно применены для выполнения гранта РФФИ (РГНФ) № 16-34-1-01033 в 2016–2017 гг. Получены свидетельства о государственной регистрации программ для ЭВМ № 2018618907, № 2018615490 от 23.07.2018.

Личное участие автора в получении результатов, изложенных в диссертации. Основные результаты, составляющие новизну диссертационной работы, получены лично автором. Обсуждение метода, алгоритмов, результатов численных экспериментов и подготовка публикаций осуществлялись совместно с научным руководителем и соавторами опубликованных работ.

Публикации. По результатам диссертационных исследований опубликовано 14 печатных работ, из них 4 — в журналах, рекомендованных ВАК (в том числе 3 статей в изданиях, индексируемых Web of Science и Scopus) [13, 14, 87, 108], 8 — в других изданиях [47, 48, 51, 53–55, 57, 60], получены 2 свидетельства о государственной регистрации программ для ЭВМ [58, 59].

Структура и объем диссертации. Диссертация состоит из введения, трех глав, заключения, списка литературы. Общий объем диссертации составляет 105 страниц; иллюстративный материал представлен 25 рисунками и 20 таблицами; список литературы содержит 127 наименований.

Во **введении** раскрывается актуальность темы диссертации, формулируются проблемы исследования, определяются цель и задачи работы, описываются методы исследования, излагаются основные научные результаты, обосновывается теоретическая и практическая значимость работы, дается общая характеристика исследования.

В **первой главе** рассматриваются современные модели представления естественно-языковых текстов и их коллекций, приводятся основные положения АФП, используемые в диссертационной работе. Формулируется задача нахождения всех формальных понятий заданного формального контекста. Данная задача анализируется с точки зрения ее вычислительной сложности, особо подчеркивается ее перечислительный комбинаторный характер, исследуются известные подходы решения задачи, приводятся родственные с ней задачи и существующие алгоритмы решения этих задач.

Во **второй главе** диссертационной работы содержатся основные результаты диссертационного исследования, связанные со снижением трудоемкости процесса нахождения множества всех формальных понятий и построения для них решетки за счет применения декомпозиционного подхода. Описывается метод «неискажающей» декомпозиции формального контекста, определяются фрагменты формального контекста и исследуются свойства этих фрагментов для установления правил эффективной организации процесса декомпозиции и восстановления искомого решения. Приводится описание алгоритмов формирования системы фрагментов FindBoxes, восстановления искомого решения на основе решений, полученных для подзадач LatticeContext и реализации запросов на извлечение знаний из решетки формальных понятий при решении практических задач Query1, Query2. Рассматриваются процедуры предобработки формального контекста без потери формальных понятий для снижения времени вычисления всех формальных понятий исходного контекста. Приводится анализ результативности разработанных метода, алгоритмов и процедур.

В **третьей главе** диссертационной работы описывается комплекс программ FCACorpus, в котором реализованы разработанные метод «неискажающей» декомпозиции формального контекста, алгоритм FindBoxes формирования системы фрагментов, алгоритм LatticeContext восстановления решетки формальных понятий, алгоритмы Query1, Query2 реализации запросов и процедуры предобработки формального контекста без потери формальных понятий. Рассматривается структура базы данных «Тувинские героические сказания» и описание специального модуля IInterface, обеспечивающего информационный интерфейс

между базой данных и комплексом программ FCACorpus. Приводится решение прикладной задачи по установлению авторского стиля сказителей тувинского героического эпоса с использованием комплекса программ FCACorpus.

В **заключении** диссертации изложены основные результаты и выводы, полученные на основе настоящей диссертационной работы.

Благодарности. Автор выражает искреннюю и глубокую благодарность д.ф.-м.н., профессору Быковой Валентине Владимировне за неоценимую помощь и поддержку на всех этапах выполнения работы.

Глава 1 Объектно-признаковые модели коллекций текстов и методы их анализа

Первая глава носит вводный характер. В 1.1 рассматриваются современные модели представления естественно-языковых текстов и их коллекций [1, 5, 12, 16, 18, 23, 32, 33, 36, 37, 40, 42, 43, 63, 74, 75]. В 1.2 приведены основные положения и типовые обозначения АФП. Далее в 1.3 формулируется в терминах АФП задача нахождения всех формальных понятий формального контекста, исследуются известные подходы решения данной задачи, приводятся родственные с ней задачи и существующие алгоритмы решения этих задач.

1.1 Модели и методы анализа естественно-языковых текстов

Анализом и моделированием естественных языков занимается компьютерная лингвистика (англ. *computational linguistics*) [1, 32]. В рамках этой научной области разрабатываются алгоритмы и программные средства для представления и обработки естественно-языковых текстов. К задачам и направлениям компьютерной лингвистики относят: обработку естественно-языковых текстов на том или ином уровне (синтаксическом, морфологическом или семантическом); автоматический перевод текстов, построение систем управления знаниями; автореферирование; извлечение фактов из текста.

Корпусная лингвистика — одно из направлений компьютерной лингвистики, целью которой является создание и использование корпусов естественных языков [33]. Под корпусом понимается информационно-лингвистическая система, сформированная на собрании оцифрованных текстов на некотором естественном языке [33]. Корпус включает в себя не только различные типы текстов на естественном языке, но и также их разметку — информацию о свойствах этих текстов. Главной характеристикой корпуса, отличающей корпус от простых электронных коллекций (или электронных библиотек) текстов, является разметка. Основной целью корпусной лингвистики является изучение рассматриваемого естественного языка. Поэтому ее ведущие направления научной деятельности: создание словарей; лексикографические исследования; получение информации о

лексическом составе языков и об относительных частотах употребления тех или иных слов; изменение в лексическом составе языков и различных их вариаций; изучение грамматики языка; обучение языку.

Таким образом, объектом исследования компьютерной и корпусной лингвистики являются тексты на естественном языке. Отличие состоит в том, что компьютерная лингвистика разрабатывает инструменты (модели, методы, алгоритмы и программные средства) для корпусной лингвистики, которые способны решать прикладные задачи корпуса.

Анализ отдельного текста может состоять в его разметке, в получении информации о некоторых часто встречающихся слов, их словосочетаниях и т. п. Интеллектуальный анализ текстовых документов (англ. *text mining*) занимается извлечением знаний из коллекции естественно-языковых текстов с применением методов машинного обучения [5]. Переход от изучения отдельного текста к анализу коллекций текстов расширяет ряд решаемых задач. В интеллектуальном анализе данных решаются такие группы задач как классификация, кластеризация, тематическое моделирование, информационный поиск, извлечение новых знаний [5, 12, 21, 35, 65, 69, 70]. Можно выделить два основных подхода к анализу естественно-языковых текстов. Первый подход основан на применении семантических моделей текста: семантические сети, концептуальные графы и онтологии [12, 43, 74, 120]. Второй подход — количественный, использующий вероятностные, алгебраические методы исследования отдельных текстов и коллекции текстов. В данном подходе моделью конкретного текста является некоторая его языковая модель, а моделями коллекции текстов — тематическая модель, объектно-признаковая таблица [23, 40, 41, 43, 69, 74, 75, 120].

Семантическая модель некоторой предметной области — группы однородных объектов, связанных между собой отношениями. В данном случае под однородностью объектов понимается как наличие у них одних и тех же признаков. Традиционно группы однородных объектов называют концептами, понятиями или сущностями. Объекты могут иметь различную природу. Если в качестве объектов выступают естественно-языковые тексты, то семантическая модель коллекции текстов задает перечень взаимосвязанных понятий предметной

области, описываемых этими текстами. Например, онтологии — семантические модели коллекций естественно-языковых текстов. Под онтологией традиционно понимают систему понятий некоторой предметной области, соединенных отношением «общее–частное» или «часть–целое» [23, 36, 40, 75]. Семантические модели могут быть также использованы при исследовании семантики отдельного текста [12, 74]. Здесь в качестве понятий или концептов выступают слова или словосочетания. Например, семантическая сеть отдельного текста может быть представлена в виде графа, вершинами которого являются понятия, а дуги — отношениям между ними. Концептуальный граф — это семантическая модель отдельного текста. Задается двудольным графом, состоящим из множества вершин, которое разбито на две части: вершины, отвечающие понятиям; вершины, соответствующие отношениям между понятиями. Понятия отображаются прямоугольниками, отношения между ними — эллипсами.

Языковые модели текстов, как правило, применяются при количественном подходе к анализу отдельных текстов и представляются цепями Маркова. В ней каждый узел означает одно слово, а ребра — вероятности того, что одно слово встретится после другого. Данная модель используются в тех случаях, когда важно отразить короткие связи между словами и словосочетаниями. Например, для решения задачи машинного перевода, распознавания речи, исправления опечаток и др. Основными видами языковых моделей являются модели униграмм (одиночных слов), модели биграмм (последовательных пар слов), n-грамм. Подробное описание перечисленных моделей приведено в работе [120].

Тематическая модель коллекции документов устанавливает к каким темам относится отдельный документ и какие слова (термины) образуют всякую тему. К таким моделям относятся модели латентно-семантического анализа или вероятностного латентного семантического анализа. Они описывают каждую тему дискретным распределением вероятностей слов, а каждый документ — дискретным распределением вероятностей тем. Тематические модели широко используются в задачах поиска по запросу, классификации, автоматического реферирования текстов, фильтрации спама [16, 18, 42, 85, 98].

В тематических моделях каждый текстовый документ можно описать некоторым набором признаков. Признаками могут выступать различные характеристики текста: лингвистические, статистические, структурные. Например, частота определенных слов или словосочетаний в документе, наличие некоторых синтаксических конструкций и др. Для описания отдельных текстов, а также их коллекций используются следующие модели: векторная модель, мешок-слов и объектно-признаковая таблица.

При использовании мешка-слов и векторной модели каждый отдельный текст представляется в виде точки в признаковом пространстве, а в качестве признаков чаще всего берутся частотные показатели слов. Мешок-слов представляет собой множество пар «слово — вес», где вес слов (или термов) определяется как:

– бинарный вес, наличие или отсутствие некоторых слов в документе;
– количество вхождений слов в документе;
– функция от количества вхождений терма в документе (TF term frequency);
– обратная частота документов (IDF inverse document frequency), т. е. вес вычисляется как произведение функции от количества вхождений слова в документ и функции от величины, обратной количеству документов коллекции, в которых встречается это слово.

При этом не учитываются порядок слов в тексте и морфологические формы представления слов. Когда порядок слов важен, употребляется термин «векторная модель» текста [120, 122].

Мешок-слов и векторная модель текста применяются не только в тематическом моделировании, но и при решении других задач интеллектуального анализа естественно-языковых текстов: поиск текстового документа, классификация или кластеризация документов. Для решения этих задач используются методы машинного обучения и математической статистики, основанные на количественных мерах близости рассматриваемых текстов [17].

Объектно-признаковая таблица — модель представления коллекции текстов, в которой каждый столбец соответствует некоторому признаку, а каждая строка определяет признаковое описание отдельного текста. Например, в корпусной лингвистике в качестве набора признаков может выступить паспорт (метоописа-

ние) текста. Под паспортом текста понимается экстралингвистические параметры текста: сведения об авторе; библиографические сведения; жанровые, стилевые и другие особенности текста [33]. Они характеризуют текст в целом. Эта информация, чаще всего, входит в разметку произведений, включенных в рассматриваемый корпус языка [72]. Чем больше набор признаков имеет паспорт текста, тем шире возможности поиска в корпусе для решения филологических и лингвистических задач. Как правило, информация, отражающая паспорт текста хорошо структурирована и допускает представление ее в виде объектно-признаковой таблицы.

В настоящее время для сохранения национального литературного наследия и проведения научных исследований по изучению языков народов Российской Федерации активно создаются электронные корпусы. Разработкой электронного корпуса текстов тувинского языка занимается студенты, аспиранты и сотрудники Тувинского государственного и Сибирского федерального университетов [2, 52, 62, 73]. На данный момент в корпусе тувинского языка содержатся тексты официально-деловых документов и тувинской художественной литературы. В корпус также входят тувинско-русский словарь «ТывЛин», частотный словарь по художественным произведениям на тувинском языке, словарь диалектных слов алтайского диалекта и морфемно-орфографический словарь тувинского языка, составленный М. В. Бавуу-Сюрюн и С. М. Далаа. Работы по углублению уровня обработки текстов и расширению информационного содержания корпуса тувинского языка продолжаются.

Важной составляющей этнокультурного наследия Республики Тыва являются исследования произведений тувинского героического эпоса [51]. В научном архиве Тувинского института гуманитарных и прикладных социально-экономических исследований Республики Тыва находится сформированный фонд рукописных и магнитофонных записей всех жанров тувинского фольклора, в том числе хранятся около 300 записей произведений тувинского героического эпоса. Однако свет увидели немногие. Поскольку данные произведения хранятся в рукописном виде или напечатаны в старых ветхих изданиях. В настоящее время имеются лишь 14 напечатанных собраний тувинских героических ска-

заний. На основе этих публикаций под руководством профессора Тувинского государственного университета М.В. Бавуу-Сюрюн создана электронная коллекция «Тувинские героические сказания» [3, 4]. В коллекции «Тувинские героические сказания» хранятся оцифрованные тексты более 50 произведений, а также их метаописания, включая сведения о сказителях, представленные в виде объектно-признаковой таблицы. С использованием коллекции «Тувинские героические сказания» можно решать филологические и лингвистические задачи: выявление индивидуального авторского стиля сказителей тувинского героического эпоса, особенности использования языковых клише и диалектных вариантов эпических выражений и др. Эти задачи сводятся к задаче концептуального моделирования коллекции «Тувинские героические сказания».

Существует формализованный подход, известный в литературе как анализ формальных понятий (АФП), который позволяет построить концептуальную модель, исходя из объектно-признаковой таблицы на основе алгебраической теории решеток. Построенная концептуальная модель позволяет решать различные лингвистические и филологические задачи, связанные с анализом естественно-языковых текстов. В рамках анализа формальных понятий объектно-признаковая таблица моделируется формальным контекстом, отражающим отсутствие или наличие признаков, присущих исследуемому множеству текстов. Анализ формальных понятий возник в начале 80-годов XX века с появлением работ Р. Вилле и Б. Гантера как прикладное направление теории решеток Г. Биркгофа [9, 10, 91–93]. В нем общепринятый термин «понятие» формализуется, определяется с использованием соответствий Галуа и задается парой множеств (объем, содержание), называемой формальным понятием. Такое определение полностью соответствует традиционной трактовке термина «понятие», применяемого в гуманитарных науках [125]. Использование методов АФП для исследования текстов на естественном языке ограничивается проблемой размерности объектно-признаковой таблицы, описывающей рассматриваемую коллекцию текстов.

1.2 Теоретические основы анализа формальных понятий

Рассмотрим основные положения и типовые обозначения АФП [10, 19, 20, 66–68, 93].

Бинарное отношение $\sqsubseteq$ на множестве P называется отношением (нестрогого) частичного порядка, если оно обладает для всех $x, y, z \in P$ следующими свойствами:

– $x \sqsubseteq x$ (*рефлексивность*),

– если $x \sqsubseteq y$ и $y \sqsubseteq x$, то $x = y$ (*антисимметричность:*),

– если $x \sqsubseteq y$ и $y \sqsubseteq z$, то $x \sqsubseteq z$ (*транзитивность*).

Множество P с определенным на нем отношением частичного порядка $\sqsubseteq$ называется частично упорядоченным множеством (или просто далее упорядоченным множеством) и обозначается через $(P, \sqsubseteq)$. Верхней гранью подмножества $X \subseteq P$ в упорядоченном множестве $(P, \sqsubseteq)$ называется элемент $a \in P$ такой, что $x \sqsubseteq a$ для всех $x \in X$. Точная (или наименьшая) верхняя грань множества X, обозначаемая через $\sup X$, есть такая его верхняя грань a, что $a \sqsubseteq b$ для любой верхней грани b этого множества. Двойственным образом определяется понятие $\inf X$, т. е. точной (или наибольшей) нижней грани множества $X \subseteq P$.

Решеткой называется упорядоченное множество L, в котором любые два элемента x и y имеют точную нижнюю грань (или пересечение, обозначаемое $x \sqcap y$) и точную верхнюю грань (или объединение, обозначаемое $x \sqcup y$). Решетка L называется полной, если любое подмножество $X \subseteq L$ имеет точную верхнюю и точную нижнюю грани.

Пусть определены два непустых конечных множества: множество объектов G (*нем. Gegenstände*) и множество признаков или свойств M (*нем. Merkmale*). Пусть также задано непустое отношение инцидентности $I \subseteq G \times M$. Отношение I содержит информацию о выполнимости свойств из M на объектах из G, т. е. запись $(g, m) \in I$ означает, что объект g обладает признаком m и наоборот, признак m присущ объекту g. Тройку $K = (G, M, I)$ принято называть формальным контекстом [93].

Далее будем полагать, что множества G и M линейно упорядочены (например, лексикографически):

$$G = \left\{g_1, g_2, \ldots, g_{|G|}\right\},$$

$$M = \left\{m_1, m_2, \ldots, m_{|M|}\right\}.$$

В этом случае формальный контекст $K = (G, M, I)$ однозначно задается 0,1-матрицей $T = (t_{ij})$:

$$t_{ij} = \begin{cases} 0, & \text{если } (g_i, m_j) \notin I, \\ 1, & \text{если } (g_i, m_j) \in I, \end{cases}$$

($i = 1, 2, \ldots, |G|$; $j = 1, 2, \ldots, |M|$).

Выберем в $K = (G, M, I)$ два произвольных элемента $g \in G$, $m \in M$ и определим для них отображения φ и ψ:

$$\varphi(g) = \{m \in M \,:\, (g, m) \in I\},$$

$$\psi(m) = \{g \in G \,:\, (g, m) \in I\}.$$

Согласно данному определению, $\varphi(g)$ определяет набор признаков, присущих объекту g, а множество $\psi(m)$ задает семейство объектов, обладающих признаком m. Отображения φ и ψ легко обобщаются на множества $A \subseteq G$ и $B \subseteq M$:

$$\varphi(A) = \bigcap_{g \in A} \varphi(g) = \{m \in M \,:\, \forall\, g \in A\ (g, m) \in I\},$$

$$\psi(B) = \bigcap_{m \in B} \psi(m) = \{g \in G \,:\, \forall\, m \in B\ (g, m) \in I\}.$$

Таким образом, $\varphi(A)$ — набор признаков, характерных для всех объектов из A, а $\psi(B)$ — семейство объектов, имеющих все признаки из B. Отображения φ и ψ определены так, что если $A_1, A_2 \subseteq G$ и $B_1, B_2 \subseteq M$, то

$$\varphi(A_1 \cup A_2) = \varphi(A_1) \cap \varphi(A_2),$$

$$\psi(B_1 \cup B_2) = \psi(B_1) \cap \psi(B_2).$$

Целесообразно положить, что $\varphi(\varnothing) = M$ и $\psi(\varnothing) = G$, т. е. пустому множеству объектов присущи все признаки из M и каждый объект обладает пустым множеством признаков. Согласно традициям АФП для отображений φ и ψ применяется единое обозначение $(\cdot)'$. Тогда приведенные выше формулы записываются следующим образом:

$$g' = \{m \in M \,:\, (g, m) \in I\}, \tag{1.1}$$

$$m' = \{g \in G \,:\, (g, m) \in I\}, \tag{1.2}$$

$$A' = \bigcap_{g \in A} g' = \{m \in M \,:\, \forall\, g \in A \; (g, m) \in I\}\,, \tag{1.3}$$

$$B' = \bigcap_{m \in B} m' = \{g \in G \,:\, \forall\, m \in B \; (g, m) \in I\}\,. \tag{1.4}$$

Если $g \in G$ и $m \in M$, то обозначения g' и m' обычно служат сокращенной формой записи множеств $\varphi(g) = \{g\}'$ и $\psi(m) = \{m\}'$ соответственно.

Отображения $(\cdot)'$ обладают рядом свойств, вытекающих из их определения и вполне реалистичного, постулируемого в анализе данных положения: расширение (сокращение) множества признаков уменьшает (увеличивает) число объектов, обладающих этими признаками. Для всякого формального контекста $K = (G, M, I)$ и любых подмножеств $B_1, B_2 \subseteq M$ верны свойства:

– если $B_1 \subseteq B_2$, то $(B_2)' \subseteq (B_1)'$ (*антимонотонность*);

– $B_1 \subseteq (B_1)''$, где $(B_1)'' = ((B_1)')' \subseteq M$ (*экстенсивность*).

Аналогичные свойства справедливы для любых подмножеств $A_1, A_2 \subseteq G$:

– если $A_1 \subseteq A_2$, то $(A_2)' \subseteq (A_1)'$ (*антимонотонность*);

– $A_1 \subseteq (A_1)''$, где $(A_1)'' = ((A_1)')' \subseteq G$ (*экстенсивность*).

Заметим, что согласно указанным свойствам, отображения φ и ψ составляют пару соответствий Галуа между множествами 2^G и 2^M частично упорядоченными по теоретико-множественному включению. Известно, что для соответствий Галуа φ и ψ верны равенства:

$$\varphi(\psi(\varphi(A))) = \varphi(A), \qquad \psi(\varphi(\psi(B))) = \psi(B)$$

или, то же самое, в единых обозначениях

$$\big((A')'\big)' = (A'')' = A', \qquad ((B')')' = (B'')' = B'. \tag{1.5}$$

Двойное применение отображения $(\cdot)'$ устанавливает оператор замыкания $(\cdot)''$ на 2^M в алгебраическом смысле. Этому оператору присущи свойства:

– для любого $B \subseteq M$ всегда $B \subseteq B''$ (*рефлексивность*);

– если $B_1 \subseteq B_2 \subseteq M$, то $(B_1)'' \subseteq (B_2)'' \subseteq M$ (*монотонность*);

– для любого $B \subseteq M$ всегда $(B'')'' = B''$ (*идемпотентность*).

Множество $(B)''$ можно трактовать как набор признаков, которые неизменно появляются в объектах формального контекста $K = (G, M, I)$ вместе с признаками из B, причем это множество является наибольшим по включению в пределах этого контекста. Если $B = B''$, то B называется замкнутым множеством относительно оператора $(\cdot)''$. Аналогичным образом можно определить оператор замыкания на 2^G.

Из рефлексивности оператора $(\cdot)''$ и антимонотонности отображений $(\cdot)'$ вытекает справедливость следующего высказывания: для всякого формального контекста $K = (G, M, I)$ и любых $A \subseteq G$ и $B \subseteq M$ включение $A \subseteq B'$ верно тогда и только тогда, когда $B \subseteq A'$.

С учетом (1.3) и (1.4) замыкание для $B \subseteq M$ относительно формального контекста $K = (G, M, I)$ можно непосредственно вычислить по формуле:

$$B'' = (B')' = \varphi(\psi(B)) = \begin{cases} \left(\bigcap\limits_{m \in B} m'\right)' = \bigcap\limits_{g \in B'} g', & \text{если } B' \neq \varnothing, \\ M, & \text{если } B' = \varnothing. \end{cases} \tag{1.6}$$

Пара множеств (A, B), $A \subseteq G$, $B \subseteq M$, таких, что $A' = B$ и $B' = A$, называется формальным понятием формального контекста $K = (G, M, I)$ с объемом A и содержанием B. Далее в ряде случаев определение «формальный» перед словами «контекст» или «понятие» будет опускаться.

Из (1.5) и определения оператора $(\cdot)''$ вытекает справедливость высказывания: пара множеств (A, B) является формальным понятием тогда и только

тогда, когда $A = A''$ и $B = B''$. Очевидно также, что всякое формальное понятие уникально в заданном контексте, т. е. отличается от других формальных понятий объемом и/или содержанием. Если формальный контекст представлен 0,1-матрицей T, то при $A \neq \varnothing$ и $B \neq \varnothing$ формальному понятию (A, B) отвечает максимальная полная подматрица матрицы T. Строки этой подматрицы соответствуют элементам из A, а столбцы — элементам из B. Здесь под максимально полной подматрицей понимается подматрица, все элементы которой равны 1 и которая не содержится в других полных подматрицах.

Обозначим через FC множество всех формальных понятий формального контекста $K = (G, M, I)$. Пусть $(A_1, B_1), (A_2, B_2) \in FC$. Множество FC частично упорядочено отношением

$$(A_1, B_1) \sqsubseteq (A_2, B_2) \tag{1.7}$$

тогда и только тогда, когда $A_1 \subseteq A_2$. Отметим, что последнее эквивалентно условию $B_2 \subseteq B_1$. Согласно (1.7) формальное понятие (A_2, B_2) является более общим, чем формальное понятие (A_1, B_1). Поскольку понятие (A_2, B_2) имеет меньшее число присущих признаков, следовательно, больший набор объектов, которые обладают этими признаками.

Каждое формальное понятие $(A, B) \in FC$ определяет для исследуемой предметной области совокупность однородных объектов A со своим специфичным набором признаков B. Если в формальном контексте $K = (G, M, I)$ нет признаков, которые присущи всем объектам из G, то множество FC будет содержать формальное понятие $(G, \varnothing)$. Если в контексте нет объектов, обладающих всеми признаками из M, то $(\varnothing, M) \in FC$. Если имеют место оба случая одновременно, то $(G, \varnothing) \in FC$ и $(\varnothing, M) \in FC$. Эти формальные понятия называются тривиальными.

По определению формального контекста $K = (G, M, I)$ отношение I непустое. Следовательно, отвечающая формальному контексту матрица T всегда ненулевая, а соответствующее ему множество FC не является пустым.

Определим на FC операции объединения $\sqcup$ и пересечения $\sqcap$ через одноименные теоретико-множественные операции $\cap$ и $\cup$ следующим образом:

$$(A_1, B_1) \sqcup (A_2, B_2) = ((B_1 \cap B_2)', B_1 \cap B_2), \tag{1.8}$$

$$(A_1, B_1) \sqcap (A_2, B_2) = (A_1 \cap A_2, (A_1 \cap A_2)'). \tag{1.9}$$

Тогда упорядоченное множество $(FC, \sqsubseteq)$ образует решетку $L = (FC, \sqcap, \sqcup)$. Установленные соотношениями (1.8) и (1.9) операции $\sqcap$ и $\sqcup$ удовлетворяют законам ассоциативности, коммутативности, идемпотентности и поглощения. Решетка $L = (FC, \sqcap, \sqcup)$ называется решеткой формальных понятий контекста $K = (G, M, I)$ и обозначается через L. Известно, что L является полной решеткой. Единицей решетки L является формальное понятие (G, G'), содержащее все объекты контекста $K = (G, M, I)$, а нулем — формальное понятие (M', M), имеющее множество всех признаков рассматриваемого контекста. Подрешеткой решетки L назовем $L_X \subseteq L$ такое, что если $(A_1, B_1) \in L_X$ и $(A_2, B_2) \in L_X$, то $(A_1, B_1) \sqcap (A_2, B_2) \in L_X$, $(A_1, B_1) \sqcup (A_2, B_2) \in L_X$. Подрешетка L_X сама является решеткой с операциями $\sqcap$, $\sqcup$, определенными формулами (1.8) и (1.9).

Решетка L (или L_X) связывает все (или часть) элементы множества FC в определенную иерархическую структуру. Чем выше уровень расположения формального понятия в решетке, тем оно является более общим по отношению к другим формальным понятиям, расположенным ниже в этой решетке. Решетку формальных понятий можно рассматривать в качестве концептуальной модели исследуемой предметной области, которая определяет классы однородных объектов и связи между ними. Здесь в роли классов выступают объемы соответствующих формальных понятий, а в роли наборов атрибутов, характеризующих объекты этих классов, — содержания формальных понятий. Связи между классами определяются отношением (1.6).

Рассмотрим демонстрационный пример.

Пример 1.1. Задан контекст $K = (G, M, I)$ с матрицей инцидентности I, приведенной в таблице 1.1, где $G = \{1, 2, 3, 4, 5\}$ — множество объектов, $M = \{a, b, c, d, e\}$ — множество признаков. Традиционно данный контекст ис-

пользуется в качестве примера в работах по алгоритмам вычисления формальных понятий для демонстрации и тестирования. Далее для краткости при написании множеств опустим фигурные скобки, запятые между элементами множеств. Например, вместо $FC = \{(\{1,2,3\},\{a,c,d\})\}$ пишем $FC = \{(123, acd)\}$. Поскольку множества G и M линейно упорядочены, то элементы этих множеств расположены в лексикографическом порядке.

Таблица 1.1 — Матрица инцидентности I контекста $K = (G, M, I)$

	a	*b*	*c*	*d*	*e*
1	1	0	1	1	0
2	0	1	1	0	1
3	1	1	1	0	1
4	0	1	0	0	1
5	1	1	1	0	1

Для рассматриваемого контекста $K = (G, M, I)$ получим следующее множество формальных понятий:

$$\begin{aligned} FC \;=\; & \{(G,\varnothing),(1,acd),(135,ac),(1235,c),(35,abce), \\ & (235,bce),(2345,be),(\varnothing,M)\}. \end{aligned} \tag{1.10}$$

Решетка формальных понятий L заданного контекста $K = (G, M, I)$ изображена на рисунке 1.1. □

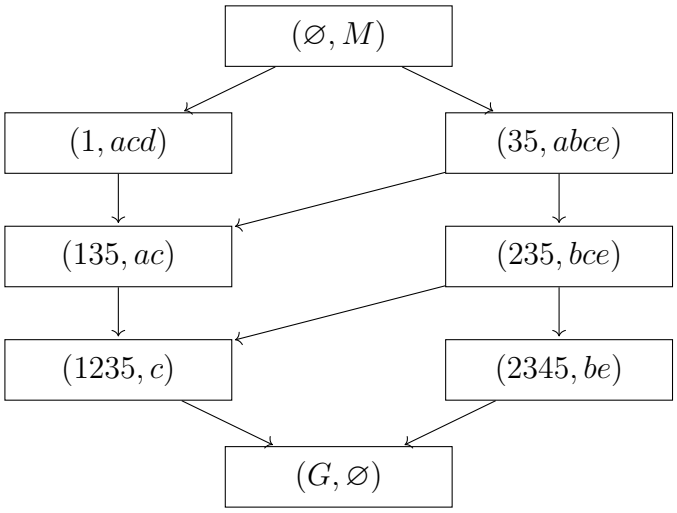


Рисунок 1.1 — Решетка формальных понятий L контекста $K = (G, M, I)$

1.3 Постановка задачи нахождения всех формальных понятий

С применением методов АФП решаются различные прикладные задачи, связанные с классификацией и кластеризацией данных, выявлением зависимостей между данными и семантическим анализом естественно-языковых текстов [11, 21, 35, 44, 46, 88, 102, 113, 114, 117, 126]. В них формальные понятия трактуются как перекрестные ассоциации, кластеры или бикластеры. В рамках АФП решение указанных задач сводится к нахождению всех формальных понятий исходного формального контекста с последующим связыванием их в решетку. Полученная решетка служит концептуальной моделью исследуемой предметной области и основой для решения указанных прикладных задач.

При всей привлекательности методов АФП их практическое применение ограничивается высокой трудоемкостью процесса извлечения множества формальных понятий из контекста большой размерности. В рамках АФП задача нахождения всех формальных понятий формулируется следующим образом.

Задан формальный контекст $K = (G, M, I)$.

Требуется найти для $K = (G, M, I)$ множество FC всех формальных понятий.

Задача нахождения всех формальных понятий контекста детально изучена в работах [88, 93, 103]. Простейшим способом осуществления этих действий является перебор всех возможных подмножеств множества объектов или признаков с вычислением для каждого из них замыкания по формуле (1.6). Далее с помощью (1.7)—(1.9) строится решетка L для исходного формального контекста. Данная задача относится к комбинаторным перечислительным задачам и является $\#P$-полной [103]. Высокая вычислительная сложность задачи объясняется тем, что в общем случае число формальных понятий экспоненциально зависит от размера исходного контекста. Например, это имеет место для контекста вида $K = (G, G, \neq)$. Такому контексту соответствует 0,1-матрица, все элементы которой равны единице, кроме диагональных элементов. Легко убедиться, что такой контекст содержит ровно $2^{|G|}$ формальных понятий.

На сегодняшний день для определения множества FC и построения решетки $L = (FC, \sqcap, \sqcup)$ разработано много алгоритмов [45, 80, 84, 89, 94, 95, 99–101, 106, 107, 112]. Традиционно эти алгоритмы могут быть сгруппированы следующим образом:

– пакетные алгоритмы, которые строят решетку понятий из ранее найденных формальных понятий;
– инкрементные алгоритмы, которые достраивают решетку понятий посредством постепенного добавления объектов и пересечения с имеющимися формальными понятиями.

В таблице 1.2 приведены основные из известных на сегодняшний день алгоритмы нахождения множества всех формальных понятий FC и построения решетки формальных понятий L.

Таблица 1.2 — Алгоритмы нахождения множества FC и построения решетки L

Алгоритм	Вид алгоритма	Стратегия расчета
Bordat [84]	Пакетный	Сверху-вниз: алгоритм начинает построение L с максимального формального понятия, а затем из каждого вычисленного формального понятия порождаются все его нижние соседи. Процесс повторяется для каждого полученного формального понятия
NextClosure [94]	Пакетный	Лексикографический порядок
Close-by-One [101]	Пакетный	Лексикографический порядок
Nourine [112]	Инкрементный	Лексикографическое дерево
Lindig [106]	Пакетный	Снизу-вверх: алгоритм начинает построения L с минимального формального понятия. Затем для каждого формального понятия, которое генерируется в первый раз, порождает всех его верхних соседей
Godin [95]	Инкрементный	Хэш-функция
Dowling [89]	Инкрементный	Лексикографический порядок
Norris [99]	Инкрементный	Лексикографический порядок

Время выполнения указанных алгоритмов в худшем случае составляет $O(|FC| \cdot |G|^2 \cdot |M|)$. Поскольку величина $|FC|$ экспоненциально зависит от $|G|$ и $|M|$, то время выполнения данных алгоритмов также может быть экспоненциальным.

Наиболее известными программными системами являются Concept Explorer, ToscanaJ, Galicia, Lattice Minner, OpenFCA, FCART [27, 81, 86, 104, 111, 116, 123]. Многие из них находятся в открытом доступе. Программа Concept

Explorer позволяет обрабатывать формальные контексты, шкалировать многозначные признаки формального контекста и визуализировать решетки формальных понятий [27]. Автоматическое формирование исходного контекста на основе реляционных баз данных реализовано в системе ToscanaJ [81]. Программа Galicia имеет широкие возможности по визуализации решеток формальных понятий в трехмерном пространстве, а основная цель OpenFCA — отображение решеток через веб-приложения [86, 123]. Lattice Minner — программный инструмент для построения, визуализации и манипулирования решетками формальных понятий [104]. Все эти программные средства являются специализированными продуктами. Создатели программы FCART объединили полный цикл исследований с применением методов АФП в одну универсальную интегрированную среду [111].

В настоящее время актуальны исследования по снижению вычислительной сложности задачи нахождения всех формальных понятий. Первое направление исследований связано с разработкой новых алгоритмов отбора информативных, релевантных формальных понятий при построении решетки L [25, 35, 64, 77, 78]. Такой подход к решению рассматриваемой задачи позволяет уменьшить ее выход. Второе направление исследований рассматривает повышение производительности существующих алгоритмов нахождения множества FC за счет уменьшения входа задачи путем его декомпозиции [83, 90, 91, 119, 121]. Такое направление исследований является более универсальным. В данной работе применяется декомпозиционный подход с сохранением всех искомых формальных понятий. Основная цель обоих направлений исследований — сделать более доступными методы АФП для анализа больших данных.

Важно отметить, что задача нахождения всех формальных понятий контекста $K = (G, M, I)$ эквивалентна задаче определения всех максимально полных подматриц 0,1-матрицы T, отвечающей этому контексту. Существуют и другие родственные с ней задачи, например задачи, связанные с нахождением биклик в заданном двудольном графе. В самом деле, бинарную матрицу T можно рассматривать в качестве матрицы смежности двудольного графа, две доли которого соответствуют множеству строк и множеству столбцов матрицы T. Тогда

всякая полная подматрица матрицы T определяет в заданном двудольном графе полный двудольный подграф, т. е. биклику, а максимально полная подматрица — максимальную биклику этого графа. К поиску максимальных биклик сводятся следующие теоретико-графовые задачи:

– в заданном двудольном графе найти все максимальные биклики;
– для заданного двудольного графа найти наименьшее покрытие всех ребер максимальными бикликами;
– в заданном двудольном графе найти наибольшую биклику;
– для заданного двудольного графа найти наименьшее бикликовое разбиение множества его вершин;
– является ли заданный двудольный граф (k, l)-редким? По определению такой двудольный граф не содержит биклик размера $k \times l$.

Все указанные выше задачи относятся к классу $\#P$-полных или NP-полных задач [22, 24]. Известно, что в общем случае число максимальных биклик графа экспоненциально зависит от числа вершин [109, 118, 127]. Также доказано, что двудольный граф на n вершинах может содержать до $2^{n/2} \approx 1,41^n$ максимальных биклик [115, 124]. Перечисленные теоретико-графовые задачи достаточно хорошо изучены и имеют многочисленные приложения [24–26, 124]. Однако большинство известных алгоритмов решения этих задач неприемлемо долго работают на графах большой размерности.

В рамках диссертационного исследования предлагается использование декомпозиционного подхода для повышения производительности существующих алгоритмов нахождения множества всех формальных понятий.

1.4 Выводы по главе 1

1. В интеллектуальном анализе данных, включая тексты на естественном языке, изучаемая предметная область часто представляется объектно-признаковой таблицей, в которой каждый столбец соответствует некоторому признаку, а каждая строка определяет признаковое описание отдельного объекта. Подобное представление позволяет при анализе данных применять АФП, представлять объектно-признаковую таблицу формальным контекстом и моделировать его 0,1-матрицей.

2. С применением методов АФП решаются различные прикладные задачи, связанные с классификацией и кластеризацией данных, выявлением зависимостей между данными и семантическим анализом естественно-языковых текстов. В них формальные понятия трактуются как перекрестные ассоциации, кластеры или бикластеры. В рамках АФП решение указанных задач сводится к нахождению всех формальных понятий исходного формального контекста с последующим связыванием их в решетку. Полученная решетка служит концептуальной моделью исследуемой предметной области и основой для решения указанных прикладных задач. Данная решетка может быть построена из формального контекста предметной области, представленного 0,1-матрицей.

3. Задача нахождения всех формальных понятий формального контекста относится к комбинаторным перечислительным задачам и является $\#P$-полной. В общем случае число формальных понятий экспоненциально зависит от размера исходного контекста. Эта объясняет высокую вычислительную сложность данной задачи. В настоящее время актуальны исследования по снижению вычислительной сложности данной задачи. Первое направление исследований связано с разработкой алгоритмов отбора релевантных формальных понятий при построении решетки. Такой подход позволяет уменьшить выход рассматриваемой задачи. Второе направление исследований связано с повышением производительности существующих алгоритмов нахождения формальных понятий за счет уменьшения входа задачи путем его декомпозиции с сохранением всех искомых формальных понятий. Такое направление исследований является более универсальным. Основная цель обоих направлений исследований — сделать более доступными методы АФП для анализа больших данных.

4. В настоящей диссертационной работе предлагается метод и алгоритмы, реализующие один из возможных подходов второго направления исследований.

Глава 2 Снижение размерности формального контекста без потери искомых формальных понятий

Вторая глава содержит основные результаты диссертационного исследования, связанные со снижением трудоемкости процесса нахождения множества всех формальных понятий и построения для них решетки за счет применения декомпозиционного подхода, разработкой метода декомпозиции формального контекста на части (названные фрагментами) без потери искомых формальных понятий и алгоритма, реализующего предложенный метод декомпозиции. В ней решаются задачи 1 – 3 диссертационного исследования. Основные результаты решения данных задач опубликованы в работах [13, 14, 47, 48, 87, 108].

В 2.1 подробно описывается метод декомпозиции формального контекста, определяются фрагменты формального контекста и исследуются свойства этих фрагментов. Свойства фрагментов исследуются для установления правил эффективной организации процесса декомпозиции и восстановления искомого решения, исходя из решений, полученных для подзадач. Доказывается, что разложение формального контекста на фрагменты является «неискажающим» относительно формальных понятий: при декомпозиции ни одно формальное понятие не теряется и не появляются новые формальные понятия.

В 2.2 приводится описание алгоритма формирования системы фрагментов с учетом правил разложения контекста на части и остановки процесса разложения. Далее в 2.3 описывается алгоритм восстановления искомого решения на основе решений, полученных для подзадач. Оценивается вычислительная сложность данного алгоритма. В 2.4 приводится описание алгоритмов реализации запросов на извлечение знаний из решетки формальных понятий при решении практических задач.

В 2.5 рассматриваются процедуры предобработки формального контекста для снижения времени вычисления всех формальных понятий исходного контекста. В подразделе 2.6 приводится анализ результативности разработанных метода, алгоритмов и процедур.

2.1 Метод декомпозиции контекста без потери формальных понятий

Декомпозиционный подход к решению задачи нахождения всех формальных понятий заданного контекста — это сведение ее к конечной серии подзадач. Каждая из этих подзадач — уменьшенная копия исходной задачи, которая решается на некоторой части заданного контекста. Процесс декомпозиции направлен на последовательное уменьшение размеров частей контекста. В итоге формируется конечное множество различных частей (в общем случае разного размера и имеющих непустое пересечение). Процесс декомпозиции реализуется итерационно, поскольку рекурсия в подобных случаях более трудоемка по времени [15]. Для эффективной организации процесса декомпозиции требуется определить: правило разложения контекста на части (что является частью и как ее выделять в контексте); оценку числа частей, получаемых на каждой итерации разложения; правило остановки процесса разложения. Кроме того, для всякого декомпозиционного метода решения задачи обязательны правила восстановления искомого решения, исходя из решений, полученных для подзадач. Полиномиальность по времени процедур разделения исходных данных решаемой задачи на части — требование, при выполнении которого достигается эффект декомпозиции. Далее опишем предлагаемый метод декомпозиции формального контекста. Докажем, что данный метод позволяет разлагать контекст без потери формальных понятий, а также установим правила эффективной организации процесса декомпозиции.

Пусть $K = (G, M, I)$ — контекст, FC — множество всех его формальных понятий и T — соответствующая ему 0,1-матрица.

Определение 2.1. Контекст $K_1 = (G_1, M_1, I_1)$ назовем частью контекста $K = (G, M, I)$, если $G_1 \subseteq G$, $M_1 \subseteq M$ и для любых $x \in G_1$, $y \in M_1$ отношение $(x, y) \in I_1$ верно тогда и только тогда, когда $(x, y) \in I$.

Заметим, что контексту $K_1 = (G_1, M_1, I_1)$ отвечает подматрица матрицы T, у которой удалены строки, соответствующие объектам из $G \backslash G_1$, и столбцы, соответствующие признакам из $M \setminus M_1$. Всякое нетривиальное формальное понятие из FC можно рассматривать в роли части контекста $K = (G, M, I)$.

Части $K_1 = (G_1, M_1, I_1)$ и $K_2 = (G_2, M_2, I_2)$ контекста $K = (G, M, I)$ будем считать различными, если $G_1 \neq G_2$ и/или $M_1 \neq M_2$.

Определение 2.2. Разложение контекста $K = (G, M, I)$ на конечное множество различных частей назовем «неискажающим» относительно формальных понятий, если оно удовлетворяет следующим условиям:

– каждая часть содержит, по крайней мере, одно формальное понятие из FC;
– ни одно формальное понятие из FC не теряется и не возникают новые формальные понятия.

«Неискажающее» разложение также называют «безопасным» разложением [14, 108]. Если 0,1-матрица T полная, то результирующее множество будет состоять только из одной части, представляющей сам контекст $K = (G, M, I)$, и эта часть будет содержать только одно формальное понятие (G, M). Очевидно, что наибольшее число различных частей, на которые можно разложить контекст, равно числу $|FC|$ формальных понятий контекста $K = (G, M, I)$. Поскольку существуют контексты, для которых число формальных понятий экспоненциально зависит от $|G|$ и $|M|$, то целесообразно оценить число частей, получаемых на каждой итерации разложения, и определить правило остановки для реализации всего процесса разложения за полиномиальное время.

Пусть $g \in G$ и $m \in M$ — произвольные элементы контекста $K = (G, M, I)$.

Определение 2.3. Пары множеств (g'', g') и (m', m'') образуют формальные понятия, первое из которых назовем объектным, а второе — признаковым формальным понятием контекста $K = (G, M, I)$.

Обозначим через $O = \{(g'', g') : \forall g \in G\} \subseteq FC$ множество всех объектных формальных понятий и через $S = \{(m', m'') : \forall m \in M\} \subseteq FC$ множество всех признаковых формальных понятий контекста $K = (G, M, I)$.

Предложение 2.1. *Всякое объектное формальное понятие (g'', g') формального контекста $K = (G, M, I)$ имеет самое большое по размеру содержание среди других формальных понятий, имеющих в объеме объект $g \in G$, а признаковое формальное понятие (m', m'') обладает самым большим объемом среди других формальных понятий, имеющих в содержании признак $m \in M$.*

Доказательство. Справедливость предложения 2.1 непосредственно следует из свойств оператора $(\cdot)''$ и определения формального понятия. □

Определение 2.4. Пара формальных понятий $(g'', g') \in O$, $(m', m'') \in S$ определяет фрагмент $\omega = (m', g', J)$ как часть контекста $K = (G, M, I)$, если

$$(g'', g') \sqsubseteq (m', m'')\,, \tag{2.1}$$

что эквивалентно $g'' \subseteq m'$ (или $m'' \subseteq g'$).

Про такой фрагмент будем говорить, что он образован элементами $g \in G$ и $m \in M$. Далее вместо $\omega = (m', g', J)$ будем кратко писать $\omega = (m', g')$ или (m', g').

Пример 2.1. Рассмотрим формальный контекст $K = (G, M, I)$ из примера 1.1. Зафиксируем объект $g = \{2\}$ и признак $m = \{b\}$. Тогда объектное понятие имеет вид $(g'', g') = (235, bce)$ (таблица 2.1 а), а признаковое понятие — $(m', m'') = (2345, be)$ (таблица 2.1 б). Нетрудно увидеть, что для рассматриваемых объектного и признакового понятий условие (2.1) выполняется. Следовательно, согласно определению 2.4 пара формальных понятий $(235, bce)$, $(2345, be)$ образует фрагмент $\omega = (m', g') = (2345, bce)$ контекста $K = (G, M, I)$ (таблица 2.1 в). □

Таблица 2.1 — Пример формирования фрагмента $\omega = (2345, bce)$ как части контекста $K = (G, M, I)$

	a	*b*	*c*	*d*	*e*
1	1	0	1	1	0
2	0	1	1	0	1
3	1	1	1	0	1
4	0	1	0	0	1
5	1	1	1	0	1

а) объектное понятие $(g'', g') = (235, bce)$

	a	*b*	*c*	*d*	*e*
1	1	0	1	1	0
2	0	1	1	0	1
3	1	1	1	0	1
4	0	1	0	0	1
5	1	1	1	0	1

б) признаковое понятие $(m', m'') = (2345, be)$

	a	*b*	*c*	*d*	*e*
1	1	0	1	1	0
2	0	1	1	0	1
3	1	1	1	0	1
4	0	1	0	0	1
5	1	1	1	0	1

в) фрагмент $\omega = (2345, bce)$, образованный элементами $g = \{2\}$, $m = \{b\}$

Предложение 2.2. *Для всякого формального контекста* $K = (G, M, I)$ *и любых* $(g'', g') \in O$, $(m', m'') \in S$ *отношение порядка* $(g'', g') \sqsubseteq (m', m'')$ *выполняется тогда и только тогда, когда* $(g, m) \in I$.

Доказательство. Пусть $(g'', g') \sqsubseteq (m', m'')$. Тогда $g'' \subseteq m'$, $m'' \subseteq g'$. Согласно рефлексивности оператора $(\cdot)''$ имеем

$$\{g\} \subseteq g'' \subseteq m', \quad \{m\} \subseteq m'' \subseteq g'.$$

Из (1.1), (1.2) следует, что $(g, m) \in I$.

Докажем обратное. Пусть $(g, m) \in I$. Это означает, что

$$\{g\} \subseteq m', \quad \{m\} \subseteq g'.$$

В силу монотонности оператора $(\cdot)''$ верны включения

$$g'' \subseteq (m')'', \quad m'' \subseteq (g')''.$$

Отсюда в силу рефлексивности оператора $(\cdot)''$ и равенств (1.5) имеем

$$\{g\} \subseteq g'' \subseteq m', \quad \{m\} \subseteq m'' \subseteq g'.$$

Следовательно, $(g'', g') \sqsubseteq (m', m'')$. Предложение 2.2 доказано. □

Из предложения 2.2 следует, что число различных фрагментов, порождаемых всевозможными элементами формального контекста $K = (G, M, I)$, не превышает веса 0,1-матрицы T, т. е. величины $\|T\|$ — числа единичных элементов этой матрицы. Очевидно, что $1 \leq \|T\| \leq |G| \cdot |M|$.

Определение 2.5. Будем говорить, что формальное понятие $(A, B) \in FC$ вложено в фрагмент (m', g') формального контекста $K = (G, M, I)$, и писать $(A, B) \preceq (m', g')$, если $A \subseteq m'$, $B \subseteq g'$.

Всякий фрагмент (m', g') не является пустым, поскольку согласно (2.1) он всегда содержит формальные понятия $(g'', g') \in O$ и $(m', m'') \in S$.

Предложение 2.3. *Всякое нетривиальное формальное понятие* (A, B) *контекста* $K = (G, M, I)$*, которое вложено в фрагмент* (m', g')*, образованный элементами* $g \in G$ *и* $m \in M$*, всегда содержит эти элементы и их замыкания, т. е. если* $(A, B) \preceq (m', g')$*, то неизменно*

1) $g \in A$ *и* $m \in B$*;*

2) $g'' \subseteq A$ *и* $m'' \subseteq B$*.*

Доказательство. Если $(A, B) \preceq (m', g')$, то $A \subseteq m'$, $B \subseteq g'$. В силу антимонотонности отображений $(\cdot)'$ верно

$$m'' \subseteq A', \quad g'' \subseteq B'.$$

Для формального понятия (A, B) по определению $A = B'$, $B = A'$. Тогда

$$m'' \subseteq B, \quad g'' \subseteq A.$$

В силу рефлексивности оператора $(\cdot)''$ имеем $\{m\} \subseteq B$, $\{g\} \subseteq A$. Отсюда следует справедливость обоих высказываний предложения 2.3. □

Согласно предложению 2.3, пару (g'', m'') можно рассматривать в качестве типичного представителя не только фрагмента (m', g'), но и всех формальных понятий контекста $K = (G, M, I)$, вложенных в этот фрагмент. Это правомерно, поскольку подматрица, соответствующая фрагменту (m', g'), во всех строках из g'' и всех столбцах из m'' постоянно имеет единичные элементы.

Соответствие между фрагментами и формальными понятиями контекста устанавливает следующая теорема.

Теорема 2.1. *Для всякого формального контекста* $K = (G, M, I)$*, множества* FC *всех его формальных понятий и любой пары множеств* (A, B)*,* $\varnothing \neq A \subseteq G, \varnothing \neq B \subseteq M$*, справедливы высказывания:*

1) если $(A, B) \in FC$*, то всегда в контексте* $K = (G, M, I)$ *существует фрагмент* $\omega = (m', g')$*,* $g \in G$ *и* $m \in M$*, причем возможно не единственный, в который это формальное понятие вложено;*

2) если (A, B) *— формальное понятие некоторого фрагмента* $\omega = (m', g')$ *формального контекста* $K = (G, M, I)$*, то оно также принадлежит* FC*.*

Доказательство. Докажем первое высказывание. Пусть (A, B) — произвольное формальное понятие контекста $K = (G, M, I)$ и $\varnothing \neq A \subseteq G$, $\varnothing \neq B \subseteq M$. По определению формального понятия для него верны равенства

$$(A, B) = (B', A') = (A'', B'')\,. \tag{2.2}$$

Рассмотрим некоторый объект $g \in A$ и найдем соответствующее ему объектное формальное понятие (g'', g'). Поскольку $\{g\} \subseteq A$, то в силу антимонотонности отображений $(\cdot)'$, монотонности оператора $(\cdot)''$ и равенств (2.2) справедливы отношения

$$A' \subseteq g', \quad g'' \subseteq A'' = A. \tag{2.3}$$

Аналогично для произвольного признака $m \in B$ и признакового формального понятия (m', m'') верны отношения

$$B' \subseteq m', \quad m'' \subseteq B'' = B. \tag{2.4}$$

Из (2.2)–(2.4) вытекает, что $g'' \subseteq m'$ и $m'' \subseteq g'$. Следовательно, пара формальных понятий (g'', g') и (m', m'') определяет фрагмент $\omega = (m', g')$. Кроме того, также

$$A = B' \subseteq m', \quad B = A' \subseteq g'.$$

Это означает, что формальное понятие (A, B) вложено в фрагмент $\omega = (m', g')$. Ясно, что если выбрать другой объект из A и/или другой признак из B, то получим тот же самый фрагмент или возможно другой фрагмент, содержащий формальное понятие (A, B). Первое высказывание теоремы 2.1 доказано.

Докажем теперь второе высказывание. Пусть (A, B) — формальное понятие некоторого фрагмента $\omega = (m', g')$ как части контекста $K = (G, M, I)$. Далее результаты отображений $(\cdot)'$, вычисленные для фрагмента $\omega = (m', g')$, а не контекста $K = (G, M, I)$ в целом, будем отмечать символом ω в нижнем индексе. В этих обозначениях имеем

$$A = B'_\omega \subseteq m', \quad B = A'_\omega \subseteq g'. \tag{2.5}$$

Отношения (2.5) отражают вложенность понятия (A, B) в фрагмент $\omega = (m', g')$. Заметим, что любое формальное понятие (A, B) фрагмента (m', g') всегда содержит объект g и признак m. Действительно, согласно (1.1), (1.2) и определению 2.4 для любого объекта $a \in m'$ верно $(a, m) \in J$ и для всякого признака $b \in g'$ справедливо $(g, b) \in J$. Таким образом, фрагмент $\omega = (m', g', J)$ всегда содержит единичную строку и единичный столбец, соответствующие объекту g и признаку m, образующих этот фрагмент.

Если понятие (A, B) совпадает с объектным (g'', g') или признаковым формальным понятием (m', m''), по которым образован фрагмент $\omega = (m', g')$, то искомое высказывание тривиальным образом выполняется.

Пусть формальное понятие (A, B) отлично от (g'', g') и (m', m'') и для него верны отношения (2.5). Требуется показать, что объем и содержание формального понятия (A, B) не могут выйти за границы фрагмента $\omega = (m', g')$ при вычислении результатов отображений $(\cdot)'$ применительно к контексту $K = (G, M, I)$, т. е. обязательно верны отношения

$$B'_\omega = B' \subseteq m', \quad A'_\omega = A' \subseteq g', \tag{2.6}$$

Предположим, что (2.6) не выполняются, т.е.

$$m' \subset B' \text{ или } g' \subset A'. \tag{2.7}$$

Поскольку всегда $g \in A$ и $m \in B$, то в силу антимонотонности отображений $(\cdot)'$ справедливо $A' \subseteq g'$ и $B' \subseteq m'$, что означает вложенность (B', A') в фрагмент $\omega = (m', g')$. Получили противоречие с (2.7), следовательно (2.6) верно.

Справедливость (2.6) означает, что (A, B) является не только формальным понятием фрагмента $\omega = (m', g')$, но и формальным понятием исходного контекста $K = (G, M, I)$. Теорема 2.1 доказана. □

Заметим, что предположение (2.7) противоречит предложению 2.1, согласно которому формальное понятие (m', m'') обладает самым большим объемом среди других формальных понятий, имеющих в содержании признак $m \in M$, а формальное понятие (g'', g') имеет самое большое по размеру содержание среди других формальных понятий, имеющих в объеме объект $g \in G$.

Согласно теореме 2.1 разложение контекста $K = (G, M, I)$ на фрагменты является «неискажающим» для любого формального понятия из FC. В теореме 2.1 исключены случаи, когда FC содержит хотя бы одно из тривиальных формальных понятий $(G, \varnothing)$, $(\varnothing, M)$. Поскольку всегда верны отношения

$$(\varnothing, M) \sqsubseteq (G, \varnothing), \quad (\varnothing, M) \sqsubseteq (G, G'), \quad (M', M) \sqsubseteq (G, \varnothing),$$

то контекст $K = (G, M, I)$ можно рассматривать как фрагмент (G, M). Следовательно, и даже в этих исключительных случаях каждый фрагмент содержит, по крайней мере, одно формальное понятие из FC, при этом ни одно формальное понятие из FC не теряется. Из теоремы 2.1 вытекает важное практическое следствие: искомое множество FC может быть восстановлено путем объединения множеств формальных понятий, выявленных в фрагментах контекста $K = (G, M, I)$.

Очевидно, что процесс разложения формального контекста $K = (G, M, I)$ на фрагменты может быть организован итерационно, поскольку каждый выявленный на первой итерации фрагмент можно рассматривать в качестве исходного контекста и вновь подвергать декомпозиции. Оценку числа фрагментов, получаемых на каждой итерации разложения, устанавливает предложение 2.2.

Определим теперь правила остановки итерационного процесса разложения. Для этого введем понятие плотности фрагмента. Пусть $|m'| \cdot |g'|$ — размер фрагмента (m', g'), а $\|(m', g')\|$ — число его единичных элементов.

Определение 2.6. Плотностью фрагмента (m', g') назовем величину

$$\sigma(m', g') = \frac{\|(m', g')\|}{|m'| \cdot |g'|}.$$

Верны естественные границы $0 < \sigma(m', g') \leq 1$.

Предложение 2.4. *Если фрагмент (m', g'), сформированный элементами $g \in G$ и $m \in M$, имеет плотность $\sigma(m', g') = 1$, то $g'' = m'$, $m'' = g'$.*

Доказательство. Поскольку $\sigma(m', g') = 1$, то $|m'| \cdot |g'| = \|(m', g')\|$. Это означает, что для любого объекта $g \in m'$ и для всякого признака $m \in g'$ верно $(g, m) \in I$. Отсюда $g'' = m'$, $m'' = g'$. □

Предложение 2.5. *Всякий фрагмент* (m', g') *с плотностью* $\sigma(m', g') = 1$ *содержит ровно одно нетривиальное формальное понятие* (A, B) *контекста* $K = (G, M, I)$*, совпадающее с ним, т. е.* $A = m'$ *и* $B = g'$.

Доказательство. Пусть $(A, B) \preceq (m', g')$, тогда $A \subseteq m'$, $B \subseteq g'$. Из предложения 2.4 следует, что $A \subseteq m' = g''$ и $B \subseteq g' = m''$, а значит,

$$A \subseteq g'' \text{ и } B \subseteq m''.$$

Между тем по предложению 2.3 верны обратные включения

$$g'' \subseteq A \text{ и } m'' \subseteq B.$$

Следовательно, $A = m'$, $B = g'$. □

Из предложения 2.5 следует, что фрагмент (m', g') с плотностью 1 вырождается в нетривиальное формальное понятие и не подлежит дальнейшему разложению. Заметим, что время построения одного фрагмента для контекста $K = (G, M, I)$ составляет $O(|G| \cdot |M|)$. Согласно предложению 2.2 число фрагментов, возникающих при однократном разложении $K = (G, M, I)$, равно

$$\|T\| = \sigma(G, M) \cdot |G| \cdot |M|.$$

В целом время необходимое на однократное разложение контекста $K = (G, M, I)$ на фрагменты составляет $O\left(\sigma(G, M) \cdot |G|^2 \cdot |M|^2\right)$. Таким образом, чем меньше плотность контекста, тем быстрее осуществляется его разложение на фрагменты. При $\sigma(G, M) = 1$ контекст $K = (G, M, I)$ не подлежит разложению. Очевидно, что если ограничить число итераций процесса декомпозиции некоторой константой, то разложение можно осуществить за полиномиальное время. Дополнительно можно установить ограничение на плотность формируемых фрагментов.

Число фрагментов, возникающих на каждой отдельной итерации процесса декомпозиции, в ряде случаев может быть уменьшено за счет удаления вложенных и кратных фрагментов. Рассмотрим для контекста $K = (G, M, I)$ множество фрагментов $\Omega = \left\{\omega_1, \omega_2, \ldots, \omega_{\|T\|}\right\}$, где $\omega_i = (m_i', g_i')$, $i = 1, 2, \ldots, \|T\|$.

Определение 2.7. Будем говорить, что фрагмент $\omega_1 = (m_1', g_1')$ вложен в фрагмент $\omega_2 = (m_2', g_2')$, и писать $\omega_1 \preceq \omega_2$, если верны теоретико-множественные включения

$$m_1' \subseteq m_2', \quad g_1' \subseteq g_2'. \tag{2.8}$$

При $m_1' = m_2'$ и $g_1' = g_2'$ фрагменты ω_1 и ω_2 назовем кратными. Будем считать, что фрагменты ω_1 и ω_2 сравнимы между собой, если $\omega_1 \preceq \omega_2$ или $\omega_2 \preceq \omega_1$, иначе несравнимы. Таким образом, множество Ω частично упорядочено относительно введенного выше отношения порядка. Множество типичных представителей частично упорядочено относительно того же отношения только в обратном порядке. Следующее предложение доказывает данный факт.

Предложение 2.6. *Пусть $h_1 = (g_1'', m_1'')$ — типичный представитель фрагмента $\omega_1 = (m_1', g_1')$, а $h_2 = (g_2'', m_2'')$ — типичный представитель фрагмента $\omega_2 = (m_2', g_2')$. Тогда, отношение порядка $h_2 = (g_2'', m_2'') \preceq h_1 = (g_1'', m_1'')$ выполняется, т. е. верны включения*

$$m_2'' \subseteq m_1'', \quad g_2'' \subseteq g_1'', \tag{2.9}$$

тогда и только тогда, когда $\omega_1 \preceq \omega_2$.

Доказательство. Пусть верны $m_2'' \subseteq m_1'', g_2'' \subseteq g_1''$. Тогда в силу (1.5) и антимонотонности отображений $(\cdot)'$ имеем

$$m_1' \subseteq m_2', \quad g_1' \subseteq g_2'.$$

Из (2.8) следует, что $\omega_1 \preceq \omega_2$. Аналогично доказывается в обратном порядке. □

С учетом теоремы 2.1 справедливо следующее следствие.

Следствие 2.1. *Для любых $\omega_1, \omega_2 \in \Omega$ таких, что $\omega_1 \preceq \omega_2$, все формальные понятия фрагмента ω_1 также являются формальными понятиями фрагмента ω_2 и контекста $K = (G, M, I)$.*

Известно, что в частично упорядоченном множестве всегда можно найти взаимно непересекающиеся цепи [96]. Непустое подмножество $\{\omega_{i1}, \omega_{i2}, \ldots, \omega_{il}\}$ множества Ω является цепью, если все элементы этого подмножества попарно

сравнимы между собой и линейно упорядочены $\omega_{i1} \preceq \omega_{i2} \preceq \ldots \preceq \omega_{il}$. Элемент ω_{il} называется максимальным элементом, а величина l — длиной этой цепи. Цепь называется максимальной, если ее объединение с любым, не принадлежащим ей элементом, цепью не является. Две цепи называются взаимно непересекающимися, если они не содержат общих элементов. Число максимальных взаимно непересекающихся цепей и длина самой длинной такой цепи определяются теоремой Дилоурса [96].

Согласно следствию 2.1 максимальный элемент всякой цепи сохраняет все формальные понятия остальных элементов этой цепи. Данные элементы могут быть удалены и тем самым уменьшено число фрагментов, получаемых на каждой отдельной итерации разложения. Существуют случаи, когда указанный прием не дает эффекта. Например, когда все элементы множества Ω несравнимы между собой, или когда множество Ω линейно упорядочено. Однако эти случаи крайне редки для реальных контекстов.

2.2 Алгоритм формирования системы фрагментов контекста

Предложенный в диссертационной работе метод «неискажающей» декомпозиции контекста реализует алгоритм FindBoxes. Теоретическим обоснованием алгоритма FindBoxes являются теорема 2.1 и предложения 2.2, 2.3, 2.5, 2.6.

Процесс декомпозиции реализуется итерационно. Для организации процесса декомпозиции требуется определить: правило разложения контекста на части (что является частями и как их выделять); оценку числа частей, получаемых на каждой итерации разложения; правило остановки процесса разложения. Основным условием останова итерационного процесса декомпозиции является ограничение на число итераций.

Входными данными алгоритма FindBoxes являются исходный формальный контекст $K = (G, M, I)$ и целое положительное число k — число итераций. Результат работы алгоритма: Ω — множество фрагментов и H — множество типичных представителей фрагментов, входящих в Ω.

Алгоритм FindBoxes включает следующие основные процедуры: Boxes, Delete, SearchChains. В описании FindBoxes используются обозначения:

Алгоритм 1. FindBoxes

Вход: исходный контекст $K = (G, M, I)$, k — количество итераций

begin
$\Omega_1 \leftarrow (G, M, I)$
$\Omega_2 \leftarrow \varnothing$
$H_1 \leftarrow (G'', M'')$
$H_2 \leftarrow \varnothing$
while $(k \neq 0 \ \& \ \Omega_1 \neq \varnothing)$ **do**
    $Q \leftarrow \varnothing$
    $R \leftarrow \varnothing$
    for all $\omega \in \Omega_1$ **do**
        if $\sigma(\omega) \neq 1$ **then**
            Boxes(ω, X, Y)
            $Q \leftarrow Q \cup X$
            $R \leftarrow R \cup Y$
        else
            $\Omega_2 \leftarrow \Omega_2 \cup \omega$
            $H_2 \leftarrow H_2 \cup H_1$
        end if
    end for
    $\Omega_1 \leftarrow Q$
    $H_1 \leftarrow R$
    Delete $(\Omega_1 \cup \Omega_2, H_1 \cup H_2)$
    if $\Omega_1 \neq \varnothing$ **then**
        SearchChains(Ω_1, H_1)
    end if
    $k \leftarrow k - 1$
end while
$\Omega \leftarrow \Omega_1 \cup \Omega_2$
$H \leftarrow H_1 \cup H_2$
end

Выход: Ω — множество фрагментов, H — множество типичных представителей фрагментов

Ω_1 — множество фрагментов, подлежащих дальнейшему разложению, т. е. фрагментов с плотностью отличной от 1;

Ω_2 — множество фрагментов, не подлежащих дальнейшему разложению, т. е. фрагментов с плотностью равной 1. Согласно предложению 2.6 каждый такой фрагмент является формальным понятием;

H_1 — множество типичных представителей фрагментов, входящих в Ω_1;

H_2 — множество типичных представителей фрагментов, входящих в Ω_2.

Процедура Boxes разлагает заданный фрагмент ω, плотность которого отлична от 1, на более мелкие фрагменты и находит для них типичные представители. Вычисление плотности фрагмента производится вспомогательной проце-

дурой σ. Результаты процедуры Boxes записываются в X, Y. Процедура Boxes подробно описана в алгоритме 2.

Алгоритм 2. Boxes

Вход: исходный фрагмент $\omega = (G_1, M_1, I_1)$ как контекст $K = (G, M, I)$ или его часть

begin
for all $g \in G_1$ **do**
    ObConcept (g'', g')
end for
for all $m \in M_1$ **do**
    PrConcept (m', m'')
end for
for all $g \in G_1$ **do**
    for all $m \in M_1$ **do**
        if $((g, m) \in I_1)$ **then**
            $X \leftarrow X \cup (m', g')$
            $Y \leftarrow Y \cup (g'', m'')$
        end if
    end for
end for
end

Выход: X — множество фрагментов, являющихся частями исходного фрагмента, Y — множество их типичных представителей

В процедуре Boxes используются вспомогательные процедуры ObConcept и PrConcept, предназначенные для нахождения множества объектных и признаковых понятий соответственно. Вычисление типичных представителей фрагментов осуществляется согласно предложению 2.3: фрагменту (m', g') ставится в соответствие пара (g'', m''), названная типичным представителем этого фрагмента.

Заметим, что после каждой отдельной итерации процесса разложения множество $\Omega_1 \cup \Omega_2$ не может быть пустым. Поскольку, если разлагаемый фрагмент ω имеет плотность 1, то согласно предложению 2.5 он попадает в Ω_2, иначе данный фрагмент разлагается на более мелкие фрагменты и их число по предложению 2.2 равно весу 0,1-матрицы, представляющей данный фрагмент. По предложению 2.3 каждому фрагменту соответствует типичный представитель. Поэтому также $H_1 \cup H_2 \neq \varnothing$.

Среди фрагментов, возникающих на каждой отдельной итерации процесса декомпозиции, могут быть вложенные и кратные фрагменты. Удаление кратных фрагментов и фрагментов, совпадающих с исходным фрагментом, осуществляет процедура Delete алгоритма FindBoxes. Процедура SearchChains алгорит-

ма FindBoxes выявляет вложенные фрагменты, выполняет построение взаимно непересекающихся цепей частично упорядоченного множества фрагментов Ω_1, и далее находит для этих цепей максимальные элементы. Данная процедура базируется на следствии 2.1 и теореме Дилоурса [96]. Согласно теореме Дилоурса число максимальных взаимно непересекающихся цепей равно длине самой длинной такой цепи.

По следствию 2.1 максимальный элемент всякой цепи сохраняет все формальные понятия остальных элементов этой цепи. Поэтому данные элементы цепи могут быть удалены и тем самым уменьшено число фрагментов, получаемых на каждой отдельной итерации разложения.

Входными данными процедуры SearchChains являются частично упорядоченное множество Ω_1 фрагментов и множество H_1 их типичных представителей. Результатами выполнения данной процедуры являются Ω_1 — множество всех максимальных элементов непересекающихся цепей и H_1 — их типичные представители. Существуют случаи, когда входные данные процедуры SearchChains совпадают с выходными данными. Это имеет место, когда все элементы множества Ω_1 несравнимы между собой, или когда множество Ω_1 линейно упорядочено. Однако эти случаи крайне редки для реальных контекстов.

Алгоритм 3. SearchChains

Вход: Ω_1 — множество фрагментов, H_1 — множество типичных представителей фрагментов

begin
$W \leftarrow \Omega_1$
$V \leftarrow \Omega_1$
$E \leftarrow \varnothing$
for all $\omega_i \in W$ **do**
    for all $\omega_j \in V$ **do**
        if $h_j < h_i$ **then**
            $E \leftarrow E \cup (\omega_i, \omega_j)$
        end if
    end for
end for
if $(E \neq \varnothing)$ **then**
    GreatestMatch (E, U)
    Transitivity (U, Ω_1, H_1)
end if
end

Выход: Ω_1 — множество максимальных элементов взаимно непересекающихся цепей, H_1 — множество типичных представителей фрагментов

Для нахождения взаимно непересекающихся цепей используется известный полиномиальный алгоритм, основанный на вычислении максимального паросочетания двудольного графа, предусматривающий следующие действия [28].

Пусть $\Omega_1 = \{\omega_1, \ldots, \omega_l\}$ — множество фрагментов, подлежащих дальнейшему разложению. Вначале по частично упорядоченному множеству Ω_1 строится ориентированный двудольный граф $Gr\,(W, V; E)$ с долями $W = \{\omega_1, \ldots, \omega_l\}$, $V = \{\omega_1, \ldots, \omega_l\}$ и множеством дуг E, при этом дуга $(\omega_i, \omega_j) \in E$, исходящая из $\omega_i = (m'_i, g'_i)$ и заходящая в $\omega_j = \left(m'_j, g'_j\right)$, создается, если и только если верно отношение вложенности $\omega_i \preceq \omega_j$, т. е. выполняются включения $m'_i \subseteq m'_j, g'_i \subseteq g'_j$.

Данную проверку вложенности фрагментов можно организовать эффективнее, если использовать предложение 2.6. Действительно, пусть $h_i = (g''_i, m''_i)$ является типичным представителем фрагмента $\omega_i = (m'_i, g'_i)$, а $h_j = \left(g''_j, m''_j\right)$ — типичный представитель фрагмента $\omega_j = \left(m'_j, g'_j\right)$. Тогда если $h_j \preceq h_i$, т. е. верны включения $m''_j \subseteq m''_i,\ g''_j \subseteq g''_i$, то в силу антимонотонности отображений $(\cdot)'$ имеем $m'_i \subseteq m'_j, g'_i \subseteq g'_j$. Это означает справедливость вложенности фрагментов: $\omega_i \preceq \omega_j$. Сравнение типичных представителей фрагментов предпочтительнее с вычислительной точки зрения, поскольку, согласно предложению 2.1, объемы m'_i, m'_j и содержания g'_i, g'_j фрагментов ω_i , ω_j, как правило, имеют большую мощность, чем мощность множеств m''_i, m''_j и g''_i, g''_j. В процедуре SearchChains на шаге 7 для построения графа используется проверка условия вложенности фрагментов по их типичным представителям.

Далее на шаге 13 процедура GreatestMatch находит в ориентированном двудольном графе $Gr(W, V; E)$ наибольшее независимое множество дуг, т. е. наибольшее паросочетание $U \subseteq E$. Для нахождения наибольшего паросочетания обычно используют метод чередующихся цепей [28]. Этот метод дает точное решение задачи о наибольшем паросочетании и время его работы для $Gr(W, V; E)$ составляет $O\left(|W| \cdot |E|\right)$ или в худшем случае $O\left(|W|^3\right)$. Для поиска наибольшего паросочетания известна также эффективная жадная эвристика [28]. Она находит приближенное решение задачи — решение, близкое к оптимальному или совпадающее с оптимальным. Именно жадная эвристика реализована в процедуре GreatestMatch.

Суть жадной эвристики заключается в следующем:

– на каждом шаге в доле W ищется неизолированная вершина ω_i с наименьшей степенью исхода;
– выбирается любая дуга, исходящая из данной вершины, например, дуга $(\omega_i, \omega_j) \in E$, и добавляется в искомое паросочетание U;
– затем из доли W удаляется вершина ω_i, а из доли V — вершина ω_j;
– процесс завершается, когда в доле W остаются только изолированные вершины или V становится пустым.

Следует отметить, что всякая изолированная вершина $\omega_i = (m'_i, g'_i)$ образует одноэлементную цепь с максимальным элементом $\omega_i = (m'_i, g'_i)$, а каждая отдельная дуга (ω_i, ω_j) множества U определяет двухэлементную цепь. Нетрудно убедиться, что время работы жадной эвристики составляет $O\left(|W|^2\right)$. Таким образом, жадная эвристика менее трудоемка по времени по сравнению с методом чередующихся цепей.

Далее на шаге 14 процедуры SearchChains строятся максимальные взаимно непересекающиеся цепи с применением вспомогательной процедуры Transitivity. Для этого к U применяется свойство транзитивности отношения вложенности фрагментов. Затем выбираются максимальные элементы полученных цепей, а остальные элементы этих цепей удаляются. Если в Ω_1 имеются фрагменты, которые не вошли ни в одну из полученных цепей, т. е. одноэлементные цепи, то данная процедура осуществляет их добавление к найденному набору максимальных элементов цепей. Таким образом, выходными данными процедуры Transitivity являются Ω_1 множество фрагментов, сформированное из максимальных элементов взаимно непересекающихся цепей и H_1 — множество типичных представителей фрагментов из Ω_1. Именно эти множества — результаты выполнения процедуры SearchChains и одной итерации алгоритма FindBoxes.

Оценим вычислительную сложность алгоритма FindBoxes. Время построения одного фрагмента для контекста $K = (G, M, I)$ процедурой Boxes составляет $O\left(|G| \cdot |M|\right)$. Согласно предложению 2.2 число фрагментов, возникающих при однократном разложении $K = (G, M, I)$, равно

$$\|T\| = \sigma\left(G, M\right) \cdot |G| \cdot |M|\,,$$

где $\sigma(G, M)$ — плотность исходного контекста $K = (G, M, I)$. В целом время необходимое на однократное разложение контекста $K = (G, M, I)$ на фрагменты составляет

$$O\left(\sigma(G, M) \cdot |G|^2 \cdot |M|^2\right). \tag{2.10}$$

Таким образом, чем меньше плотность контекста, тем быстрее осуществляется его разложение на фрагменты.

Для выполнения процедуры Delete требуется $O\left(|G|^2 \cdot |M|^2\right)$ время. Время работы процедуры SearchChains с учетом построения двудольного ориентированного графа и нахождения максимальных элементов взаимно непересекающихся цепей сопоставимо с $O\left(|G|^2 \cdot |M|^2\right)$. Поскольку все указанные процедуры в алгоритме FindBoxes выполняются последовательно, то при числе итераций k процесс разложения исходного контекста на фрагменты алгоритмом FindBoxes выполняется за время $O\left(|G|^{2k} \cdot |M|^{2k}\right)$. Если k фиксировано, то время выполнения алгоритма FindBoxes полиномиальное относительно размера исходного формального контекста. Очевидно, что значение k целесообразно задавать равным не более 3. Поскольку с каждой отдельной итерацией разложения число фрагментов лавинообразно увеличивается. Устранение кратных и вложенных фрагментов сдерживает этот процесс, но не исключает его вовсе.

Ранее было отмечено, что наибольшее число частей, на которые можно разложить контекст без потери формальных понятий, равно числу $|FC|$ формальных понятий контекста $K = (G, M, I)$. Поскольку существуют контексты, для которых число формальных понятий экспоненциально зависит от $|G|$ и $|M|$. Если количество итераций k — достаточно большое число, то на каждой итерации разложения количество частей, подлежащие дальнейшему разложению увеличивается, а их размеры уменьшаются. В этом случае мощность результирующего множества фрагментов будет сопоставима или равна числу $|FC|$ формальных понятий исходного контекста. Таким образом, при больших значениях k процесс декомпозиции формального контекста может оказаться неэффективным.

Для дополнительного ограничения числа частей, получаемых на каждой итерации разложения, следует устанавливать пороговое значение на плотность

фрагментов, подлежащих дальнейшему разложению. Это достигается заменой на шаге 10 алгоритма FindBoxes условия $\sigma(\omega) \neq 1$ условием $\sigma(\omega) < \sigma_0$, где $\sigma(\omega)$ — плотность фрагмента, σ_0 — пороговое значение на плотность фрагментов, которые подлежит дальнейшему разложению. Если задавать значения σ_0 числом близким к 1, то мощность результирующего множества фрагментов может быть сопоставима с числом $|FC|$ формальных понятий контекста $K = (G, M, I)$. А если значение σ_0 достаточно мало (например, 0,1 или 0,2), то исходный контекст может вообще не разлагаться на части. Таким образом, при решении конкретных практических задач необходимо подбирать подходящие значения k и σ_0, например, k устанавливать равным 1 или 2, а σ_0 выбрать из интервала $\sigma_K < \sigma_0 < 1$, где σ_K — плотность исходного контекста $K = (G, M, I)$.

Демонстрирует работу алгоритма FindBoxes следующий пример.

Пример 2.2. Рассмотрим формальный контекст $K = (G, M, I)$ из примера 1.1 и применим к нему алгоритм FindBoxes. Данный контекст имеет плотность, отличную от единицы: $\sigma_K = 16/25 = 0,64 \neq 1$.

При $k = 1$ выполнение алгоритма FindBoxes приводит к следующим результатам. Вначале сформированное множество Ω_1 содержит 16 фрагментов. Затем процедура Delete, выполняемая на шаге 21 алгоритма FindBoxes, удаляет совпадающие фрагменты и выдает лишь 9 различных фрагментов (таблица 2.2).

Таблица 2.2 — Фрагменты и их типичные представители

Множество фрагментов Ω_1	Множество типичных представителей фрагментов H_1
$\omega_1 = (135, acd)$	$h_1 = (1, ac)$
$\omega_2 = (1235, acd)$	$h_2 = (1, c)$
$\omega_3 = (1, acd)$	$h_3 = (1, acd)$
$\omega_4 = (2345, bce)$	$h_4 = (235, be)$
$\omega_5 = (1235, bce)$	$h_5 = (235, c)$
$\omega_6 = (135, abce)$	$h_6 = (35, ac)$
$\omega_7 = (2345, abce)$	$h_7 = (35, be)$
$\omega_8 = (1235, abce)$	$h_8 = (35, c)$
$\omega_8 = (2345, be)$	$h_9 = (2345, be)$

Процедура SearchChains, выполняемая на шаге 23 алгоритма FindBoxes, находит по частично упорядоченному множеству фрагментов Ω_1 четыре максимальные цепи:

P_1: $\omega_3 = (1, acd) \preceq \omega_1 = (135, acd) \preceq \omega_2 = (1235, acd)$;

P_2: $\omega_6 = (135, abce) \preceq \omega_8 = (1235, abce)$;

P_3: $\omega_9 = (2345, be) \preceq \omega_4 = (2345, bce) \preceq \omega_7 = (2345, abce)$;

P_4: $\omega_5 = (1235, bce) \preceq \omega_8 = (1235, abce)$.

Согласно предложению 2.6, типичные представители фрагментов, образующих эти цепи, также упорядочены, но в обратном порядке:

$h_3 = (1, acd) \succeq h_1 = (1, ac) \succeq h_2 = (1, c)$;

$h_6 = (35, ac) \succeq h_8 = (35, c)$;

$h_9 = (2345, be) \succeq h_4 = (235, be) \succeq h_7 = (35, be)$;

$h_5 = (235, c) \succeq h_8 = (35, c)$.

Максимальные элементы цепей P_2, P_4 и их типичные представители полностью совпадают. Таким образом, в результате работы алгоритма SearchChains формируются 3 фрагмента, представленные в таблице 2.3.

Таблица 2.3 — Множества Ω и H, построенные алгоритмом FindBoxes

Фрагмент — максимальный элемент цепи	Плотность фрагмента	Типичный представитель фрагмента	Формальные понятия фрагмента
$\omega_2 = (1235, acd)$	$8/12 \approx 0,67$	$h_2 = (1, c)$	$(135, ac)$ $(1, acd)$ $(1235, c)$
$\omega_7 = (2345, abce)$	$13/16 \approx 0,81$	$h_7 = (35, be)$	$(2345, be)$ $(235, bce)$ $(35, abce)$
$\omega_8 = (1235, abce)$	$13/16 \approx 0,81$	$h_8 = (35, c)$	$(135, ac)$ $(1235, c)$ $(235, bce)$ $(35, abce)$

В третьем столбце таблицы 2.3 указаны формальные понятия, входящие в соответствующие фрагменты. Из таблицы 2.3 видно, что все формальные понятия фрагмента $\omega_8 = (1235, abce)$ содержатся в фрагменте $\omega_2 = (1235, acd)$ или в фрагменте $\omega_7 = (2345, abce)$. Поскольку исходный контекст содержит тривиальные формальные понятия $(G, \varnothing)$, $(\varnothing, M)$, то множество формальных понятий, установленных с использованием разложения контекста на фрагменты, равно

$$\{(G, \varnothing), (1, acd), (135, ac), (1235, c), (35, abce), (235, bce), (2345, be), (\varnothing, M)\},$$

что полностью совпадает с множеством FC, приведенным в (1.10). Таким образом, все формальные понятия контекста $K = (G, M, I)$ сохранены. Это подтверждает справедливость теоремы 2.1.

Если выполнить еще одну итерацию алгоритма FindBoxes, то получим результирующие множества фрагментов и их типичных представителей, представленные в таблице 2.4. Из таблицы 2.4 видно, что увеличение числа итераций приводит к увеличению числа частей, подлежащих дальнейшему разложению, к уменьшению их размеров, к увеличению плотности формируемых фрагментов. Если установить $k = 3$ и $\sigma_0 = 0,8$, то получим результаты работы алгоритма FindBoxes, приведенные в таблице 2.5.

Сравнение результатов выполнения алгоритма FindBoxes, представленные в таблицах 2.4 и 2.5, показывает, что задание ограничения σ_0 на плотность формируемых фрагментов существенно уменьшает мощность результирующих множеств Ω и H. □

Таблица 2.4 — Множества Ω и H после второй итерации разложения

Фрагмент — максимальный элемент цепи	Плотность фрагмента	Типичный представитель фрагмента	Формальные понятия фрагмента
$(135, acd)$	$7/9 \approx 0,78$	$(1, ac)$	$(1, acd)$ $(135, ac)$
$(1235, ac)$	$7/8 \approx 0,875$	$(135, c)$	$(135, ac)$ $(1235, c)$
$(2345, bce)$	$11/12 \approx 0,92$	$(235, be)$	$(235, bce)$ $(2345, be)$
$(235, abce)$	$11/12 \approx 0,92$	$(35, bce)$	$(235, bce)$ $(35, abce)$
$(1235, bce)$	$10/12 \approx 0,83$	$(235, c)$	$(235, bce)$ $(1235, c)$
$(135, abce)$	$10/12 \approx 0,83$	$(35, ac)$	$(135, ac)$ $(35, abce)$

Таблица 2.5 — Результаты работы алгоритма FindBoxes при $k = 3$ и $\sigma_0 = 0,8$

Фрагмент — максимальный элемент цепи	Плотность фрагмента	Типичный представитель фрагмента	Формальные понятия фрагмента
$(1, acd)$	$3/3 = 1$	$(1, acd)$	$(1, acd)$
$(135, ac)$	$6/6 = 1$	$(135, ac)$	$(135, ac)$
$(1235, ac)$	$7/8 \approx 0,875$	$(135, c)$	$(135, ac)$
			$(1235, c)$
$(1235, abce)$	$13/16 \approx 0,81$	$(35, c)$	$(135, ac)$
			$(1235, c)$
			$(235, bce)$
			$(35, abce)$
$(2345, abce)$	$13/16 \approx 0,81$	$(35, be)$	$(2345, be)$
			$(235, bce)$
			$(35, abce)$

2.3 Алгоритм восстановления решетки формальных понятий

Для организации процесса декомпозиции обязательны правила восстановления искомого решения исходя из решений, полученных для подзадач.

Пусть Ω — множество фрагментов исходного контекста $K = (G, M, I)$, полученное в результате выполнения алгоритма FindBoxes. Для восстановления искомого множества FC всех формальных понятий контекста $K = (G, M, I)$ и построения по нему решетки L необходимо:

– в каждом фрагменте $\omega \in \Omega$ найти все формальные понятия FC_ω;

– восстановить искомое множество FC всех формальных понятий контекста $K = (G, M, I)$ с использованием теоретико-множественной операции объединения:

$$FC = \bigcup_{\omega \in \Omega}^{|\Omega|} FC_\omega.$$

Кроме того если требуется найти решетку L формальных понятий исходного контекста $K = (G, M, I)$, то необходимы следующие действия:

– вначале для каждого фрагмента $\omega \in \Omega$ следует построить решетку L_ω;

– затем сформировать множество решеток L_Ω;

– восстановить решетку L формальных понятий контекста $K = (G, M, I)$ из решеток L_Ω.

Нахождение множества формальных понятий FC_ω и связывание этих понятий в решетку L_ω выполняется алгоритмом LatticeBox.

Алгоритм 4. LatticeBox

Вход: Ω — множество фрагментов

1: **begin**
2: $L_\Omega \leftarrow \varnothing$
3: **for all** $\omega \in \Omega$ **do**
4: Lattice (ω, L_ω)
5: $L_\Omega \leftarrow L_\Omega \cup L_\omega$
6: **end for**
7: **end**

Выход: L_{Ω_1} — множество решеток формальных понятий для всех фрагментов из Ω

В алгоритме LatticeBox процедура Lattice создает решетку формальных понятий для одного заданного фрагмента ω на основе формул (1.6)–(1.9). Данная процедура выполняет действия, аналогичные действиям известного алгоритма Close-by-One («замыкай по одному»), который, по мнению многих исследователей, является одним из самых эффективных современных алгоритмов построения решеток формальных понятий [101]. Время выполнения данного алгоритма для контекста $K = (G, M, I)$ составляет $O\left(|FC| \cdot |G|^2 \cdot |M|\right)$. Авторами алгоритма Close-by-One являются С. О. Кузнецов, С. А. Объедков [101].

Для восстановления искомой решетки L, каждую из решеток $L_\omega \in L_\Omega$ представляют ориентированным графом $L_\omega\,(W, E)$, вершинами W которого являются формальные понятия решетки L_ω, а дугами E — связи между формальными понятиями в решетке L_ω. В этом случае восстановление решеток сводится к объединению (наложению) ориентированных графов, соответствующих решеткам из L_Ω [28].

Уточним правила построения $L_\omega\,(W, E)$. Пусть (A, B), (C, D) — формальные понятия решетки L_ω. Тогда дуга $\{(A, B), (C, D)\} \in E$, идущая из (A, B) и в (C, D), имеет место в $L_\omega\,(W, E)$, если верно условие (1.7): $(A, B) \sqsubseteq (C, D)$, если $A \subseteq C$ (или $D \subseteq B$), т. е. формальное понятие (C, D) является более общим,

чем понятие (A, B). Отсюда ориентированный граф $L_\omega(X, E)$ можно представить в виде перечня дуг — последовательности пар вида $\{(A, B), (C, D)\}$, где $(A, B), (C, D) \in FC_\omega$.

Пример 2.3. Для фрагментов из примера 1.1 решетки формальных понятий L_{ω_2}, L_{ω_7}, L_{ω_8}, построенные алгоритмом LatticeBox при $k = 1$, изображены на рисунках 2.1–2.3. Там же указаны списки дуг, описывающие данные решетки. Типичные представители фрагментов выделены в соответствующих формальных понятиях полужирным шрифтом. □

L_{ω_2}:
$\{(1, acd), (135, ac)\}$
$\{(135, ac), (1235, c)\}$

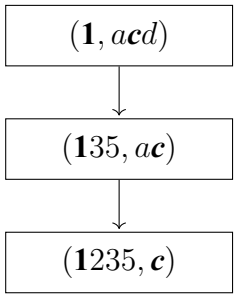


Рисунок 2.1 — Решетка L_{ω_2} для фрагмента $\omega_2 = (1235, acd)$ с типичным представителем $h_2 = (1, c)$

L_{ω_7}:
$\{(35, abce), (235, bce)\}$
$\{(235, bce), (2345, be)\}$

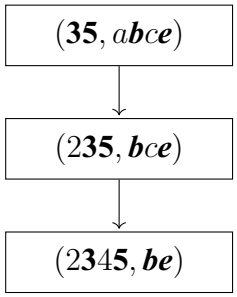


Рисунок 2.2 — Решетка L_{ω_7} для фрагмента $\omega_7 = (2345, abce)$ с типичным представителем $h_7 = (35, be)$

Согласно определению операции объединения графов получим: наложение ориентированных графов $L_{\omega_1} = (W_1, E_1)$, $L_{\omega_2} = (W_2, E_2)$ приводит к ориентированному графу $L_{\omega_1} \cup L_{\omega_2} = L(W, E)$, для которого $W = W_1 \cup W_2$, $E = E_1 \cup E_2$ [28]. Обобщение этой операции на множество L_Ω приводит:

$$L = \bigcup_{\omega \in \Omega}^{|L_\Omega|} L_\omega. \tag{2.11}$$

L_{ω_8} :
$\{(35, abce), (135, ac)\}$
$\{(35, abce), (235, bce)\}$
$\{(135, ac), (1235, c)\}$
$\{(235, bce), (1235, c)\}$

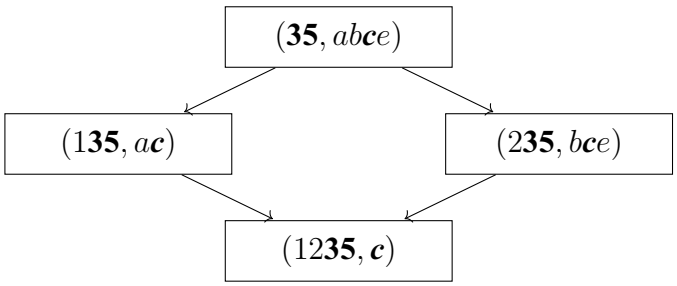


Рисунок 2.3 — Решетка L_{ω_8} для фрагмента $\omega_8 = (1235, abce)$ с типичным представителем $h_8 = (35, c)$

Алгоритм 5. LatticeContext

Вход: L_Ω — множество решеток
1: **begin**
2: $L \leftarrow L_{\omega_1}$
3: **for all** $L_\omega \in L_\Omega \setminus L_{\omega_1}$ **do**
4: **for all** $(W, E) \in L_\omega$ **do**
5: **if** $(W, E) \notin L$ **then**
6: $L \leftarrow L \cup (W, E)$
7: **end if**
8: **end for**
9: **end for**
10: **end**
Выход: L — решетка формальных понятий контекста $K = (G, M, I)$

Построение искомой решетки L формальных понятий из множества решеток L_Ω на основе (2.11) реализуется алгоритмом LatticeContext.

Пример 2.4. Результат объединения решеток формальных понятия L_{ω_2}, L_{ω_7}, L_{ω_8} из примера 2.3, выполненная по алгоритму LatticeContext, дает решетку L (рисунок 2.4).

Построенная решетка полностью совпадает с искомой решеткой L контекста $K = (G, M, I)$, изображенной на рисунке 1.1 главы 1. □

Оценим вычислительную сложность алгоритма LatticeContext. Как было отмечено ранее, задача нахождения всех формальных понятий относится к классу $\#P$-полных задач и носит перечислительный комбинаторный характер. Результирующее число формальных понятий в общем случае может экспоненциально зависеть от размеров исходного контекста $K = (G, M, I)$.

Применение декомпозиционного подхода не меняет сложность рассматриваемой задачи. Его цель — дать возможность на практике решать эту задачу за реальное время. При удачном подборе значений k и σ_0 возможно построение

L :

$\{(\varnothing, M), (1, acd)\}$
$\{(\varnothing, M), (35, abce)\}$
$\{(1, acd), (135, ac)\}$
$\{(35, abce), (235, bce)\}$
$\{(35, abce), (135, ac)\}$
$\{(135, ac), (1235, c)\}$
$\{(235, bce), (2345, be)\}$
$\{(235, bce), (1235, c)\}$
$\{(1235,), (G, \varnothing)\}$
$\{(2345, be), (G, \varnothing)\}$

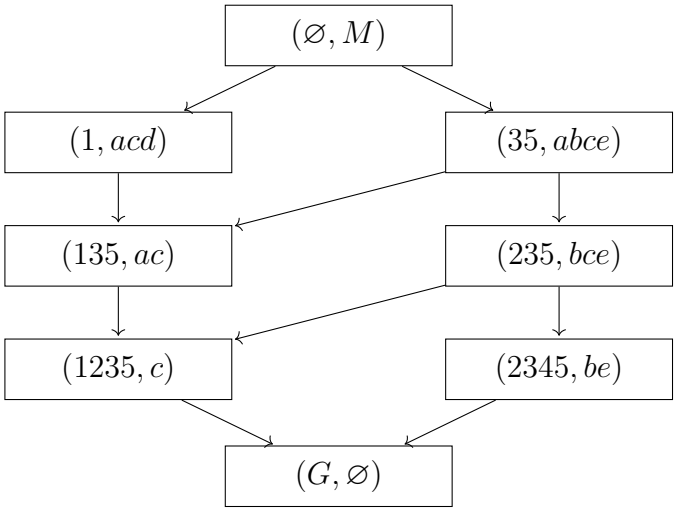


Рисунок 2.4 — Решетка L формальных понятий контекста $K = (G, M, I)$

полиномиального числа $|\Omega| = p(|G|, |M|)$ фрагментов — небольших по размеру частей исходного контекста. Время построения L_ω для отдельного фрагмента $\omega \in \Omega$ контекста $K = (G, M, I)$ алгоритмом Close-by-One составляет

$$O\left(|FC| \cdot |G|^2 \cdot |M|\right).$$

Тогда для построения L_Ω потребуется время

$$O\left(p(|G|, |M|) \cdot |FC| \cdot |G|^2 \cdot |M|\right). \tag{2.12}$$

Сложность по времени объединения решеток, т. е. алгоритма LatticeContext, определяется оценкой (2.12). Указанная оценка сильно завышена. Как показывают вычислительные эксперименты, результаты которых представлены в 2.6, на практике применение декомпозиционного подхода почти всегда оправдано.

Для контекстов большой размерности каждую решетку L_ω целесообразно хранить отдельно с целью более эффективного поиска необходимых формальных понятий в этой решетке, ее визуализации на экране компьютера и объединять только те решетки, которые необходимы пользователю.

2.4 Алгоритмы реализации запросов на извлечение знаний из решетки формальных понятий

Как было отмечено в главе 1, часто пользователю при решении практических задач требуется найти все формальные понятия, включающие в объеме множество объектов X и в содержании множество признаков Y, и связи между этими формальными понятиями. Назовем пару (X, Y), $X \in G$ и $Y \in M$, (X, Y)-запросом на построение маршрута в заданной решетке L.

Для решения этой задачи достаточно обойти решетку и выявить всякое формальное понятие $(A, B) \in FC$ такое, что верны отношения вложенности:

$$X \subseteq A \text{ и } Y \subseteq B. \tag{2.13}$$

При работе с формальным контекстом большой размерности, когда выполнено его разложение на фрагменты, для каждого из которых построена решетка L_ω, хранящаяся отдельно от других решеток, для выполнения (X, Y)-запроса требуется также найти необходимый состав фрагментов исходного контекста и осуществить объединение их решеток по алгоритму LatticeContext. Состав фрагментов определяется исходя из X, Y и образующих элементов фрагментов путем проверки следующего условия: фрагмент $\omega = (m', g')$, образованный элементами $g \in G$ и $m \in M$, включается в состав фрагментов контекста для объединения, если верны включения $X \subseteq m'$ и $Y \subseteq g'$. Все эти действия выполняет алгоритм Query1.

Возможен другой вид (X, Y)-запроса на извлечение знаний из решетки L: установление в L как общих, так и частных понятий для заданного формального понятия $(X, Y) \in FC$. Для решения этой задачи достаточно обойти решетку и выявить такие формальные понятия $(A, B) \in FC$, которые удовлетворяют следующим условиям

$$X \subseteq A, \tag{2.14}$$

$$A \subseteq X. \tag{2.15}$$

Алгоритм 6. Query1

Вход: Ω — множество фрагментов, L_Ω — множество решеток, (X, Y)-запрос

begin
$L_{XY} \leftarrow \varnothing$
$L \leftarrow \varnothing$
for all $\omega \in \Omega$ **do**
    if $X \subseteq m' \;\&\; Y \subseteq g'$ **then**
        $L \leftarrow L \cup L_\omega$
    end if
end for
LatticeContext (L)
for all $\{(A, B), (C, D)\} \in E$ **do**
    if $X \subseteq A \;\&\; X \subseteq C \;\&\; Y \subseteq B \;\&\; Y \subseteq D$ **then**
        $L_{XY} \leftarrow L_{XY} \cup \{(A, B), (C, D)\}$
    end if
end for
end

Выход: L_{XY} — решетка формальных понятий (X, Y)-запроса

Если выполнено условие (2.14), то формальное понятие (A, B) является более общим, чем понятие (X, Y), а если верно условие (2.15), то формальное понятие (A, B) является частным по отношению к (X, Y). Согласно определению формальных понятий, вместо (2.14) и (2.15) можно проверять условия: $B \subseteq Y$ и $Y \subseteq B$. Реализация данного вида запроса приведена в алгоритме Query2. В алгоритме Query2 также как в алгоритме Query1 предусмотрено объединение решеток, соответствующих фрагментам.

Алгоритм 7. Query2

Вход: Ω — множество фрагментов, L_Ω — множество решеток, (X, Y)-запрос

begin
$L_{XY} \leftarrow \varnothing$
$L \leftarrow \varnothing$
for all $\omega \in \Omega$ **do**
    if $X \subseteq m' \;\&\; Y \subseteq g'$ **then**
        $L \leftarrow L \cup L_\omega$
    end if
end for
LatticeContext (L)
for all $\{(A, B), (C, D)\} \in E$ **do**
    if $(X = A \;\&\; Y = B) \mid (X = C \;\&\; Y = D)$ **then**
        $L_{XY} \leftarrow L_{XY} \cup \{(A, B), (C, D)\}$
    end if
end for
end

Выход: L_{XY} — решетка формальных понятий (X, Y)-запроса

Если исходный контекст не подвергался разложению, то время выполнения алгоритмов Query1 и Query2 составляет $O(|FC|)$, иначе в данной оценке необходимо учесть мощность Ω. В итоге получим $O(|\Omega| \cdot |FC|)$.

Пример 2.5. Для решетки формальных понятий L_Ω, изображенной на рисунке 2.4, при $X = \{35\}$ и $Y = \{c\}$ алгоритм Query1 выдает результат, представленный на рисунке 2.5. □

L_{XY} :
$\{(35, abce), (235, bce)\}$
$\{(35, abce), (135, ac)\}$
$\{(135, ac), (1235, c)\}$
$\{(235, bce), (1235, c)\}$

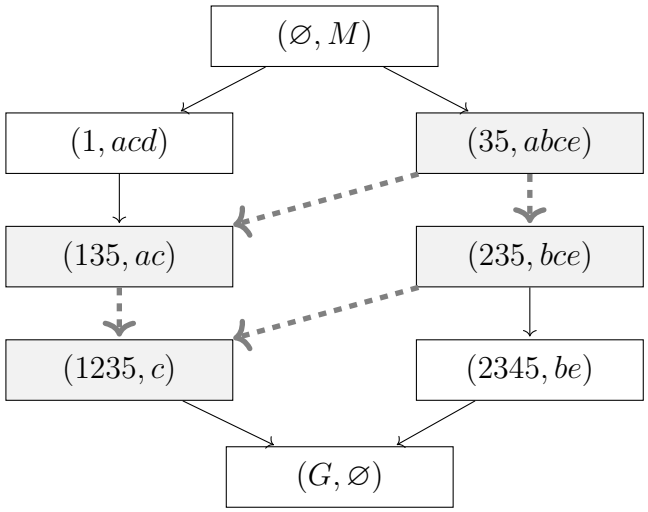


Рисунок 2.5 — Справа: список дуг, описывающих связи между выявленными формальными понятиями при $X = \{35\}$ и $Y = \{c\}$; слева: исходная решетка, на которой отмечены эти связи и соответствующие формальные понятия

Пример 2.6. Для решетки формальных понятий L_Ω, изображенной на рисунке 2.4, при $X = \{235\}$ и $Y = \{bce\}$ алгоритм Query2 выдает результат, представленный на рисунке 2.6. Из рисунка 2.6 видно, что для формального понятия $(235, bce)$ более общими являются формальные понятия $(2345, be)$ и $(1235, c)$, а частным по отношению к нему является формальное понятие $(35, abce)$. □

Для запросов из примеров 2.5, 2.6 объединению подлежат все три решетки, изображенные на рисунках 2.1 – 2.3. Результатом является искомая решетка, представленная на рисунках 2.5, 2.6.

L_{XY} :
$\{(35, abce), (235, bce)\}$
$\{(235, bce), (2345, be)\}$
$\{(235, bce), (1235, c)\}$

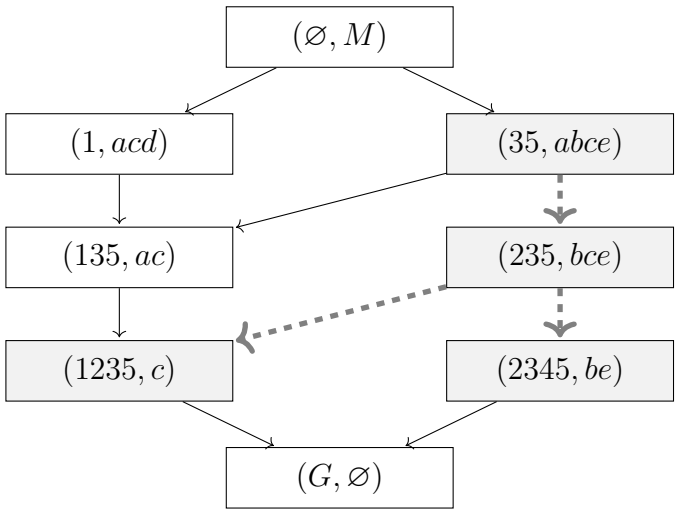


Рисунок 2.6 — Справа: список дуг, описывающих связи между выявленными формальными понятиями при $X = \{235\}$ и $Y = \{bce\}$; слева: исходная решетка, на которой отмечены эти связи и соответствующие формальные понятия

2.5 Процедуры предобработки формального контекста

Снизить время вычисления всех формальных понятий для заданного формального контекста $K = (G, M, I)$ можно, в первую очередь, за счет уменьшения значения величин $|G|$ и $|M|$ путем предобработки контекста $K = (G, M, I)$. Предобработка выполняется так, чтобы не изменилось число и состав формальных понятий в FC. Сокращение может затрагивать как множество объектов, так и множество признаков. Рассмотрим возможные случаи предобработки контекста $K = (G, M, I)$ с соответствующей матрицей инцидентности T.

Случай 1 (*дубликаты строк и столбцов*). Пусть в $K = (G, M, I)$ существует множество объектов $A = \{g_1, g_2\}$ таких, что $g_1' = g_2' = B$. Тогда $A'' = (g_1' \cap g_2')' = (B \cap B)' = (B)' = A$, т. е A является замкнутым множеством. Следовательно, объект g_2 можно удалить из $K = (G, M, I)$ и не учитывать при вычислении формальных понятий. Однако при построении решетки L объект g_2 необходимо добавить в объемы тех формальных понятий, в которые вошел объект g_1. Аналогично, если в контексте $K = (G, M, I)$ существует множество признаков $B = \{m_1, m_2\}$ таких, что $m_1' = m_2' = A$. Тогда $B'' = (m_1' \cap m_2')' = (A \cap A)' = (A)' = B$. Поэтому признак m_2 нужно не учитывать при вычислении формальных понятий, а при построении L необходимо добавить его в содержания тех формальных понятий, в которые вошел m_1.

Случай 2 (*нулевые строки и столбцы*). Если в $K = (G, M, I)$ существует объект $g \in G$ такой, что $g' = \varnothing$, то $g'' = (g')' = (\varnothing)' = G$. Аналогично, если в контексте $K = (G, M, I)$ имеется признак $m \in M$, такой, что $m' = \varnothing$, то $m'' = (m')' = (\varnothing)' = M$. Тогда на момент вычисления формальных понятий объект g и признак m следует отбросить, а затем при построении L объект g добавить в единицу (G, G'), а признак m — в ноль (M', M) этой решетки.

Случай 3 (*единичные строки и столбцы*). Если в $K = (G, M, I)$ существует объект $g \in G$ такой, что $g' = M$, то $g'' = (g')' = (M)' = g$. Тогда объект g необходимо опустить при нахождении формальных понятий, однако затем добавить в решетку L новое формальное понятие (g, M), а объемы всех ранее полученных формальных понятий пополнить объектом g. Аналогично, если имеется признак $m \in M$ такой, что $m' = G$, то m вначале нужно опустить, а потом добавить в содержание всех формальных понятий решетки L.

Время реализации предобработки исходного контекста $K = (G, M, I)$ составляет $O(|G| \cdot |M|)$. При предобработке исходного контекста $K = (G, M, I)$ формальные понятия этого контекста сохраняются и не появляются новые формальные понятия. Процедура предобработки исходного контекста без потери формальных понятий выполняется в самом начале, т. е. перед решением рассматриваемой задачи — задачи нахождения всех формальных понятий с применением или без применения декомпозиционного подхода.

Пример 2.7. Для контекста из примера 1.1 применим только случай 1 — удаление дубликатов, а именно строки 5 (она полностью совпадает со строкой 3) и столбца e (он полностью совпадает со столбцом b). На рисунке 2.7 представлен исходный контекст $K = (G, M, I)$ и соответствующие ему множества FC, полученные до предобработки и после предобработки контекста. Формальные понятия, вычисленные для предобработанного контекста, корректируются следующим образом: в объемах формальных понятий, где есть объект 3, добавляется объект 5, а признак e добавляется в содержания тех формальных понятий, в которые вошел признак b. □

	a	b	c	d	e
1	1	0	1	1	0
2	0	1	1	0	1
3	1	1	1	0	1
4	0	1	0	0	1
5	1	1	1	0	1

а)

$(\varnothing, M)$
$(1, acd)$
$(3\mathbf{5}, abc\boldsymbol{e})$
$(13\mathbf{5}, ac)$
$(23\mathbf{5}, bc\boldsymbol{e})$
$(123\mathbf{5}, c)$
$(234\mathbf{5}, b\boldsymbol{e})$
$(G, \varnothing)$

	a	b	c	d
1	1	0	1	1
2	0	1	1	0
3	1	1	1	0
4	0	1	0	0

б)

$(\varnothing, M)$
$(1, acd)$
$(3, abc)$
$(13, ac)$
$(23, bc)$
$(123, c)$
$(234, b)$
$(G, \varnothing)$

Рисунок 2.7 — а: до предобработки; б: после предобработки

2.6 Анализ результативности разработанных алгоритмов

Для оценки результативности предложенного метода разложения бинарного контекста на фрагменты и алгоритмов FindBoxes, LatticeContext были проведены вычислительные эксперименты. Эксперименты проводились с применением программного комплекса FCACorpus, осуществляющего нахождение всех формальных понятий. Описание программного комплекса FCACorpus приведено в третьей главе диссертации. Использовались контексты, сгенерированные случайным образом. Для каждого контекста $K = (G, M, I)$ осуществлялось нахождение множества FC всех формальных понятий без разложения на фрагменты и с итеративным разложением на фрагменты.

Результаты вычислительных экспериментов приведены в таблицах 2.6–2.9, где $|G|$ — количество объектов, $|M|$ — количество признаков, $\sigma\,(G, M)$ — плотность исходного контекста $K = (G, M, I)$, $\|T\|$ — вес матрицы, соответствующей этому контексту, N — количество образованных фрагментов, $|FC|$ — число найденных формальных понятий, t — время выполнения программы. Вычислительные эксперименты выполнялись на компьютере с процессором Intel Core i7-720QM Processor (6M Cache, 1.60 GHz) и ОЗУ размером 4 ГБ.

Цель первой серии экспериментов — оценка результативности алгоритма FindBoxes при числе итераций $k = 1$ и без ограничения на плотность фрагментов. Анализировались два случая. Случай 1: в процедуре SearchChains проверка вложенности фрагментов $w_i \preceq w_j$ осуществляется по формуле (2.8). Случай 2: проверка вложенности фрагментов $w_i \preceq w_j$ выполняется с использованием ти-

пичных представителей по формуле (2.9). Результаты экспериментов представлены в таблице 2.6. Результативность алгоритма FindBoxes оценивалась по времени затраченному на поиск всех формальных понятий анализируемых контекстов.

Таблица 2.6 — Оценка эффективности процесса декомпозиции контекста

	Характеристика исходного контекста				Результаты		
	$\|G\|$	$\|M\|$	$\|\|T\|\|$	$\sigma\,(G, M)$	N	$\|FC\|$	t, мс
Без разложения на фрагменты	100	20	1000	0,5	–	4962	145125
С разложением на фрагменты (сл. 1)					883	4962	2878
С разложением на фрагменты (сл. 2)					883	4962	**2200**
Без разложения на фрагменты	200	30	2940	0,49	–	10567	794520
С разложением на фрагменты (сл. 1)					2895	10567	97906
С разложением на фрагменты (сл. 2)					2895	10567	**90908**

Из таблицы 2.6 следует, что

– значения $|FC|$ в случаях без разложения и с разложением на фрагменты полностью совпадают. Это иллюстрирует справедливость теоремы 2.1, т. е. «неискажаемость» разложения контекста на фрагменты относительно формальных понятий;
– число N фрагментов, образованных при разложении контекста неизменно не превышает величины $\|T\|$, что свидетельствует о правильности предложения 2.2;
– применение предложенного метода декомпозиции дает значительный выигрыш по времени: время выполнения программы FCACorpus при разложении контекста на фрагменты уменьшается в несколько раз;
– проверка вложенности фрагментов по (2.9) дает незначительный эффект по времени работы алгоритма FindBoxes.

Цель второй серии экспериментов — оценка числа фрагментов в зависимости от плотности исходного контекста. Оценка осуществлялась при $|G| = 100$, $|M| = 20$, $k = 1$. Проверка вложенности фрагментов $w_i \preceq w_j$ выполнялась с использованием типичных представителей по формуле (2.9). Результаты экспериментов представлены в таблице 2.7 и зависимости N от σ_K и t от σ_K отражены на рисунках 2.8, 2.9.

Таблица 2.7 — Оценка числа фрагментов и времени работы алгоритма FindBoxes в зависимости от плотности контекста

Плотность исходного контекста	Результаты	
σ_K	N	t, мс
0,1	137	11
0,2	261	108
0,3	466	506
0,4	677	1760
0,5	826	3234
0,6	828	6262
0,7	883	10974
0,8	402	14438
0,9	478	15004

Из таблицы 2.7 следует, что чем выше плотность исходного контекста, тем больше времени требуется на однократное разложение контекста. Таким образом, метод декомпозиции дает значительный выигрыш по времени, если плотность σ_K исходного контекста будет не высокой.

Цель третьей серии экспериментов — исследовать формальный контекст вида $K = (G, G, \neq)$ и оценить для него число возможных итераций процесса декомпозиции и время, необходимое для реализации этого процесса. Данный контекст в АФП принято рассматривать в качестве худшего случая для алгоритмов нахождения всех формальных понятий, поскольку число формальных понятий в данном случае экспоненциально зависит от размера контекста $K = (G, G, \neq)$.

Обозначим через $|G| = n$. Тогда число формальных понятий исходного контекста $K = (G, G, \neq)$ будет составлять

$$|FC| = 2^n - 2.$$

Вес 0,1-матрицы T, представляющей данный контекст, равно $\|T\| = n(n-1)$. Отсюда плотность контекста $K = (G, G, \neq)$ равна

$$0 < \sigma_K = \frac{\|T\|}{|G| \cdot |M|} = \frac{n(n-1)}{n} = 1 - \frac{1}{n} < 1.$$

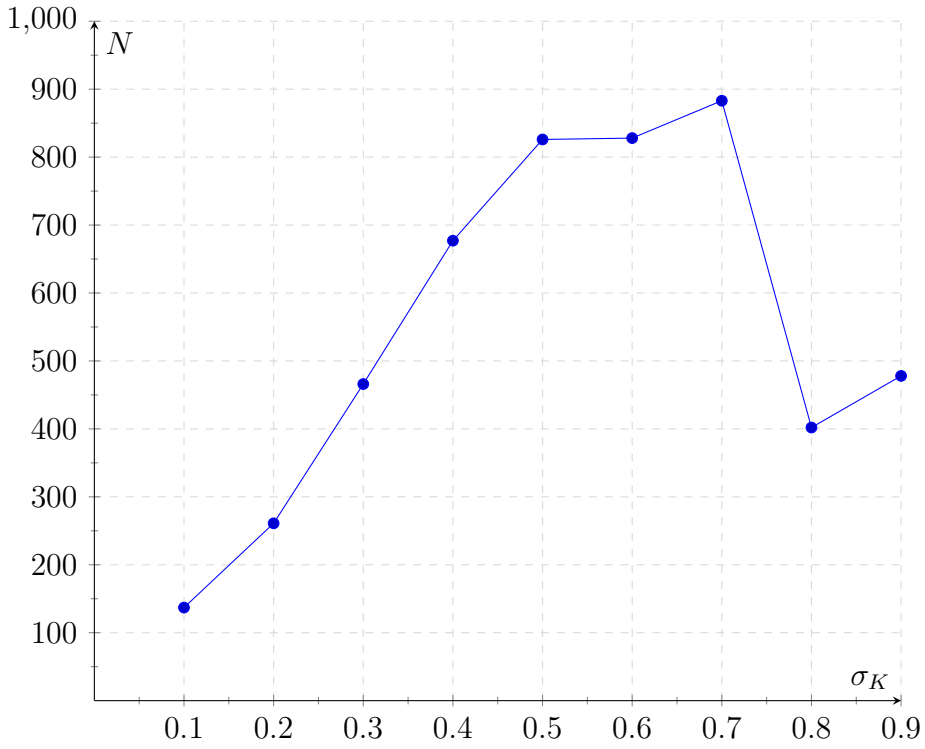


Рисунок 2.8 — Зависимость числа фрагментов N от плотности контекста σ_K

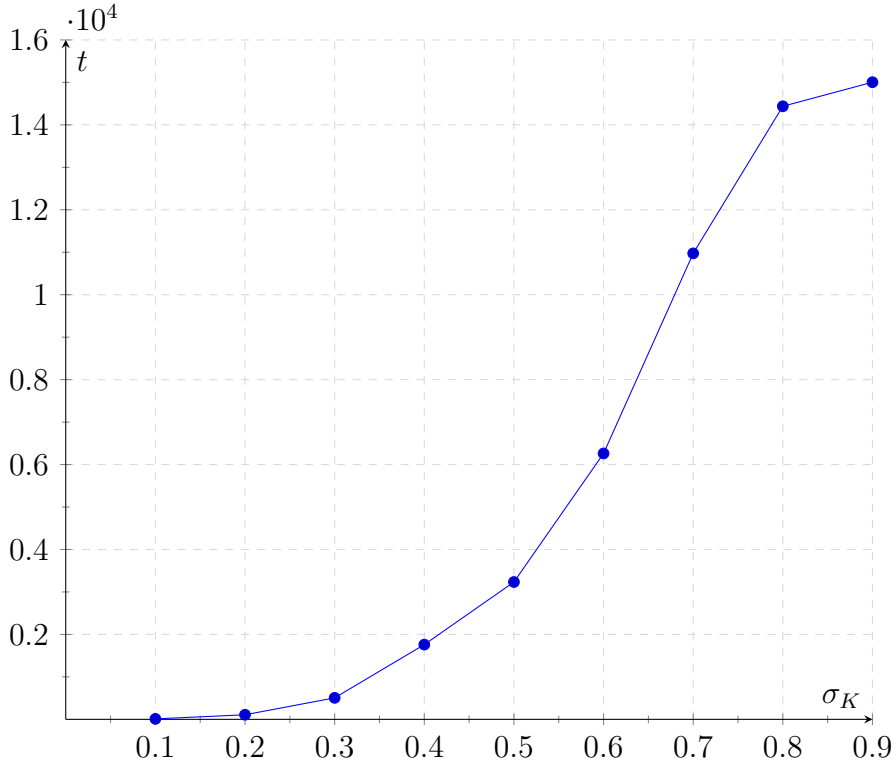


Рисунок 2.9 — Зависимость времени работы t от плотности контекста σ_K

Согласно предложению 2.2 число фрагментов, возникающих при однократном разложении контекста $K = (G, G, \neq)$ будет составлять

$$N = \|T\| = n(n-1). \tag{2.16}$$

С учетом оценки (2.10) время, необходимое на однократное разложение контекста $K = (G, G, \neq)$ на фрагменты, составляет

$$t(n) = O\left(\sigma_K \cdot |G|^2 \cdot |M|^2\right) = O\left(\sigma_K \cdot n^4\right) = O\left(n^4\right). \qquad (2.17)$$

Оценки (2.16), (2.17) являются теоретическими. Для экспериментальной оценки числа k рассмотрим контекст $K = (G, G, \neq)$ при $|G| = n = 7$. В этом случае $\sigma_K = 0,857$, $|FC| = 126$, число фрагментов при однократном разложении будет равна $N = 42$.

Результаты вычислительных экспериментов для $k = 1, \ldots, 6$ приведены в таблице 2.8.

Таблица 2.8 — Оценка числа фрагментов и времени работы алгоритма FindBoxes для контекста $K = (G, G, \neq)$ в зависимости от числа итераций

k	1	2	3	4	5	6
N	42	210	490	630	434	126
t, мс	1	75	360	640	1820	2430

Из таблицы 2.8 видно, что увеличение числа итераций приводит к увеличению числа частей, подлежащих дальнейшему разложению. На шестой итерации, т. е. когда $k = n - 1$, каждый из 126 сформированных фрагментов вырождается в формальное понятие. Зависимости N от k и t от k отражены на графиках, представленных на рисунках 2.10 и 2.11. Из этих графиков следует, что при больших значениях k число фрагментов и время разложения формального контекста $K = (G, G, \neq)$ быстро растет и $k = n - 1$ сопоставимо с числом $|FC| = 2^n - 2 = O(2^n)$. Комбинаторный взрыв числа фрагментов, возникающий при $k \ll n/2$, объясняется тем, что в этой ситуации возникает большое число неравных и не вложенных между собой фрагментов. Поэтому значение k следует выбирать малым и, конечно, значительно меньше, чем $k \ll n/2$.

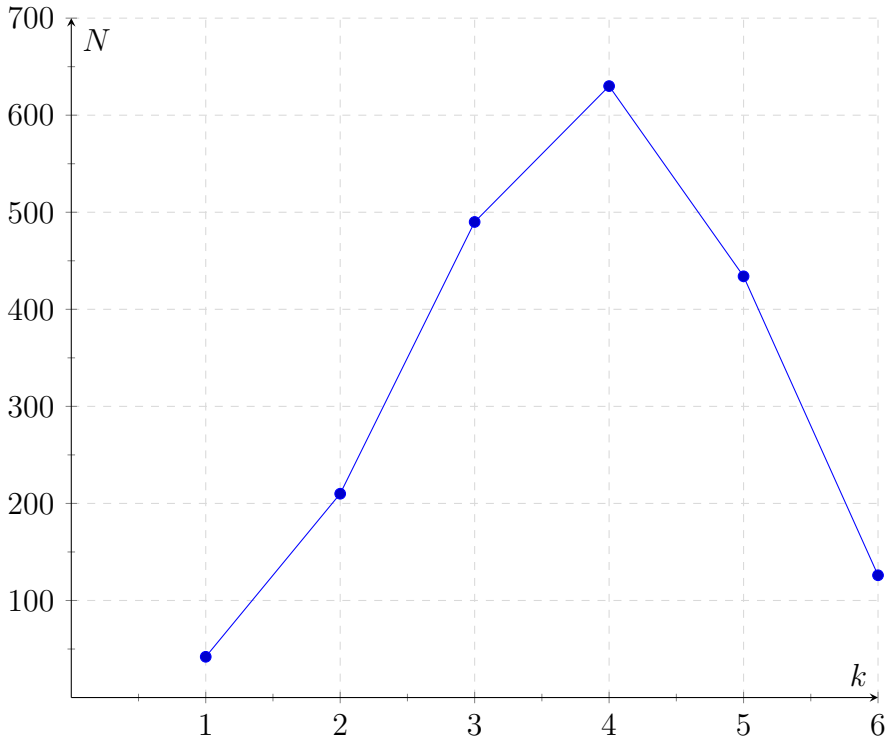


Рисунок 2.10 — Зависимость числа фрагментов N от количества итераций k

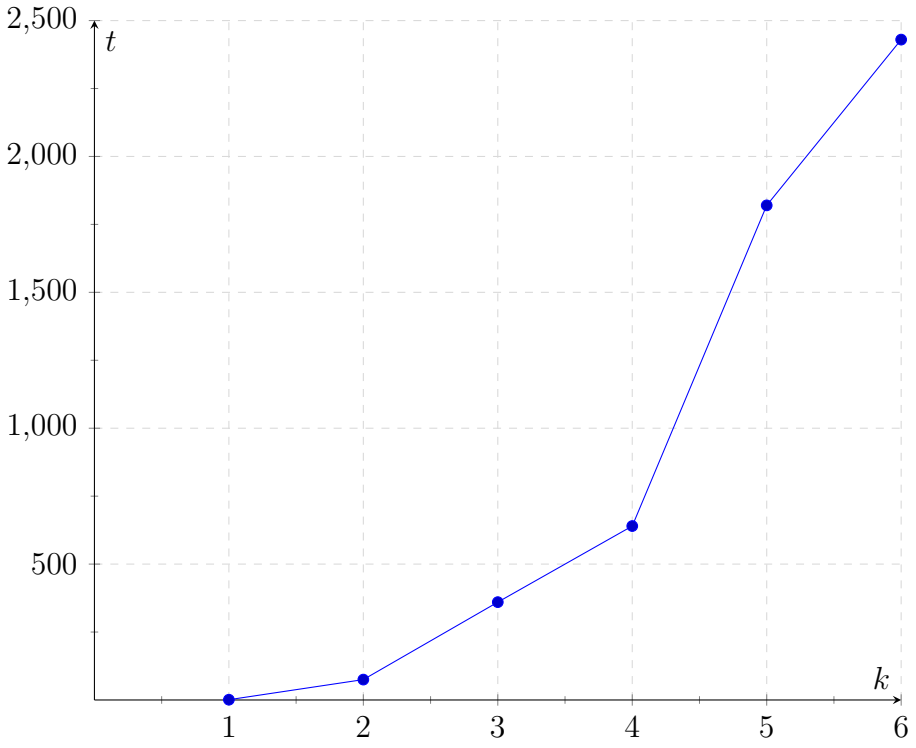


Рисунок 2.11 — Зависимость времени работы t от количества итераций k

Цель четвертой серии экспериментов — оценить для формального контекста $K = (G, G, \neq)$ возможное пороговое значение σ_0 на плотность фрагментов, по которому можно останавливать процесс декомпозиции, при больших значе-

ниях k (или то же самое, когда значение k не задано). Оценка осуществлялось при $|G| = 7$, $\sigma_K = 0,857$, $k = n - 1$ и $\sigma_0 = 1; 0,9; 0,87; 0,865; 0,86$. Результаты экспериментов представлены в таблице 2.9 и зависимости N от k и t от k отражены на графиках, изображенных на рисунках 2.12, 2.13. Здесь графики при количестве итераций $k = 1$, $k = 2$, $k = 3$, $k = 4$ совпадают. Поскольку число формируемых фрагментов на каждой итерации разложения до комбинаторного взрыва равны.

Из таблицы 2.9 и рисунков 2.12, 2.13 следует, что пороговое значение на плотность фрагментов, подлежащих дальнейшему разложению, необходимо выбрать из интервала $\sigma_K < \sigma_0 < 1$. Если σ_0 близко или равно единице, например, $\sigma_0 = 1$ и $\sigma_0 = 0,9$, то количество формируемых фрагментов увеличивается и соответственно время разложения формального контекста увеличивается. Если σ_0 близко к σ_K, например, $\sigma_0 = 0,86$, то формальных контекст однократно разлагается на фрагменты.

Таблица 2.9 — Оценка числа фрагментов и времени работы алгоритма FindBoxes для контекста $K = (G, G, \neq)$ в зависимости от порогового значения σ_0

$\sigma_0 = 1$						
k	1	2	3	4	5	6
N	42	210	490	630	434	126
t, мс	1	75	360	640	1820	2430
$\sigma_0 = 0,9$						
k	1	2	3	4	5	6
N	42	210	490	630	714	
t, мс	1	75	350	940	1620	
$\sigma_0 = 0,87$						
k	1	2	3	4	5	6
N	42	210	490			
t, мс	1	60	380			
$\sigma_0 = 0,865$						
k	1	2	3	4	5	6
N	42	210				
t, мс	1	60				
$\sigma_0 = 0,86$						
k	1	2	3	4	5	6
N	42					
t, мс	3					

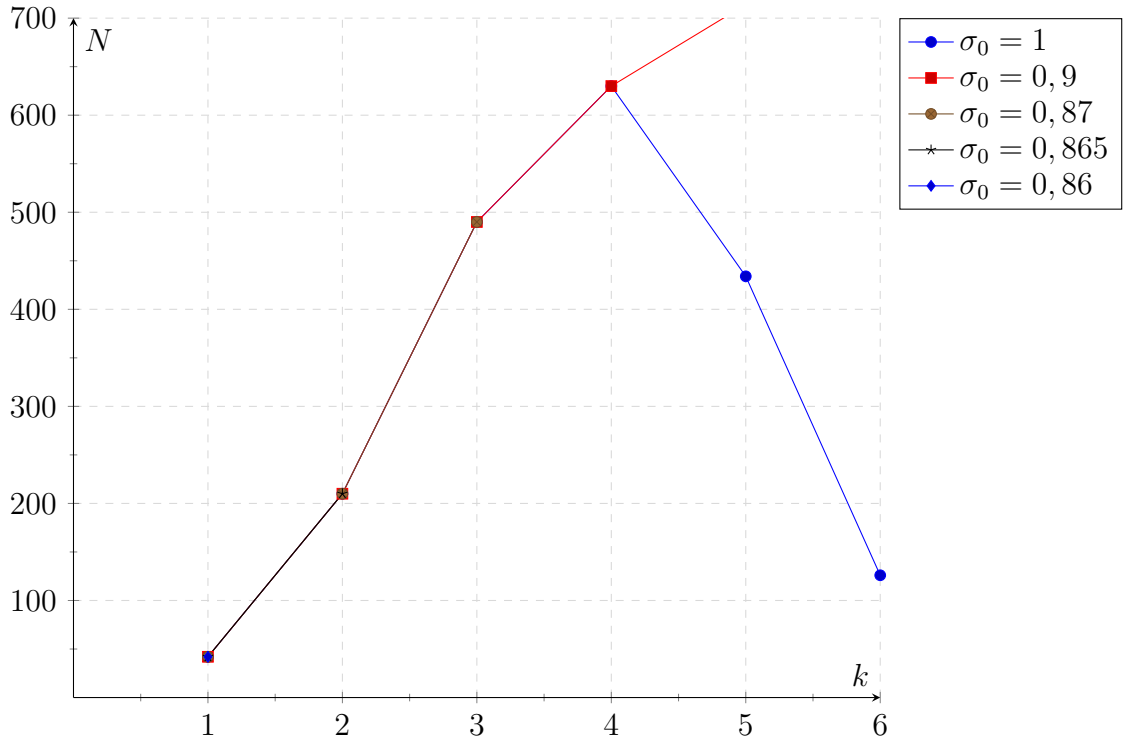


Рисунок 2.12 — Зависимость числа фрагментов N от количества итераций k

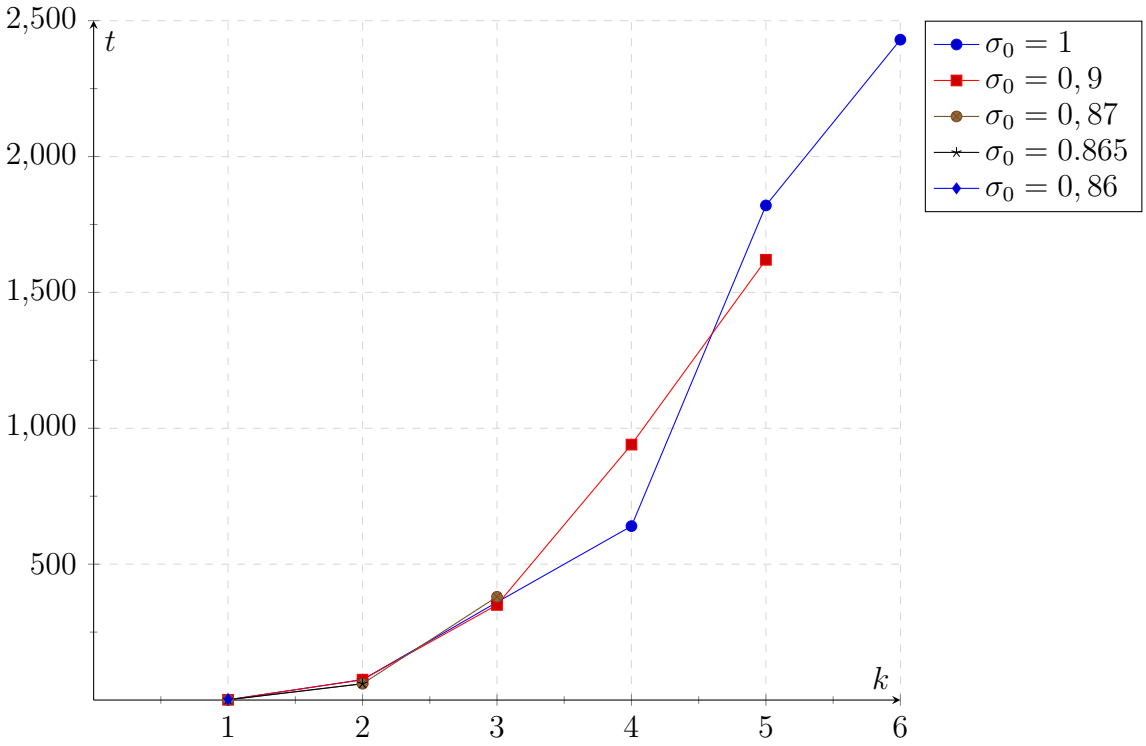


Рисунок 2.13 — Зависимость времени работы t от количества итераций k

2.7 Выводы по главе 2

1. Определено и доказано правило «неискажающего» разложения на фрагменты формального контекста, представленного 0,1-матрицей (теорема 2.1). Данное правило гласит: разложение на фрагменты является «неискажающим», т. е. сохраняет все формальные понятия исходного контекста и не порождает новых формальных понятий, если разложение выполнено путем сравнения объектных и признаковых формальных понятий и выявления фрагментов этого контекста. На теореме 2.1 базируется корректность предлагаемого метода декомпозиции.

2. Показано, что время формирования одного фрагмента для формального контекста $K = (G, M, I)$ составляет $O(|G| \cdot |M|)$. В целом время необходимое на однократное разложение этого контекста на фрагменты в худшем случае составляет $O\left(\sigma(G, M) \cdot |G|^2 \cdot |M|^2\right)$, где $\sigma(G, M)$ — плотность исходного формального контекста. Данное время следует из полученной оценки числа различных фрагментов, формируемых на каждой итерации декомпозиции (предложения 2.1, 2.2). Согласно этим предложениям число различных фрагментов не превышает числа единичных элементов разлагаемой 0,1-матрицы.

3. Доказано, что каждое формальное понятие фрагмента, образованного элементами g и m, обязательно содержит эти элементы, а также их замыкания, т. е. g'' и m'' (предложение 2.3, 2.6). Это позволяет рассматривать g'' и m'' в качестве типичного представителя данного фрагмента и входящих в него формальных понятий. Переход от фрагментов к их типичным представителям в большинстве случаев уменьшает на практике время выполнения алгоритмов нахождения всех формальных понятий для заданного формального контекста.

4. Установлены правила остановки процесса декомпозиции контекста на фрагменты, гарантирующего полиномиальное время выполнения всего процесса декомпозиции (предложения 2.4, 2.5): задание порогового значения σ_0 на плотность фрагментов и задание (фиксированного) числа k итераций разложения.

5. Разработан алгоритм FindBoxes формирования системы фрагментов, который реализует предложенный метод декомпозиции формального контекста без потери искомых формальных понятий. Процесс разложения исходного контек-

ста на фрагменты алгоритмом FindBoxes при числе итераций k выполняется за время $O\left(|G|^{2k} \cdot |M|^{2k}\right)$. Если k фиксировано, то время выполнения алгоритма FindBoxes полиномиальное относительно размера исходного формального контекста. А если $k = 1$, то алгоритм FindBoxes выполняется за время $O\left(|G|^2 \cdot |M|^2\right)$, что не противоречит пункту 2 выводов по главе 2.

6. Разработан алгоритм LatticeContext восстанавления искомого решения исходя из решений, полученных для подзадач. Вычислительная сложность данного алгоритма в худшем случае составляет $O\left(p\left(|G|, |M|\right) \cdot |FC| \cdot |G|^2 \cdot |M|\right)$, где $p\left(|G|, |M|\right)$ — полином от $|G|$ и $|M|$.

7. Созданы алгоритмы реализации запросов на извлечение знаний из заданной решетки формальных понятий. Алгоритмы предусматривают два вида (X, Y)-запросов: построение маршрута, содержащего в каждом узле (X, Y); установление общих и частных понятий для заданного формального понятия (X, Y). Данные алгоритмы позволяют решать прикладные задачи, связанные с классификацией и кластеризацией данных, выявлением зависимостей между данными. Вычислительная сложность этих алгоритмов сопоставима с размером заданной решетки.

8. Почти все разработанные в диссертации алгоритмы имеют высокую вычислительную сложность. Однако на практике при удачном задании значений k и σ_0 возможно построение полиномиального числа $|\Omega| = p\left(|G|, |M|\right)$ фрагментов — небольших по размеру частей исходного контекста. Проведенные вычислительные эксперименты показали, что применение предложенного метода декомпозиции существенно повышает производительность известных алгоритмов нахождения всех формальных понятий заданного контекста. Эксперименты подтверждают, что увеличение числа итераций приводит к увеличению числа частей, подлежащих дальнейшему разложению, а в свою очередь к увеличению времени выполнения алгоритма. Поэтому количество итераций разложения k рекомендуется задавать значительно меньше, чем $k \ll n/2$, где $n = |G|$, а пороговое значение выбрать из интервала $\sigma_K < \sigma_0 < 1$.

9. Для уменьшения числа анализируемых фрагментов в диссертации предусмотрены следующие приемы: удаление кратных и вложенных фрагментов пу-

тем частично-упорядочивания предварительно полученной системы фрагментов (следствие 2.1); применение типичных представителей фрагментов и предобработки исходного контекста без потери формальных понятий (предложение 2.6).

10. Поскольку рассматриваемая задача эквивалентна задачам определения всех максимально полных подматриц 0,1-матрицы и перечислению всех максимальных биклик двудольного графа, то предложенный метод декомпозиции может применяться при решении этих задач.

Глава 3 Программные средства и результаты их применения при исследовании коллекции «Тувинские героические сказания»

В третьей главе разработанные в диссертации метод «неискажающей» декомпозиции формального контекста, алгоритм формирования для заданного контекста системы фрагментов, алгоритм восстановления решетки формальных понятий, алгоритмы реализации запросов на извлечение знаний из заданной решетки формальных понятий и процедуры предобработки формального контекста без потери формальных понятий реализованы в виде модулей комплекса программ FCACorpus. Цель создания FCACorpus — проведение экспериментальных исследований на реальных данных, оценки результативности предложенных средств и включение FCACorpus в корпус тувинского языка. В качестве реальных данных взята электронная коллекция «Тувинские героические сказания».

В третьей главе представлена четвертая задача диссертационного исследования. Основные результаты этой главы опубликованы в работах [47, 49–51, 53, 55–61]. В 3.1 приводится модульная структура комплекса программ FCACorpus. Далее в 3.2 рассматриваются организация базы данных «Тувинские героические сказания» и описание специального модуля IInterface, обеспечивающегося информационный интерфейс между базой данных и комплексом программ FCACorpus. В 3.3 решается прикладная задача по установлению авторского стиля сказителей тувинского героического эпоса с помощью FCACorpus.

3.1 Описание программных средств

Комплекс программ FCACorpus реализует следующие разработанные в диссертационной работе алгоритмы:

– алгоритмы предобработки исходного контекста;
– алгоритмы разложения контекста на фрагменты с возможностью сокращения числа результирующих фрагментов (FindBoxes, Boxes, ShearchChian);
– алгоритмы построения решетки формальных понятий для заданной системы фрагментов (LatticeBox, LatticeContext);

– алгоритмы реализации запросов на извлечение знаний из решетки формальных понятий (Query1, Query2).

Данные алгоритмы подробно описаны в главе 2. Комплекс программ FCACorpus реализован на языке программирования C# в интегрированной среде разработки Microsoft Visual Studio Community 2017. Для его эксплуатации требуется персональный компьютер типа IBM PC Pentium IV с операционной системой Windows XP/Vista/7/8 и оперативной памятью от 512 Мб.

Входные данные комплекса программ FCACorpus включают формальный контекст $K = (G, M, I)$, а также другие данные в зависимости от выбранного режима работы этого комплекса. Допускается ввод контекста из внешнего текстового файла. Результатом работы FCACorpus являются построенная решетка L формальных понятий контекста $K = (G, M, I)$ и результаты запроса, если он был задан во входных данных.

Комплекс программ FCACorpus имеет модульную структуру и включает следующие основные модули:

– модуль Modes выбора режима работы;
– модуль Options выбора варианта нахождения множества FC;
– модуль BuildingLattice построения решетки формальных понятий L;
– модуль Queries реализации запросов на извлечение знаний из заданной решетки L.

Общая структура FCACorpus приведена на рисунке 3.1.

Модуль Modes осуществляет выбор режима работы комплекса программ FCACorpus. В модуле Modes предусмотрены два режима работы: тестовый, рабочий. Тестовый режим позволяет пользователю создавать формальный контекст путем ввода данных вручную или генерации случайным образом, изменять ранее созданный формальный контекст. Тестовый режим предназначен преимущественно для обучения и тестирования. Рабочий режим определен для решения конкретных задач, например, филологических и лингвистических задач, связанных с исследованием тувинского фольклора: распознавание индивидуального авторского стиля сказителей, выявление лексических и диалектных особенностей произведений тувинского эпоса. В этом случае исходные данные вводятся

из внешнего файла, формируемого на основе базы данных исследуемого набора текстов специальным модулем IInterface. Для ввода и редактирования формального контекста в модуле Modes предусмотрены соответствующие процедуры, функции которых приведены в таблице 3.1.

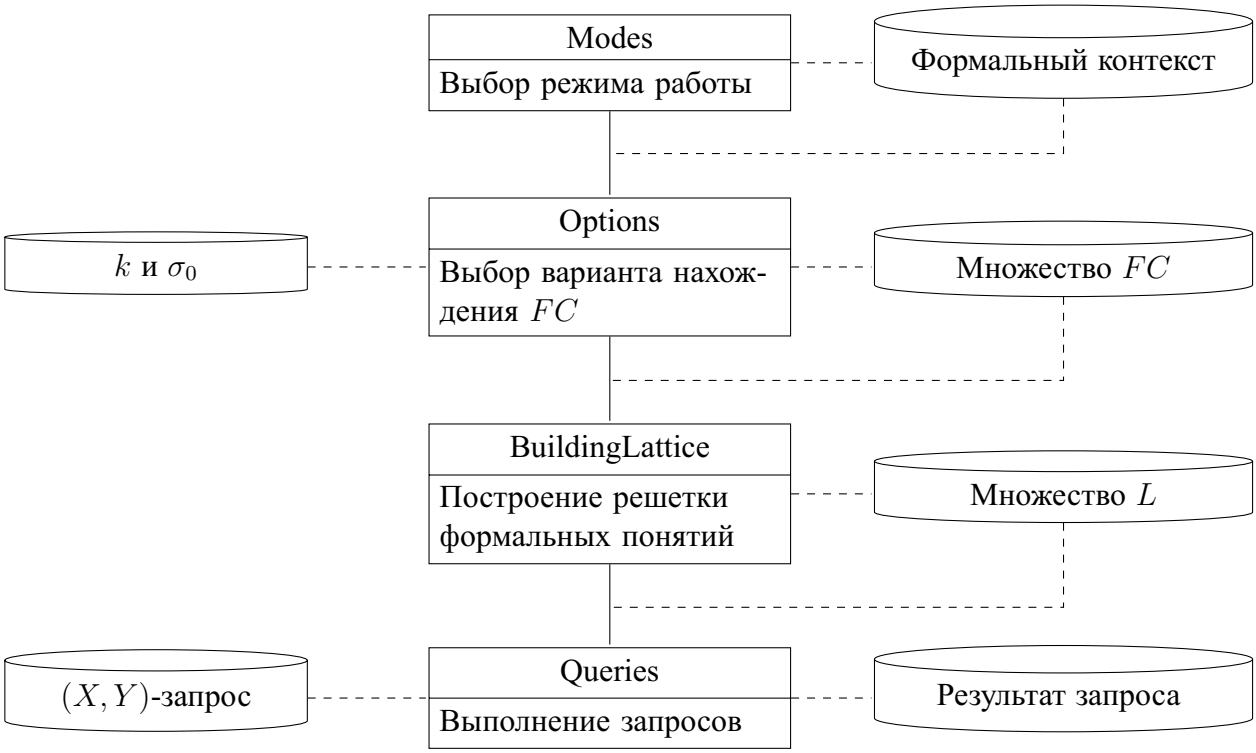


Рисунок 3.1 — Структура FCACorpus

Таблица 3.1 — Функциональное назначение процедур модуля Modes

Название процедуры	Назначение процедуры
ManualDataEntry	Создает формальный контекст $K = (G, M, I)$ путем ввода данных вручную
OpenFromFile	Вводит формальный контекст $K = (G, M, I)$ из внешнего текстового файла
GenerateRandomly	Формирует формальный контекст $K = (G, M, I)$ путем генерации случайным образом с выбором плотности контекста
EditingContext	Редактирует ранее созданный контекст (добавление, удаление объектов и признаков, сохранение и очистка контекста)

Модуль Options реализует алгоритмы FindBoxes, Boxes, ShearchChian формирования системы фрагментов и все три случая предобработки формального контекста $K = (G, M, I)$. Данный модуль осуществляет выбор варианта нахождения множества FC всех формальных понятий контекста $K = (G, M, I)$. В модуле Options предусмотрены следующие варианты: без предобработки кон-

текста, с предобработкой контекста, без разложения контекста на фрагменты, с однократным разложением контекста на фрагменты, с многократным разложением контекста на фрагменты. При многократном разложении контекста на фрагменты указываются число итераций k и пороговое значение σ_0 на плотность фрагментов. С целью снижения времени формирования множества FC в комплексе программ FCACorpus всегда выполняется сравнение числа объектов и признаков в $K = (G, M, I)$ и настройка на минимальное число. В таблице 3.2 описаны основные процедуры модуля Options, реализующие указанные выше функции.

Таблица 3.2 — Функциональное назначение процедур модуля Options

Название процедуры	Назначение процедуры
Preprocessing	Осуществляет предобработку формального контекста $K = (G, M, I)$
WithDecomposition	Однократно разлагает формальный контекст $K = (G, M, I)$ на фрагменты с применением предложенного метода декомпозиции
IterationDecomposition	Многократно разлагает формальный контекст $K = (G, M, I)$ на фрагменты. Указывается число итераций k и пороговое значение σ_0 на плотность фрагментов
AddFC	Добавляет во множество формальных понятий FC удаленные во время предобработки объекты и признаки в зависимости от конкретного случая
FindFC	Формирует множество FC всех формальных понятий для каждого фрагмента. Выполняется сравнение числа объектов и признаков в $K = (G, M, I)$ и настройка на минимальное из этих чисел

Модуль BuildingLattice выполняет алгоритм LatticeBox, осуществляющий построение решетки для заданной системы фрагментов. Модуль BuildingLattice позволяет сохранять полученную решетку в текстовом файле для последующего использования с целью более эффективного поиска необходимых формальных понятий в этой решетке, ее визуализации на экране компьютера. Основные процедуры модуля BuildingLattice приведены в таблице 3.3.

Модуль Queries реализует алгоритмы Query1, Query2, LatticeContext для выполнения запросов на извлечение знаний из решетки. Допускается объединение тех решеток, которые необходимы пользователю для решения конкретной задачи. Осуществляется построение и визуализация подрешетки L_X в виде списка формальных понятий. В таблице 3.4 приведены основные процедуры модуля Queries.

Таблица 3.3 — Функциональное назначение процедур модуля BuildingLattice

Название процедуры	Назначение процедуры
LatticeBox	Строит решетку формальных понятий
SaveLattice	Сохраняет созданную решетку формальных понятий в текстовый файл в виде списка
VisualizationLattice	Визуализирует решетку формальных понятий в виде списка на экран компьютера

Таблица 3.4 — Функциональное назначение процедур модуля Queries

Название процедуры	Назначение процедуры
LatticeContext	Осуществляет объединение выбранного состава решеток. Состав фрагментов определяется исходя из (X, Y)-запроса
Query1	Выполняет (X, Y)-запрос на выявление формальных понятий, имеющих X в объеме и Y в содержании, и строит маршрут в заданной решетке
Query2	Реализует (X, Y)-запрос на установление в заданной решетке как общих, так и частных понятий для заданного формального понятия $(X, Y) \in FC$ и строит решетку по данному запросу

Интерфейс комплекса программ FCACorpus представлен на рисунке 3.2.

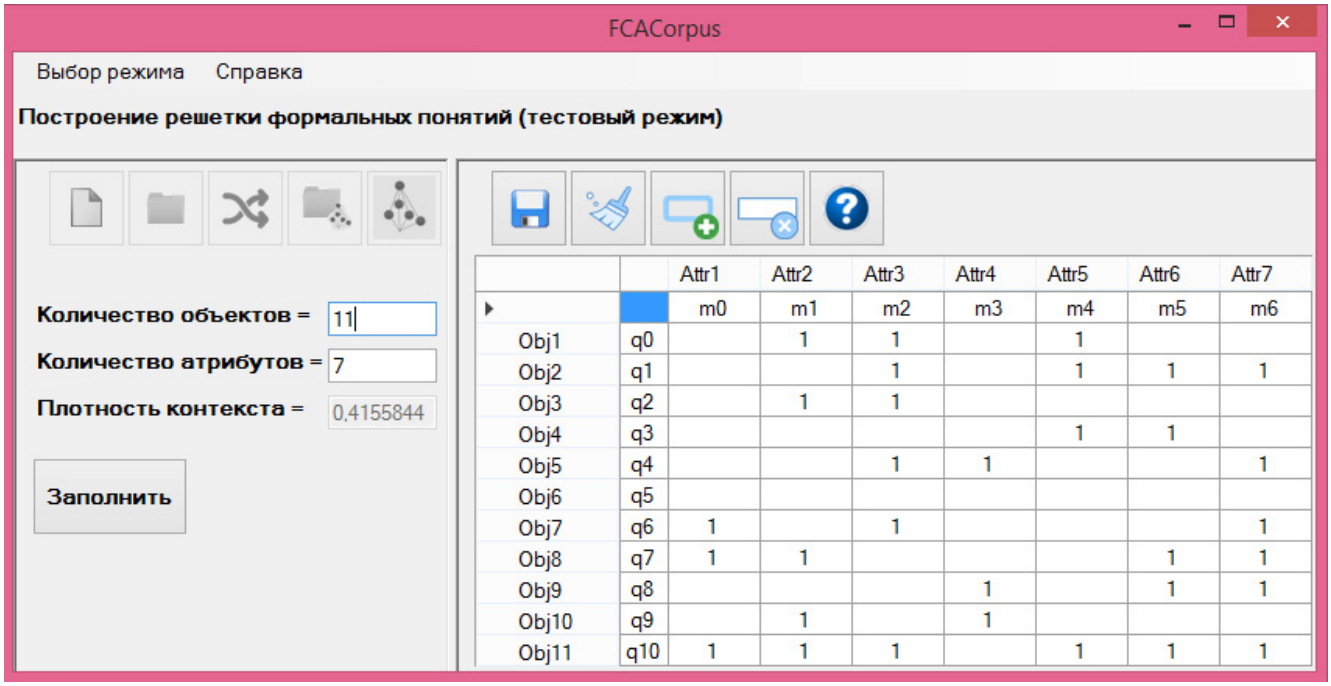


Рисунок 3.2 — Интерфейс FCACorpus

Рассмотренные выше модули комплекса программ FCACorpus носят универсальный характер и не привязаны каким-либо конкретным базам данных, на основе которых формируются исходные формальные контексты. Для привязки к

конкретной предметной области требуются база данных исследуемой предметной области и специальный модуль, обеспечивающий информационный интерфейс между базой данных и FCACorpus (рисунок 3.3). Кроме того, необходимы специальные модули, реализующие конкретные прикладные задачи. Указанные средства для электронной коллекции «Тувинские героические сказания», хранимой в корпусе тувинского языка, описаны далее.

Рисунок 3.3 — Интерфейс FCACorpus

3.2 Структура базы данных, описание информационного интерфейса

Электронная коллекция «Тувинские героические сказания» — информационная составляющая корпуса тувинского языка, разработанного преподавателями и аспирантами Тувинского государственного университета под руководством профессора М.В. Бавуу-Сюрюн [3,4]. В этой коллекции хранятся оцифрованные тексты более 50 произведений и их метаописания, включая сведения о сказителях, представленные в виде объектно-признаковой таблицы. Данная коллекция представляется базой данных «Тувинские героические сказания», созданной с применением СУБД Microsoft Office Access 2007. Структура базы данных «Тувинские героические сказания» приведена на рисунке 3.4, где list1 — метаописания героических эпосов, list2 — информация о сказителях, list3 — список клише и языковых стандартов, употребляемых в текстах тувинского героического эпоса.

Формирование и редактирование формального контекста на основе базы данных «Тувинские героические сказания» осуществляет специальный модуль IInterface [58]. Именно с использованием модуля IInterface выполняется информационная привязка всех основных модулей комплекса программ FCACorpus к корпусу тувинского языка. Общая схема работы модуля IInterface показана на рисунке 3.5, а функциональное назначение основных его процедур приведено в таблице 3.5.

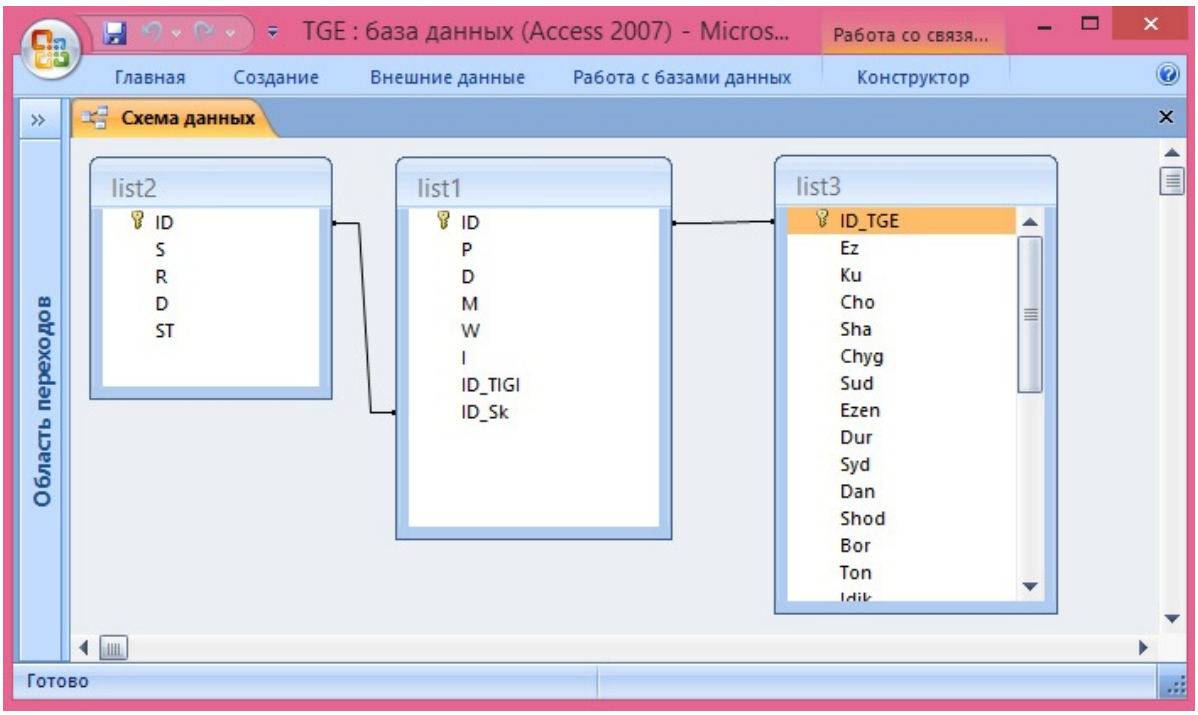


Рисунок 3.4 — Структура базы данных «Тувинские героические сказания»

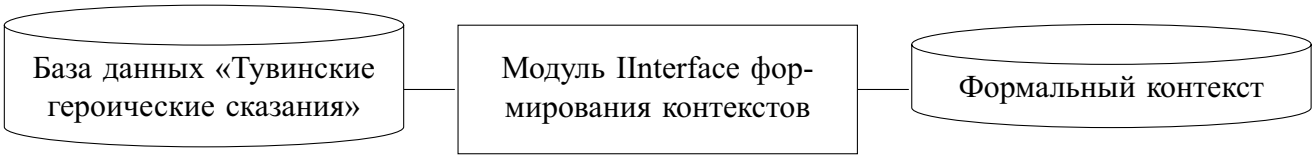


Рисунок 3.5 — Схема работы модуля IInterface

Таблица 3.5 — Функциональное назначение процедур модуля IInterface

Название процедуры	Назначение процедуры
EditingRecord	Просмотр, редактирование, добавление и удаление записей в базу данных
BinaryContext	Формирование формального контекста на основе базы данных «Тувинские героические сказания», включая шкалирование качественных признаков

Модуль IInterface допускает редактирование ранее построенного контекста, а также просмотр текстов анализируемых произведений электронной коллекции «Тувинские героические сказания». Интерфейс модуля IInterface представлен на рисунке 3.6.

В модуле IInterface имеется интерфейс для формирования контекстов (рисунок 3.7). Эксперт в зависимости от поставленной задачи из базы данных «Тувинские героические сказания» выбирает анализируемые произведения, отбирает сказителей и языковые стандарты. Затем выполняется шкалирование ка-

чественных признаков. На рисунке 3.8 показан пример приведения исходного контекста к бинарному виду. Сформированный бинарный контекст сохраняется в текстовом файле и применяется в качестве входных данных FCACorpus при решении лингвистических и филологических задач.

Номер записи	Название произведения	Год записи	Место записи	Кем записан эпос
6	Ээр-Сарыг аъттыг Экер-оол		Сүт-Хөл, с. Суг-Аксы	Самозапись
31	Эртине-Мерген			Куулар Оскал-оол
1	Эрелзей-Мерген, Харагалза...			К. Дамдин
16	Эрелзей-Мерген, Харагалза...		Таңды	Дамдын
45	Элестей ашак			Дарыма О.К.
27	Шеегүн-Бора аъттыг Шеегүн...	1963	Мөңгүн-Тайга	
28	Хунан-Кара	1959	Мөнгүн-Тайга Алда...	О.К-Ч. Дарыма Д.С. Куулар
21	Хөөкүй-кара	1951		Кызыл-оол Түлүш
5	Хартыга-Бора аъттыг Чаңг...		Тожу	
24	Хан-Шилги аъттыг Хан-Хүлүк	1961	Мөңгүн-Тайга	Бегзи Каадыр-оол Монгуш...
42	Хан-Шилги аъттыг Хан-Күчү...	1958		Дарыма О.К.
19	Хаан-Төгүлдүр	1950		С.А. Сарыг-оол, Ш.Ч. Сат

Рисунок 3.6 — Основной интерфейс модуля IInterface

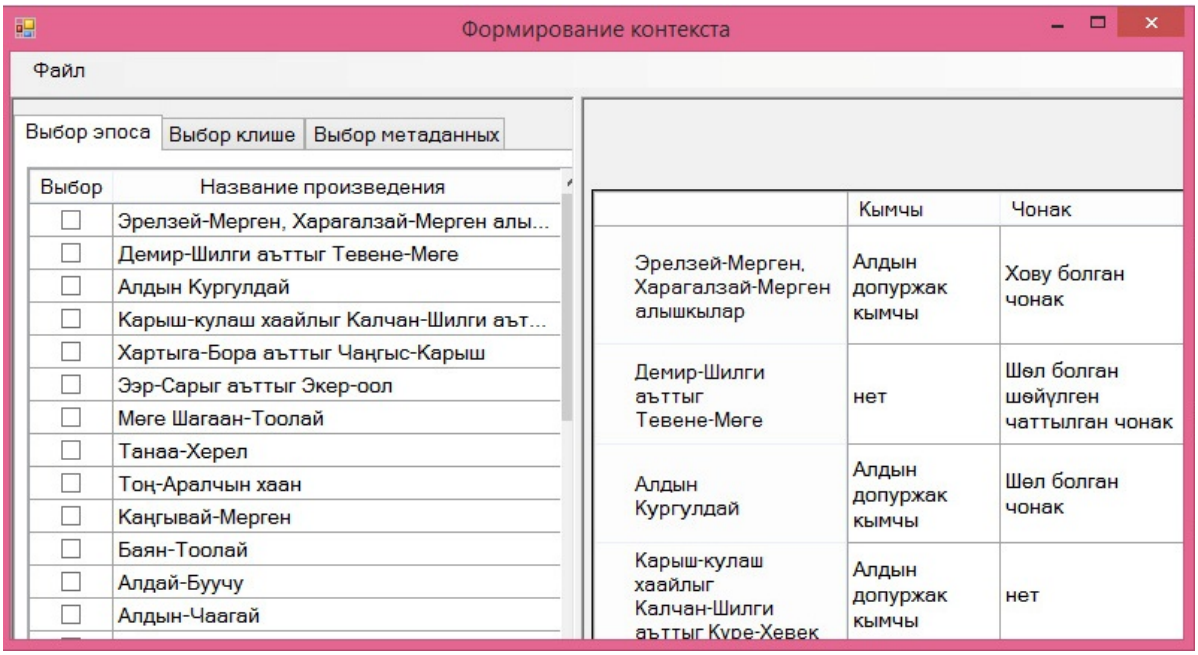
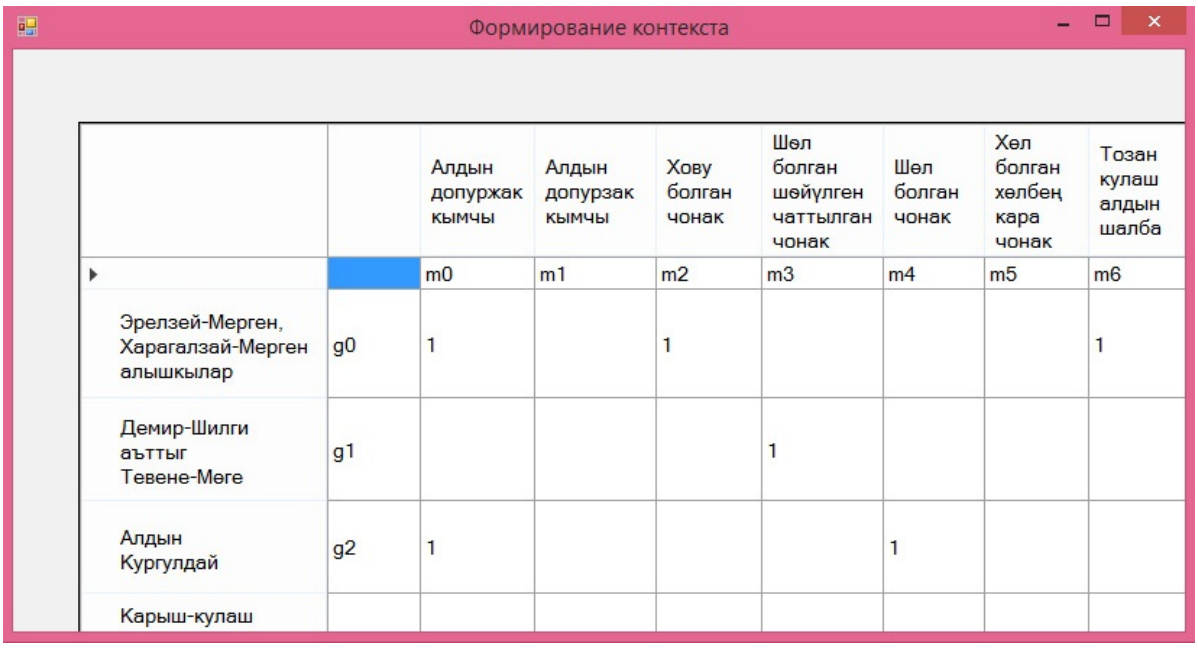


Рисунок 3.7 — Создание формального контекста: до шкалирования



		Алдын допуржак кымчы	Алдын допурзак кымчы	Хову болган чонак	Шөл болган шөйүлген чаттылган чонак	Шөл болган чонак	Хөл болган хөлбең кара чонак	Тозан кулаш алдын шалба
		m0	m1	m2	m3	m4	m5	m6
Эрелзей-Мерген, Харагалзай-Мерген алышкылар	g0	1		1				1
Демир-Шилги аттыг Тевене-Меге	g1				1			
Алдын Кургулдай	g2	1				1		
Карыш-кулаш								

Рисунок 3.8 — Создание формального контекста: после шкалирования

3.3 Установление авторского стиля сказителей

В настоящее время интенсивно развиваются различные направления языкознания. В стилистике как разделе языкознания исследуются выразительные средства языков, зафиксированные в текстах. Различают стилистику языка, стилистику речи и стилистику художественной литературы [71]. Стилистика языка связана с исследованием фактов стилистической окрашенности языковых средств, функциональных стилей языка, их взаимосвязи и взаимозависимости. Стилистика речи изучает виды жанрово-ситуативных стилей речи. Стилистика художественной литературы направлена на выявление специфики языка отдельного писателя или группы писателей, объединенных в определенную литературную школу [38]. Анализ стилистики тувинского эпоса чрезвычайно важен для выявления и сохранения историко-культурных и этнографических знаний о прошлом тувинского народа. Основными лингвистическими задачами стилистического анализа в корпусе тувинского языка являются распознавание индивидуального авторского стиля сказителей, выявление лексических и диалектных особенностей произведений эпоса. Распознавание может осуществляться на основе языковых средств, характерных для сказителя и отражающих его социальный статус, мировоззрение, творчество в той или иной период времени.

В качестве примера рассмотрим задачу установления авторских особенностей описания снаряжений коня в произведениях тувинского героического эпоса. Для решения этой задачи на основе базы данных «Тувинские героические сказания» модулем IInterface был сформирован формальный контекст $K = (G, M, I)$, который приведен в таблице 3.6. Анализировалось 14 произведений пяти сказителей тувинского героического эпоса. Списки индентификаторов произведений и сказителей приведены в таблицах 3.7 и 3.8. Каждая строка таблицы 3.6 — это характеризация отдельного произведения с использованием следующего набора признаков: сказитель, район проживания сказителя, языковые стандарты, используемые сказителем при описании снаряжения коня богатыря произведения. В качестве основных особенностей доспехов коня взяты следующие элементы снаряжения: седло, хлыст, потник, аркан, узда, лассо, стремена. Языковые стандарты для этих элементов снаряжения приведены в таблице 3.9. В качестве запросов (X, Y) были взяты сказители из таблицы 3.8.

Таблица 3.6 — Исходный контекст $K = (G, M, I)$ произведений тувинского героического эпоса

Эпос	Сказитель	Район проживания	Языковые стандарты, характеризующие снаряжение						
			седло	хлыст	потник	аркан	узда	лассо	стремена
q_0	m_{43}	m_{48}	m_0		m_{17}	m_{25} m_{26}	m_{28}	m_{32}	m_{39}
q_1	m_{44}	m_{49}	m_1 m_2	m_{10}	m_{18}		m_{29}	m_{33}	m_{40}
q_2	m_{44}	m_{49}	m_1 m_3		m_{18}		m_{29}	m_{34} m_{35}	
q_3	m_{45}	m_{50}	m_4	m_{11}			m_{29}		
q_4	m_{45}	m_{50}	m_5	m_{11}	m_{19} m_{20}		m_{29}		
q_5	m_{46}	m_{51}		m_{12}	m_{21}	m_{25}		m_{36}	
q_6	m_{46}	m_{51}						m_{36}	
q_7	m_{43}	m_{48}		m_{13}	m_{22}	m_{25}	m_{29}	m_{37}	m_{41}
q_8	m_{47}	m_{49}	m_6 m_7	m_{11}			m_{29} m_{30}	m_{38}	
q_9	m_{47}	m_{49}		m_{11}		m_{25}			m_{42}
q_{10}	m_{47}	m_{49}	m_2 m_6 m_8	m_{16}	m_{23}		m_{29} m_{31}		
q_{11}	m_{43}	m_{48}	m_9		m_{24}	m_{27}	m_{29}	m_{32}	
q_{12}	m_{45}	m_{50}	m_4	m_{11}			m_{29}		
q_{13}	m_{47}	m_{49}	m_6	m_{11}			m_{29}	m_{38}	

Таблица 3.7 — Список анализируемых тувинских героических сказаний

Идентификатор произведения	Название произведения
q_0	Демир-Шилги аттыг Тевене-Моге
q_1	Моге Шагаан-Тоолай
q_2	Танаа-Херел
q_3	Кангывай-Мерген
q_4	Алдын-Чаагай
q_5	Хан-Шилги аттыг Хан-Хулук
q_6	Арзылан-Кара аттыг Чечен-Кара Моге
q_7	Арзылан-кара аттыг Хунан-Кара
q_8	Сарыг-Хемнин иштин чурттаан Тавын-Хаан
q_9	Хан-Шилги аттыг Хан-Куче-Маадыр
q_{10}	Алдын-сарыг аттыг Анчы-Кара
q_{11}	Элестей ашак
q_{12}	Кангывай Мерген
q_{13}	Сарыг-Хемнин иштин чурттаан Сарыг-Хаан

Таблица 3.8 — Список сказителей тувинского героического эпоса

Идентификатор сказителя	Фамилия имя отчество сказителя
m_{43}	Ооржак Чанчы-Хоо Чапаажыкович
m_{44}	Ондар Тевек-Кежеге
m_{45}	Тюлюш Баазанай Халдааевич
m_{46}	Салчак Дондук Дамдынович
m_{47}	Монгуш Хургул-оол Сазыг-Хунаевич

Таблица 3.9 — Языковые стандарты тувинского героического эпоса

Имя	Языковой стандарт	
m_0	Арт болган алчайган-калчайган эзер	Седло
m_1	Арт болган ангайган-кангайган кызыл чунгуу эзер	
m_2	Ынгыржак эзер	
m_3	Кангайган-кенгейген кызыл чунгуу эзер	
m_4	Арт болган алдын кангай эзер	
m_5	Алдын хангай эзер	
m_6	Арт болган ангайган-конгайган кызыл чунгуу эзер	
m_7	Алдын, монгун кудургалыг эзер	
m_8	Шулу моогун-биле буткен эзер	
m_9	Арт болган арзайган-конгайган эзер	

Таблица 3.9 — Продолжение таблицы

Имя	Языковой стандарт	
m_{10}	Алдын допурзак кымчы	Хлыст
m_{11}	Алдын допуржак кымчы	
m_{12}	Докулчак сарыг кымчы	
m_{13}	Алдын довурзак кымчы	
m_{14}	Хулер кымчы	
m_{15}	Алдын-сарыг допуржак кымчы	
m_{16}	Алдын салбак кымчы	
m_{17}	Шол болган шойулген чаттылган чонак	Потник
m_{18}	Хол болган холбен кара чонак	
m_{19}	Дорт каът ак энчек чонак	
m_{20}	Дорт каът чонак	
m_{21}	Хову болган колбайган чонак	
m_{22}	Хол болган хову-шол чонак	
m_{23}	Хову болган чонак	
m_{24}	Шат болган чонак	
m_{25}	Алдын шалба	Аркан
m_{26}	Алдан кулаш шалба	
m_{27}	Алдан кулаш алдын шалба	
m_{28}	Алдын хумуш чуген	Узда
m_{29}	Хумуш чуген	
m_{30}	Хумуш чаагай чуген	
m_{31}	Хумуш хулер чуген	
m_{32}	Алдан кулаш сыдым	Лассо
m_{33}	Алдан кулаш арылыг чаагай сыдым	
m_{34}	Алдан кулаш арылыг чараш кара сыдым	
m_{35}	Чинге кара сыдым	
m_{36}	Алдан кулаш алдын сыдым	
m_{37}	Дошкун кара сыдым	
m_{38}	Алдан кулаш сарыг сыдым	
m_{39}	Кан болат ийи эзенги	Стремена
m_{40}	Узун эзенги	
m_{41}	Мон ыяш эзенги	
m_{42}	Хола эзенги	
m_{48}	Барун-Хемчикский район	Район
m_{49}	Сут-Хольский район	
m_{50}	Улуг-Хемский район	
m_{51}	Бай-Тайгинский район	

Формальный контекст $K = (G, M, I)$, построенный программой IInterface, включает в себя 14 строк и 52 столбцов, вес матрицы $\|T\| = 86$. Применение к нему комплекса программ FCACorpus без предобработки и без разложения на фрагменты привело к множеству FC, содержащему 30 формальных понятий:

$FC = \{(\varnothing, M)$, $(q_0, m_0\, m_{17}\, m_{25}\, m_{26}\, m_{28}\, m_{32}\, m_{39}\, m_{43}\, m_{48})$,
$(q_1, m_1\, m_2\, m_{10}\, m_{18}\, m_{29}\, m_{33}\, m_{40}\, m_{44}\, m_{49})$, $(q_2, m_1\, m_3\, m_{18}\, m_{29}\, m_{34}\, m_{35}\, m_{44}\, m_{49})$,
$(q_1\, q_2, m_1\, m_{18}\, m_{29}\, m_{44}\, m_{49})$, $(q_4, m_5\, m_{11}\, m_{19}\, m_{20}\, m_{29}\, m_{45}\, m_{50})$,
$(q_5, m_{12}\, m_{21}\, m_{25}\, m_{36}\, m_{46}\, m_{51})$, $(q_5\, q_6, m_{36}\, m_{46}\, m_{51})$,
$(q_7, m_{13}\, m_{22}\, m_{25}\, m_{29}\, m_{37}\, m_{41}\, m_{43}\, m_{48})$, $(q_0\, q_7, m_{25}\, m_{43}\, m_{48})$, $(q_9, m_{11}\, m_{15}\, m_{25}\, m_{42}\, m_{47}\, m_{49})$,
$(q_0\, q_5\, q_7\, q_9, m_{25})$, $(q_{10}, m_2\, m_6\, m_8\, m_{16}\, m_{23}\, m_{29}\, m_{31}\, m_{47}\, m_{49})$, $(q_1\, q_{10}, m_2\, m_{29}\, m_{49})$,
$(q_{11}, m_9\, m_{24}\, m_{27}\, m_{29}\, m_{32}\, m_{43}\, m_{48})$, $(q_0\, q_{11}, m_{32}\, m_{43}\, m_{48})$, $(q_7\, q_{11}, m_{29}\, m_{43}\, m_{48})$,
$(q_0\, q_7\, q_{11}, m_{43}\, m_{48})$, $(q_3\, q_{12}, m_4\, m_{11}\, m_{29}\, m_{45}\, m_{50})$, $(q_3\, q_4\, q_{12}, m_{11}\, m_{29}\, m_{45}\, m_{50})$,
$(q_8\, q_{13}, m_6\, m_7\, m_{11}\, m_{14}\, m_{29}\, m_{30}\, m_{38}\, m_{47}\, m_{49})$, $(q_8\, q_9\, q_{13}, m_{11}\, m_{47}\, m_{49})$,
$(q_8\, q_{10}\, q_{13}, m_6\, m_{29}\, m_{47}\, m_{49})$, $(q_1\, q_2\, q_8\, q_{10}\, q_{13}, m_{29}\, m_{49})$, $(q_8\, q_9\, q_{10}\, q_{13}, m_{47}\, m_{49})$,
$(q_1\, q_2\, q_8\, q_9\, q_{10}\, q_{13}, m_{49})$, $(q_3\, q_4\, q_8\, q_{12}\, q_{13}, m_{11}\, m_{29})$, $(q_3\, q_4\, q_8\, q_9\, q_{12}\, q_{13},\ m_{11})$
$(q_1\, q_2\, q_3\, q_4\, q_7\, q_8\, q_{10}\, q_{11}\, q_{12}\, q_{13}, m_{29})$, $(G, \varnothing)\}$.

Применение комплекса программ FCACorpus к этому же формальному контексту с предобработкой и однократным разложением на фрагменты привело к тому же множеству FC. Однако для построения FC потребовалось 46 мс. времени, что гораздо меньше времени чем в первом случае — 955 мс. Это подтверждает результативность предобработки и предложенного декомпозиционного подхода. При выполнении предобработки формального контекста произошло удаление 2 объектов q_{12} и q_{13}, которые полностью совпадают с объектами q_3 и q_8 соответственно. При декомпозиции формального контекста было получено 25 различных фрагментов, размерность соответствующих матриц которых не превышал 10×9:

$\Omega = \{(q_1\, q_2\, q_8\, q_9\, q_{10}\, q_{13}, m_6\, m_7\, m_{11}\, m_{14}\, m_{29}\, m_{30}\, m_{38}\, m_{47}\, m_{49})$,
$(q_0\, q_7\, q_{11}, m_9\, m_{24}\, m_{27}\, m_{29}\, m_{32}\, m_{43}\, m_{48})$,
$(q_1\, q_2\, q_8\, q_9\, q_{10}\, q_{13}, m_2\, m_6\, m_8\, m_{16}\, m_{23}\, m_{29}\, m_{31}\, m_{47}\, m_{49})$,
$(q_1\, q_2\, q_3\, q_4\, q_7\, q_8\, q_{10}\, q_{11}\, q_{12}\, q_{13}, m_2\, m_6\, m_8\, m_{16}\, m_{23}\, m_{29}\, m_{31}\, m_{47}\, m_{49})$,
$(q_1\, q_2\, q_8\, q_9\, q_{10}\, q_{13}, m_{11}\, m_{15}\, m_{25}\, m_{42}\, m_{47}\, m_{49})$,
$(q_5\, q_6, m_{12}\, m_{21}\, m_{25}\, m_{36}\, m_{46}\, m_{51})$,
$(q_0\, q_5\, q_7\, q_9, m_{12}\, m_{21}\, m_{25}\, m_{36}\, m_{46}\, m_{51})$,
$(q_0\, q_7\, q_{11}, m_0\, m_{17}\, m_{25}\, m_{26}\, m_{28}\, m_{32}\, m_{39}\, m_{43}\, m_{48})$,

$(q_1\, q_2\, q_3\, q_4\, q_7\, q_8\, q_{10}\, q_{11}\, q_{12}\, q_{13}, m_6\, m_7\, m_{11}\, m_{14}\, m_{29}\, m_{30}\, m_{38}\, m_{47}\, m_{49})$,
$(q_3\, q_4\, q_8\, q_9\, q_{12}\, q_{13}, m_4\, m_{11}\, m_{29}\, m_{45}\, m_{50})$,
$(q_1\, q_2\, q_3\, q_4\, q_7\, q_8\, q_{10}\, q_{11}\, q_{12}\, q_{13}, m_4\, m_{11}\, m_{29}\, m_{45}\, m_{50})$,
$(q_1\, q_2\, q_3\, q_4\, q_7\, q_8\, q_{10}\, q_{11}\, q_{12}\, q_{13}, m_9\, m_{24}\, m_{27}\, m_{29}\, m_{32}\, m_{43}\, m_{48})$,
$(q_3\, q_4\, q_8\, q_9\, q_{12}\, q_{13}, m_5\, m_{11}\, m_{19}\, m_{20}\, m_{29}\, m_{45}\, m_{50})$,
$(q_1\, q_2\, q_3\, q_4\, q_7\, q_8\, q_{10}\, q_{11}\, q_{12}\, q_{13}, m_5\, m_{11}\, m_{19}\, m_{20}\, m_{29}\, m_{45}\, m_{50})$,
$(q_1\, q_2\, q_3\, q_4\, q_7\, q_8\, q_{10}\, q_{11}\, q_{12}\, q_{13}, m_1\, m_3\, m_{18}\, m_{29}\, m_{34}\, m_{35}\, m_{44}\, m_{49})$,
$(q_1\, q_2\, q_3\, q_4\, q_7\, q_8\, q_{10}\, q_{11}\, q_{12}\, q_{13}, m_1\, m_2\, m_{10}\, m_{18}\, m_{29}\, m_{33}\, m_{40}\, m_{44}\, m_{49})$,
$(q_1\, q_2\, q_8\, q_9\, q_{10}\, q_{13}, m_1\, m_2\, m_{10}\, m_{18}\, m_{29}\, m_{33}\, m_{40}\, m_{44}\, m_{49})$,
$(q_0\, q_5\, q_7\, q_9, m_0\, m_{17}\, m_{25}\, m_{26}\, m_{28}\, m_{32}\, m_{39}\, m_{43}\, m_{48})$,
$(q_3\, q_4\, q_8\, q_9\, q_{12}\, q_{13}, m_{11}\, m_{15}\, m_{25}\, m_{42}\, m_{47}\, m_{49})$,
$(q_0\, q_5\, q_7\, q_9, m_{13}\, m_{22}\, m_{25}\, m_{29}\, m_{37}\, m_{41}\, m_{43}\, m_{48})$,
$(q_1\, q_2\, q_8\, q_9\, q_{10}\, q_{13}, m_1\, m_3\, m_{18}\, m_{29}\, m_{34}\, m_{35}\, m_{44}\, m_{49})$,
$(q_1\, q_2\, q_3\, q_4\, q_7\, q_8\, q_{10}\, q_{11}\, q_{12}\, q_{13}, m_{13}\, m_{22}\, m_{25}\, m_{29}\, m_{37}\, m_{41}\, m_{43}\, m_{48})$,
$(q_0\, q_7\, q_{11}, m_{13}\, m_{22}\, m_{25}\, m_{29}\, m_{37}\, m_{41}\, m_{43}\, m_{48})$,
$(q_0\, q_5\, q_7\, q_9, m_{11}\, m_{15}\, m_{25}\, m_{42}\, m_{47}\, m_{49})$,
$(q_3\, q_4\, q_8\, q_9\, q_{12}\, q_{13}, m_6\, m_7\, m_{11}\, m_{14}\, m_{29}\, m_{30}\, m_{38}\, m_{47}\, m_{49})$.

Применение модуля BuildingLattice комплекса программ FCACorpus к каждому из фрагментов дало 25 решеток.

Предположим, что задан (X, Y)-запрос, где $X = \{\varnothing\}$ и $Y = \{m_{43}\}$ =«Ооржак Ч-Х.Ч.». Требуется найти все формальные понятия, содержащие в объеме X, а в содержании Y, и указать связи между этими формальными понятиями. Это означает, что необходимо найти все языковые стандарты сказителя m_{43}. При реализации данного запроса алгоритмом Query1 модуля Queries отбираются 7 фрагментов из 25:

$(q_0\, q_7\, q_{11}, m_9\, m_{24}\, m_{27}\, m_{29}\, m_{32}\, m_{43}\, m_{48})$,
$(q_0\, q_7\, q_{11}, m_0\, m_{17}\, m_{25}\, m_{26}\, m_{28}\, m_{32}\, m_{39}\, m_{43}\, m_{48})$,
$(q_1\, q_2\, q_3\, q_4\, q_7\, q_8\, q_{10}\, q_{11}\, q_{12}\, q_{13}, m_9\, m_{24}\, m_{27}\, m_{29}\, m_{32}\, m_{43}\, m_{48})$,
$(q_0\, q_5\, q_7\, q_9, m_0\, m_{17}\, m_{25}\, m_{26}\, m_{28}\, m_{32}\, m_{39}\, m_{43}\, m_{48})$,
$(q_0\, q_5\, q_7\, q_9, m_{13}\, m_{22}\, m_{25}\, m_{29}\, m_{37}\, m_{41}\, m_{43}\, m_{48})$,
$(q_1\, q_2\, q_3\, q_4\, q_7\, q_8\, q_{10}\, q_{11}\, q_{12}\, q_{13}, m_{13}\, m_{22}\, m_{25}\, m_{29}\, m_{37}\, m_{41}\, m_{43}\, m_{48})$,
$(q_0\, q_7\, q_{11}, m_{13}\, m_{22}\, m_{25}\, m_{29}\, m_{37}\, m_{41}\, m_{43}\, m_{48})$.

Затем осуществляется объединение решеток этих фрагментов и получается решетка L. Далее производится обход решетки L и выявляются формальные понятия, удовлетворяющие условию (2.13), и строится решетка L_{XY} (рисунок 3.9).

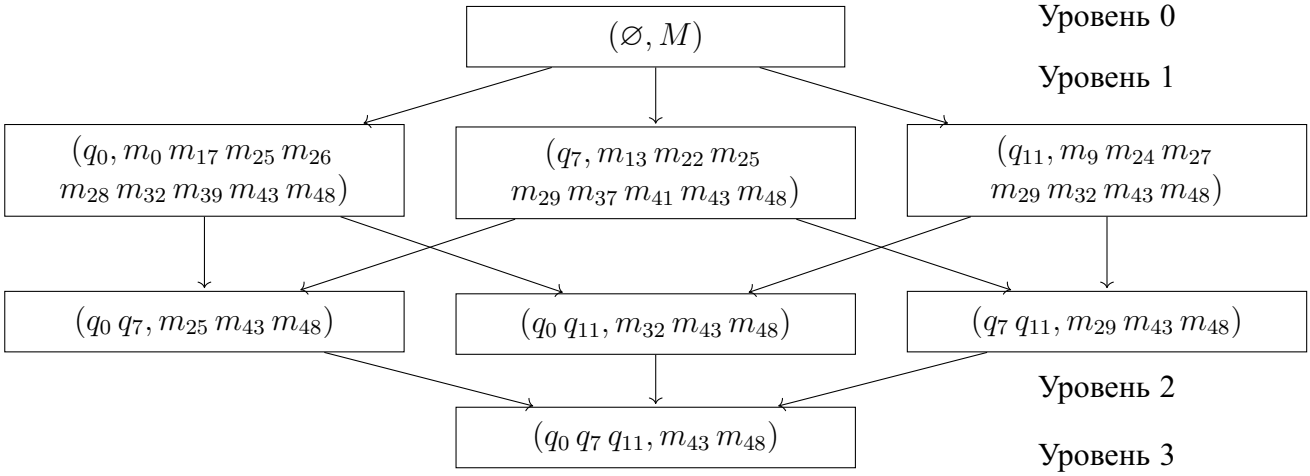


Рисунок 3.9 — Решетка L_{XY} формальных понятий для сказителя Ооржак Ч-Х.Ч.

Решетка L_{XY} характеризует авторский стиль сказителя m_{43} = «Ооржак Ч-Х.Ч.». В решетке L_{XY} формальное понятие $(q_0\, q_7\, q_{11}, m_{43}\, m_{48})$, находящееся на уровне 3, является самым общим по отношению ко всем другим формальным понятиям этой решетки. Формальное понятие $(q_0\, q_7\, q_{11}, m_{43}\, m_{48})$ означает, что сказитель m_{43} проживает в районе m_{48} и его авторский стиль выявляется в произведениях q_0 = «Демир-Шилги аъттыг Тевене-Моге», q_7 = «Арзылан-кара аъттыг Хунан-Кара» и q_{11} = «Элестей ашак». Формальные понятия, находящиеся на втором уровне решетки L_{XY}, показывают, какие языковые стандарты и в каких его произведениях употребляются:

– в произведениях q_0, q_7 при описании аркана используется языковой стандарт m_{25} = «Алдын шалба»;
– в произведениях q_0, q_{11} при описании лассо применяется языковой стандарт m_{32} = «Алдан кулаш сыдым»;
– в произведениях q_7, q_{11} при описании узды употребляется языковой стандарт m_{29} = «Хумуш чуген».

Три формальных понятия уровня 1 (их называют элементарными) описывают отдельные произведения данного сказителя со своими специфичными языковыми стандартами. Таким образом, ответом на указанный выше запрос являются: для сказителя m_{43} при описании снаряжения коня характерны языковые стандарты m_{25}, m_{32}, m_{29}.

Аналогичным образом осуществляется запрос для сказителя m_{44} = «Ондар Тевек-Кежеге». В данном случае из 25 найденных фрагментов объединению подлежат решетки лишь следующих 4 фрагментов:

$(q_1\, q_2\, q_3\, q_4\, q_7\, q_8\, q_{10}\, q_{11}\, q_{12}\, q_{13}, m_1\, m_3\, m_{18}\, m_{29}\, m_{34}\, m_{35}\, m_{44}\, m_{49})$,
$(q_1\, q_2\, q_3\, q_4\, q_7\, q_8\, q_{10}\, q_{11}\, q_{12}\, q_{13}, m_1\, m_2\, m_{10}\, m_{18}\, m_{29}\, m_{33}\, m_{40}\, m_{44}\, m_{49})$,
$(q_1\, q_2\, q_8\, q_9\, q_{10}\, q_{13}, m_1\, m_2\, m_{10}\, m_{18}\, m_{29}\, m_{33}\, m_{40}\, m_{44}\, m_{49})$,
$(q_1\, q_2\, q_8\, q_9\, q_{10}\, q_{13}, m_1\, m_3\, m_{18}\, m_{29}\, m_{34}\, m_{35}\, m_{44}\, m_{49})$.

Построенная решетка L_{XY} для данного запроса приведена на рисунке 3.10.

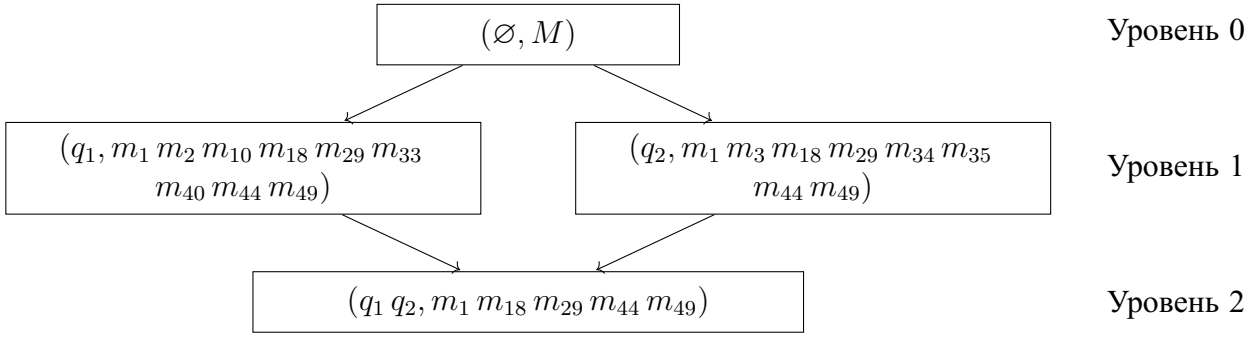


Рисунок 3.10 — Решетка L_{XY} формальных понятий запроса для сказителя Ондар Тевек-Кежеге

В этой решетке понятие $(q_1\, q_2, m_1\, m_{18}\, m_{29}\, m_{44}\, m_{49})$ уровня 2 является общим по отношению другим понятиям. Согласно этому понятию сказитель m_{44} проживает в районе m_{49} и для него характерны языковые стандарты m_1, m_{18}, m_{29}, используемые им в произведениях q_1, q_2.

Следует заметить, что сказители m_{43} и m_{44} используют языковой стандарт m_{29}. Только этот языковый стандарт является общим для данных сказителей.

Из построенных решеток можно выявлять характерные признаки не только отдельного сказителя, но и некоторых заданных произведений тувинского эпоса. Пусть $X = \{q_0\, q_{11}\}$, $Y = \{m_{32}\, m_{43}\, m_{48}\}$. Требуется в решетке L_{XY}, изображенной на рисунке 3.9, найти общие и частные понятия по отношению к формальному понятию $(q_0\, q_{11}, m_{32}\, m_{43}\, m_{48})$. Данный вид запроса реализуется алгоритмом Query2 модуля Queries путем обхода решетки L_{XY} и выявления формальных понятий, удовлетворяющим условиям (2.14), (2.15). Результат выполнения данного (X, Y)-запроса представлен на рисунке 3.11.

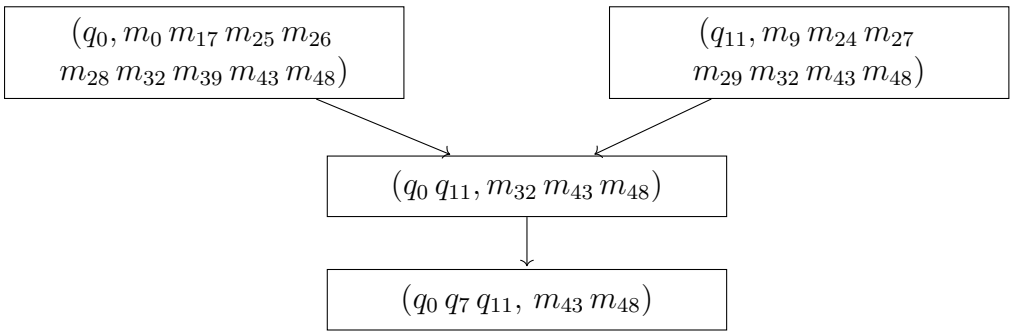


Рисунок 3.11 — Результат выполнения (X, Y)-запроса, где $X = \{q_0\, q_{11}\}$, $Y = \{m_{32}\, m_{43}\, m_{48}\}$

Таким образом, для заданного формального понятия $(q_0\, q_{11}, m_{32}\, m_{43}\, m_{48})$ общим понятием является $(q_0\, q_7\, q_{11},\ m_{43}\, m_{48})$, а частными — $(q_0, m_0\, m_{17}\, m_{25}\, m_{26}$ $m_{28}\, m_{32}\, m_{39}\, m_{43}\, m_{48})$, $(q_{11}, m_9\, m_{24}\, m_{27}\ \ m_{29}\, m_{32}\, m_{43}\, m_{48})$. Это означает, что общее понятие охватывает все произведения сказителя m_{43}, проживающего в районе m_{48}. В состав объектов частного понятия входит некоторая часть произведений из общего понятия, объединенных признаками не свойственными всему множеству произведений сказителя m_{43}. Например, для обоих произведений q_0, q_{11} характерен признак m_{32}, а только для эпоса q_0 свойственны языковые стандарты m_0, m_{17}, m_{25}, m_{26}, m_{28}, m_{39}, m_{43}, m_{48}, которые одновременно не присущи произведению q_{11}. Однако для эпоса q_{11} свойственны языковые стандарты m_9, m_{24}, m_{27}, m_{29}, m_{32}, m_{43}, m_{48}.

Экспертами установлено, что результаты выполнения этих запросов, полученных с применением разработанных в диссертации алгоритмов и программ, соответствуют действительности, т. е. являются филологически и лингвистически правильными. Комплекс программ FCACorpus позволяет эффективно решать задачу установления авторского стиля сказителей тувинского героического эпоса в рамках корпуса тувинского языка, используя базу данных «Тувинские героические сказания». Для комплексного анализа авторского стиля сказителей тувинского героического эпоса контекст $K = (G, M, I)$, представленный в таблице 3.6, необходимо расширить путем добавления в таблицу языковых стандартов, описывающих внешний вид и боевое снаряжение героя.

3.4 Выводы по главе 3

1. Созданный комплекс программ FCACorpus является универсальным и не привязан каким-либо конкретным базам данных, на основе которых формируются исходные формальные контексты. Для привязки к конкретной предметной области требуются база данных исследуемой предметной области и специальный модуль, обеспечивающий информационный интерфейс между базой данных и FCACorpus. Кроме того, необходимы специальные модули, реализующие конкретные прикладные задачи.

2. Привязка комплекса программ FCACorpus к корпусу тувинского языка выполнена с использованием разработанных модуля IInterface и базы данных «Тувинские героические сказания».

3. В рамках корпуса тувинского языка проведены экспериментальные исследования, подтверждающие результативность разработанных в диссертационной работе метода, алгоритмов и программ, при решении задачи установления авторского стиля сказителей тувинского героического эпоса. Полученные результаты показали, что разработанные средства могут быть применены не только для распознавания авторского стиля сказителей, но и для других подобных задач анализа текстов в рамках корпуса тувинского языка.

Заключение

1. Разработан и теоретически обоснован метод «неискажающего» разложения формального контекста на фрагменты (теорема 2.1). Исследована структура фрагментов и найдена оценка числа фрагментов, получаемых на каждой итерации разложения, определены правила остановки процесса разложения формального контекста на фрагменты без потери формальных понятий (предложения 2.1 – 2.6).

2. Разработаны алгоритмы формирования для заданного формального контекста системы фрагментов, восстановления искомого решения исходя из решений, полученных для подзадач, и реализации возможных запросов на извлечение знаний из решетки формальных понятий.

3. Разработаны алгоритмы предобработки формального контекста без потери формальных понятий путем удаления единичных, нулевых и кратных строк и столбцов этого контекста.

4. Создан комплекс программ, реализующий разработанные метод и алгоритмы, для проверки их результативности на случайных формальных контекстах и на реальных данных применительно к корпусу тувинского языка. Эксперименты показали, что все разработанные в диссертации алгоритмы имеют высокую вычислительную сложность. Однако на практике при удачном задании значений k и σ_0 возможно построение полиномиального числа $|\Omega| = p\left(|G|, |M|\right)$ фрагментов. Эксперименты подтверждают, что увеличение числа итераций приводит к увеличению числа фрагментов, подлежащих дальнейшему разложению, и в свою очередь к увеличению времени выполнения алгоритмов. Поэтому количество итераций разложения k рекомендуется задавать значительно меньше, чем $k \ll n/2$, где $n = |G|$, а пороговое значение выбрать из интервала $\sigma_K < \sigma_0 < 1$.

Список литературы

1. Айвазян С. А. Прикладная статистика: основы моделирования и первичная обработка данных / С. А. Айвазян, И. С. Енюков, Л. Д. Мешалкин — М.: Финансы и статистика, 1983. — 471 с.
2. Бавуу-Сюрюн М. В. Вопросы создания электронных ресурсов тувинского языка: некоторые итоги, неотложные задачи и перспективы / М. В. Бавуу-Сюрюн // Новые исследования Тувы. — 2016. — № 4.
3. Бавуу-Сюрюн М. В. Клише и стандарты в текстах тувинских героических сказаниях. Свидетельство о государственной регистрации базы данных № 2017620024 / М. В. Бавуу-Сюрюн, С. М. Далаа, Ч. М. Монгуш, М. В. Ондар. Зарегистрировано в Реестре баз данных 10 января 2017 г.
4. Бавуу-Сюрюн М. В. Тувинские героические сказания. Свидетельство о государственной регистрации базы данных № 2017620090 / М. В. Бавуу-Сюрюн, С. М. Далаа, Ч. М. Монгуш, М. В. Ондар. Зарегистрировано в Реестре баз данных 19 января 2017 г.
5. Барсегян А. А. Технологии анализа данных: Data Mining, Visual Mining, Text Mining, OLAP / А. А. Барсегян, М. С. Куприянов, В. В. Степаненко, И. И. Холод — СПб.: БВХ-Петербург, 2008. — 384 с.
6. Баскакова Л. В. Модель распознающих алгоритмов с представительными наборами и системами опорных множеств / Л. В. Баскакова, Ю. И. Журавлев // Журнал вычислительной математики и математической физики. — 1981. — Т. 21. — № 5. — С. 1264–1275.
7. Белим С. В. Использование решетки формальных понятий для построения ролевой политики разграничения доступа / С. В. Белим, Н. Ф. Богаченко // Информатика и системы управления. — 2019. — № 1(55). — С. 16–28.
8. Белим С. В. Проверка соответствий ориентированного графа алгебраической решетке / С. В. Белим, Н. Ф. Богаченко // Прикладная дискретная математика. — 2018. — № 41. — С. 54–65.
9. Биркгоф Г. Современная прикладная алгебра / Г. Биркгоф, Т. Барти — М.: Мир, 1976. — 400 с.
10. Биркгоф Г. Теория решеток / Г. Биркгоф — М.: Наука, 1984. — 568 с.

11. Богатырев М. Ю. Методы анализа формальных понятий в информационных системах технической поддержки / М. Ю. Богатырев, В. Р. Нуриахметов, В. С. Вакурин // Известия ТулГУ. Технические науки. — 2013. — Т. 2. — С. 25–36.
12. Богатырев М. Ю. Построение концептуальных графов как элементов семантической разметки текстов / М. Ю. Богатырев, В. В. Тюхтин // Компьютерная лингвистика и интеллектуальные технологии — 2009. — Вып. 8. — № 15. — С. 31–37.
13. Быкова В. В. Алгоритмы концептуального моделирования и классификации текстов в корпусе тувинского языка / В. В. Быкова, Ч. М. Монгуш // Программные продукты и системы. — 2017. — Т. 30. — № 3. — С. 487–495.
14. Быкова В. В. Декомпозиционный подход к исследованию формальных контекстов / В. В. Быкова, Ч. М. Монгуш // Прикладная дискретная математика. — 2019. — № 44. — С. 113–126.
15. Быкова В. В. Математические методы анализа рекурсивных алгоритмов / В. В. Быкова // Журнал Сибирского федерального университета. Математика и физика. — 2008. — Т. 3. — № 1. — С. 372–384.
16. Воронцов К. В. Аддитивная регуляризация тематических моделей коллекции текстовых документов / К. В. Воронцов // Доклады РАН. — 2014. — Т. 456. — № 3. — С. 268–271.
17. Воронцов К. В. Обзор современных исследований по проблеме качества обучения алгоритмов / К. В. Воронцов // Таврический вестник информатики и математики. — 2004. — № 1. — С. 5–24.
18. Воронцов К. В. Регуляризация, робастность и разреженность вероятностных тематических моделей / К. В. Воронцов, А. А. Потапенко // Компьютерные исследования и моделирование. — 2012. — Т. 4. — № 4. — С. 693–706.
19. Гретцер Г. Общая теория решеток / Г. Гретцер — М.: Мир, 1982. — 456 с.
20. Гуров С. И. Булевы алгебры, упорядоченные множества, решетки: определения, свойства, примеры / С. И. Гуров — М.: Либроком, 2013. — 221 с.
21. Гуров С. И. Классификация на основе АФП и бикластеризации: возможности подхода / С. И. Гуров, А. А. Онищенко // Прикладная математика и ин-

форматика: Труды факультета Вычислительной математики и кибернетики. — 2011. — Т. 38. — С. 77–87.

22. Гэри М. Вычислительные машины и труднорешаемые задачи : пер. с англ. / М. Гэри, Д. Джонсон — М.: Мир, 1982. — 416 с.
23. Добров Б. В. Онтологии и тезаурусы: модели, инструменты, приложения / Б. В. Добров,В. В. Иванов, Н. В. Лукашевич, В. Д. Соловьев — М.: Бином. Лаборатория знаний, 2009. — 178 с.
24. Дугинов О. И. Сложность задач покрытия графа наименьшим числом полных двудольных графов / О. И. Дугинов // Труды Института математики. — 2014. — Т. 22. — № 1. — С. 51–69.
25. Дюкова Е. В. Дискретный анализ признаковых описаний в задачах распознавания большой размерности / Е. В. Дюкова, Ю. И. Журавлев // Журнал вычислительной математики и математической физики. — 2000. — Т. 40. — № 8. — С. 1264–1278.
26. Дюкова Е. В. О процедурах классификации, основанных на построении покрытий классов / Е. В. Дюкова, А. С. Инякин // Журнал вычислительной математики и математической физики. — 2003. — Т. 43. — № 12. — С. 1884–1895.
27. Евтушенко С. А. Система анализа данных CONCEPT EXPLORER / С. А. Евтушенко // Труды VII Национальной конференции по искусственному интеллекту. — М.: Физматлит, 2000. — С. 127–134.
28. Емеличев В. А. Лекции по теории графов / В. А. Емеличев, О. И. Мельников, В. И. Сарванов, Р. И. Тышкевич — М.: Наука. Главная редакция физико-математической литературы, 1990. — 384 с.
29. Журавлев Ю. И. Алгоритмы распознавания, основанные на вычислении оценок / Ю. И. Журавлев // Кибернетика. — 1971. — № 3. — С. 1–11.
30. Журавлев Ю. И. О математических принципах классификации предметов и явлений / Ю. И. Журавлев // Дикретный анализ. — 1966. — № 7. — С. 3–15.
31. Журавлев Ю. И. Об алгоритмах распознавания с представительными наборами (о логических алгоритмах) / Ю. И. Журавлев // Журнал вычислительной математики и математической физики. — 2002. — Т. 42. — № 9. — С. 1425–1435.

32. Загоруйко Н. Г. Прикладные методы анализа данных и знаний / Н. Г. Загоруйко — Новосибирск: ИМ СО РАН, 1999. — 270 с.
33. Захаров В. П. Корпусная лингвистика / В. П. Захаров — СПб.: БВХ-Петербург, 2005. — 48 с.
34. Игнатов Д. И. Модели, алгоритмы и программные средства бикластеризации на основе замкнутых множеств: автореф. дисс. ... канд. техн. наук: 05.13.18. — М., 2010. 26 с.
35. Игнатов Д. И. О поиске сходства Интернет-документов с помощью частых замкнутых множеств признаков / Д. И. Игнатов, С О. Кузнецов // Труды 10-й национальной конференции по искусственному интеллекту с международным участием. — 2006. — Т. 2. — С. 249–258.
36. Ильвовский Д. А. Выявление дубликатов объектов в прикладных онтологиях с помощью метода анализа формальных понятий / Д. А. Ильвовский, М. А. Климушкин // Научно-техническая информация. Информационные процессы и системы. — 2013. — № 1. — С. 10–19.
37. Ильвовский Д. А. Системы автоматической обработки текстов / Д. А. Ильвовский, Е. Л. Черняк // Открытые системы. СУБД. — 2014. — № 1. — С. 51–53.
38. Карелова О. В. К вопросу изучения индивидуального стиля автора / О. В. Карелова // Известия Российского государственного педагогического университета им. А.И. Герцена. — 2006. — Т. 20. — № 3. — С. 24–29.
39. Качалов Д. Л. Исследование технологий сбора и обработки больших данных в крупномасштабных экономических системах / Д. Л. Качалов, М. П. Фархадов // Известия Волгоградского государственного технического университета. — 2017. — № 15(210). — С. 94–98.
40. Клещев А. С. Математические модели онтологий предметных областей. Часть 1. Существующие подходы к определению понятия «онтология» / А. С. Клещев, И. Л. Артемьева // Научно-техническая информация, серия 2 «Информационные процессы и системы» — 2001. — №. 2. — С. 20–27.
41. Колесникова С. И. Оценка значимости признаков для тестов в интеллектуальных системах / С. И. Колесникова, А. Е. Янковская // Известие РАН. Тео-

рия и системы управления. — 2008. — № 2. — С. 135–148.

42. Коршунов А. В. Тематическое моделирование текстов на естественном языке / А. В. Коршунов, А. Г. Гомзин // Труды института системного программирования РАН. — 2012. — Т. 23. — С. 215–244.
43. Крейнес М. Г. Модели текстов и текстовых коллекций для поиска и анализа информации / М. Г. Крейнес // Труды Московского физико-технического института. — 2017. — Т. 9. — № 3(35). — С. 132–142.
44. Кузнецов С. О. Автоматическое обучение на основе анализа формальных понятий / С. О. Кузнецов // Автоматика и телемеханика. — 2001. — № 10. — С. 3–27.
45. Кузнецов С. О. Быстрый алгоритм построения всех пересечений объектов из конечной полурешетки / С. О. Кузнецов // Научно-техническая информация. — 1993. — № 1. — С. 17–20.
46. Кузнецов С. О. ДСМ-метод как система аксиоматического обучения / С. О. Кузнецов // Интеллектуальные информационные системы. — 1991. — № 15. — С. 17–53.
47. Монгуш Ч. М. Алгебраический подход исследования текстов тувинского фольклора / Ч. М Монгуш // Материалы VI Международной конференции «Математика, ее приложения и математическое образование (МПМО2017)». — Улан-Удэ: Изд-во ВСГУТУ, 2017. — С. 277–281.
48. Монгуш Ч. М. Алгоритм «безопасной» декомпозиции формального контекста / Ч. М Монгуш // Прикладная дискретная математика. Приложение (труды Всероссийской конференции «Компьютерная безопасность и криптография»). — 2019. — № 12. — С. 227–232.
49. Монгуш Ч. М. Анализ слабоструктурированных текстов на тувинском языке / Ч. М Монгуш // Материалы III Международной научно-практической конференции молодых ученых, аспирантов и студентов «Актуальные проблемы исследования этноэкологических и этнокультурных традиций народов Саяно-Алтая». — Кызыл: Изд-во ТувГУ, 2015. — С. 86–87.
50. Монгуш Ч. М. Анализ формальных понятий при выявлении клише в тувинских сказаниях / Ч. М Монгуш // Материалы Всероссийской научно-

практической конференции «Информатизация образования: история, проблемы и перспективы». — Кызыл: Изд-во ТувГУ, 2016. — С. 16–19.

51. Монгуш Ч. М. База данных и средства создания контекстов для представления и анализа тувинского героического эпоса / Ч. М. Монгуш, М. В. Ондар // Программные продукты, системы и алгоритмы. — 2017. — № 3. — С. 1–6.
52. Монгуш Ч. М. Метатекстовая разметка в Национальном корпусе тувинского языка: структура и функциональные возможности / Ч. М. Монгуш // Новые исследования Тувы. — 2016 — № 4. — С. 1–8.
53. Монгуш Ч. М. Методы анализа формальных понятий в исследовании текстов тувинского фольклора / Ч. М Монгуш, В. В. Быкова // Материалы XV Международной конференции имени А. Ф. Терпугова «Информационные технологии и математическое моделирование» (ИТММ–2016). — Томск: Изд-во Том. ун-та, 2016. — Ч. 2. — С. 153–158.
54. Монгуш Ч. М. О «неискажающем» разложении бинарного контекста в анализе данных и комбинаторной оптимизации / Ч. М Монгуш, Д. В. Семенова // Материалы VII Международной конференции «Знания – Онтологии – Теории». — Новосибирск: Изд-во Института математики им. С. Л. Соболева СО РАН, НГУ, 2019. — С. 394–395.
55. Монгуш Ч. М. О классификации произведений тувинского фольклора и распознавании жанра героического эпоса / Ч. М Монгуш // Материалы XVII Международной конференции имени А. Ф. Терпугова «Информационные технологии и математическое моделирование» (ИТММ–2018). — Томск: Изд-во НТЛ, 2018. — С. 257–263.
56. Монгуш Ч. М. Обзор методов классификации и кластеризации текстов / Ч. М Монгуш // Материалы Всероссийской научно-практической конференции «Информатизация образования: история, проблемы и перспективы». — Кызыл: Изд-во ТувГУ, 2016. — С. 19–21.
57. Монгуш Ч. М. Программа формирования контекста для электронной коллекции «Тувинские героические сказания» / Ч. М. Монгуш // Инженерный вестник Дона. — 2018. — № 2(49). — С. 119–128.

58. Монгуш Ч. М. Программа формирования контекстов в корпусе тувинского языка. Свидетельство о государственной регистрации программы для ЭВМ № 2018618908 / Ч. М. Монгуш. Зарегистрировано в Реестре программ для ЭВМ 23 июля 2018 г.
59. Монгуш Ч. М. Программа FCACorpus концептуального моделирования тувинских текстов методами анализа формальных понятий. Свидетельство о государственной регистрации программы для ЭВМ № 2018618907. / Ч. М. Монгуш, В. В. Быкова Зарегистрировано в Реестре программ для ЭВМ 23 июля 2018 г.
60. Монгуш Ч. М. Распознавание жанра произведений тувинского фольклора на основе анализа формальных понятий / Ч. М Монгуш, В. В. Быкова // Труды Международной конференции «Актуальные проблемы прикладной математики и информационных технологий — Аль-Хорезми 2016». — Бухара: Изд-во Национ. ун-та., 2016. — С. 202–205.
61. Монгуш Ч. М. Распознавание индивидуального авторского стиля сказителей тувинского героического эпоса / Ч. М. Монгуш // Экономика и менеджмент систем управления. — 2018. — Т. 29. — № 3.1. — С. 184–194.
62. Монгуш Ч. М. Электронный корпусный словарь тувинского языка / Ч. М. Монгуш, А. С. Дагбажык // Программные продукты, системы и алгоритмы. — 2017. — № 2. — С. 1–6.
63. Найденова К. А. Машинное обучение в задачах обработки естественного языка: обзор современного состояния исследований / К. А. Найденова, О. А. Невзорова // Ученые записки Казанского государственного университета. Серия: Физико-математические науки. — 2008. — Т. 150. — № 4. — С. 5–24.
64. Новиков В. Е. Свойства решётки концептов однозначного минимального контекста / В. Е. Новиков // Сборник научных трудов. Математика. Механика. — Саратов: Изд-во Саратовского университета, 2018. — Вып. 20. — С. 53–55.
65. Пальчунов Д. Е. Разработка автоматизированных методов порождения служебных документов на естественном языке / Д. Е. Пальчунов, А. А. Финк // Вестник НГУ. Серия: Информационные технологии. — 2017. — № 3(15). — С. 79–89.

66. Панкратова И. А. Условия реализуемости функций на полурешётках устойчивыми к состязаниям схемами / И. А. Панкратова // Известия Саратовского университета. Новая серия. Серия: Математика. Механика. Информатика. — 2008. — Т. 8. — № 1. — С. 55–58.
67. Парватов Н. Г. Соответствие Галуа для замкнутых классов дискретных функций / Н. Г. Парватов // Прикладная дискретная математика. — 2010. — № 2(8). — С. 10–15.
68. Парватов Н. Г. Проблема выразимости в решетке с замыканием / Н. Г. Парватов // Дискретная математика. — 2010. — № 22(4). — С. 83–103.
69. Поддубный В. В. О возможности математического моделирования эволюции полисемии знаков естественного языка с помощью нестационарных процессов рождения и гибели / В. В. Поддубный // Вестник Томского государственного университета. УВТиИ. — 2016. — № 3(36). — С. 49–59.
70. Поддубный В. В. Сравнительный анализ эффективности распознавания авторского стиля текстов деревом решений и модифицированным наивным байесовским классификатором / В. В. Поддубный, А. И. Кубарев, К. А. Михалёва // Известия вузов. Физика. — 2015. — Т 58. — № 11(2). — С. 252–258.
71. Романова Н. Н. Стилистика и стили / Н. Н. Романова, А. В. Филиппов — М.: МАКС Пресс, 2012. — 416 с.
72. Савчук С. О. Метатекстовая разметка в национальном корпусе русского языка: базовые принципы и основные функции / С. О. Савчук // Национальный корпус русского языка: 2003—2005. Результаты и перспективы. — 2005. — С. 62–88.
73. Салчак А. Я. Электронный корпус тувинского языка: состояние, проблемы / А. Я. Салчак, А. В. Байыр-оол // Мир науки, культуры, образование. — 2013. — № 6. — С. 408–409.
74. Скороходько Э. Ф. Семантические сети и автоматическая обработка текста / Э. Ф. Скороходько — Киев: Наукова думка, 1983. — 218 с.
75. Смирнов С. В. Построение онтологий предметных областей со структурными отношениями на основе анализа формальных понятий / С. В Смирнов //

Труды Всероссийской конференции с международным участием «Знания–Онтологии–Теории». — Новосибирск: Институт математики СО РАН, 2011. — Т. 2. — С. 103–112.

76. Соловьев С. Ю. Подходы к исследованию формальных контекстов / С. Ю. Соловьев, Д. Е. Стельмашенко // Информационные процессы. — 2011. — Т. 11. — № 2. — С. 277–290.
77. Соловьев С. Ю. Применение принципов экспертной классификации для анализа формальных понятий / С. Ю. Соловьев, Д. Е. Стельмашенко // Бизнес-информатика. — 2013. — Т. 26. — № 4. — С. 53–57.
78. Стельмашенко Д. Е. Свойства формальных контекстов / Д. Е. Стельмашенко // Информационные процессы. — 2011. — № 1. — С. 86–89.
79. Aslanyan L. Comparative analysis of attack graphs / L. Aslanyan, D. Alipour, M. Heidari // Mathematical Problems of Computer Science. — 2013. — № 40. — P. 85–95.
80. Baklouti F. A fast and general algorithm for Galois lattices building / F. Baklouti, G. Levy, R. Emilion // The Electronic Journal of Symbolic Data Analysis. — 2005. — Vol. 2. — № 1. — P. 19–31.
81. Becker P. ToscanaJ: an open source tool for qualitative data analysis / P. Becker, J. Hereth, G. Stumme // 15th European Conference on Artificial intelligence : proceedings. — Lyon, France, 2002. — P. 1–2.
82. Bein D. Clustering and the Biclique Partition Problem / D. Bein, L. Morales, W. Bein, C. O. Shields, Jr., Z. Meng, I. H. Sudborough // 41th Annual Hawaii International Conference on System Sciences : proceedings. Big Island, Hawaii. — Big Island, Hawaii, 2008. — P. 475–483.
83. Belohlavek R. Discovery of optimal factors in binary data via a novel method of matrix decomposition / R. Belohlavek, V. Vychodil // Journal of Computer and System Sciences. — 2010. — Vol. 76. — № 1. — P. 3–20.
84. Bertet K. Extensions of Bordat's algorithm for attributes // 5th International Conference on Concept Lattices and their Applications : proceedings. — France: Montpellier, 2007. — P. 389.

85. Blei D. M Introduction to Probabilistic Topic Models / D. M Blei // Communications of the ACM. — 2012. — Vol. 55. — № 4. — P. 77–84.
86. Borza P. V. OpenFCA, an open source formal concept analysis toolbox / P. V. Borza, O. Sabou, C. Sacarea // International Conference on Automation Quality and Testing Robotics : proceedings. — Cluj-Napoca, Romania, 2010. —- P. 1–5.
87. Bykova V. V. On Algebraic Approach of R. Wille and B. Ganter in the Investigation of Texts / V. V. Bykova, Ch. M. Mongush // Journal of Siberian Federal University. Mathematics and Physics. — 2017. — Vol. 10. — № 3. — P. 372–384.
88. Carpineto C. Concept data analysis: theory and applications / C. Carpineto, G. Romano — New York: Wiley, 2004. — 200 p.
89. Dowling C. E. On the irredundant generation of knowledge spaces / C. E. Nourine // Journal of Mathematical Psychology. — 1993. — Vol. 37. — № 1. — P. 49–62.
90. Fu H. A parallel algorithm to generate formal concepts for large data / H. Fu, E. M. Nguifo // Lecture notes in computer science. — 2004. — Vol. 2961. — P. 394–401.
91. Ganter B. Conceptual Exploration / B. Ganter, S. A. Obiedkov — Berlin Heidelberg: Springer, 2016. — 315 p.
92. Ganter B. Formal Concept Analysis: Foundations and Applications / B. Ganter, G. Stumme, R. Wille — Berlin Heidelberg: Springer, 2005. — 315 p.
93. Ganter B. Formal Concept Analyses: Mathematical Foundations / B. Ganter, R. Wille — Springer Science and Business Media, 2012. — 314 p.
94. Ganter B. Two basic algorithms in concept analysis / B. Ganter, L. Kwuida, B. Sertkaya // 8th International Conference on Formal Concept Analysis: proceedings. — Berlin: Heidelberg, 2010. — P. 312–340.
95. Godin R. Incremental concept formation algorithms based on Galois (concept) lattices / R. Godin, R. Missaoui, H. Alaoui // Computer Intelligence. — 1995. — Vol. 11. — № 2. — P. 246–267.
96. Harzheim E. Ordered sets / E. Harzheim — New York: Springer, 2005. — 390 p.

97. Heydari M. Computing Cross Associations for Attack Graphs and other Applications / M. Heydari, L. Morales, C. O. Shields, I. H. Sudborough // 40th Annual Hawaii International Conference on System Sciences : proceedings. — Big Island: Hawaii, 2007. — P. 270.
98. Hofmann T. Latent Semantics Models for Collaborative Filtering / T. Hofmann // Transactions on Information Systems. — 2004. — Vol. 22. — № 1. — P. 89–115.
99. Kourie D. G. An incremental algorithm to construct a lattice of set intersections / D. G. Godin, S. Obiedkov, B. W. Watson, D. V. D. Merwe // Computer Programming. — 2009. — № 74. — P. 128–142.
100. Kumar N. Fast construction of concept lattice / N. Kumar, A. Gupta, V. Bhatnagar // 4th International Conference on Concept Lattices and their Applications : proceedings. — Hammamet, 2006. — P. 1–9.
101. Kuznetsov S. O. Comparing Performance of Algorithms for Generating Concept Lattices / S. O. Kuznetsov, S. A. Obiedkov // Journal of Experimental and Theoretical Artificial Intelligence. — 2002. — Vol. 14. — № 2. — P. 189–216.
102. Kuznetsov S. O. Machine Learning and Formal Concept Analysis / B. Ganter, P. W. Eklund, B. Sertkaya // 2th International Conference on Formal Concept Analysis : proceedings. — Berlin: Heidelberg, 2004. — Vol. 2961. — P. 287–312.
103. Kuznetsov S. O. Mathematical aspects of concept analysis / S. O. Kuznetsov // Journal of Mathematical Sciences. — 1996. — Vol. 80.— № 2. — P. 1654–1698.
104. Lahcen B. Lattice Miner: a tool for concept lattice construction and exploration / U. Priss // 8th International Conference on Formal concept analysis: proceedings. — Marocco: Agadir, 2010. — P. 59–66.
105. Li J. Maximal Biclique Subgraphs and Closed Pattern Pairs of the Adjacency Matrix: A One-to-one Correspondence and Mining Algorithms / J. Li, G. Liu, H. Li, L. Wong// Journal IEEE Transactions on Knowledge and Data Engineering. — 2007. — № 19. — P. 1625–1637.
106. Lindig C. Fast concept analysis / N. Kumar, A. Gupta, V. Bhatnagar // 8th International Conference on Conceptual Structures : proceedings. — Germany: Darmstadt, 2000. — P. 152–161.

107. Merwe D. V. D. AddIntent: a new incremental algorithm for constructing concept lattices / D. V. D. Kumar, S. Obiedkov, D. G. Kourier // International Conference on Formal Concept Analysis : proceedings. — Sydney, Australia: Springer, 2004. — P. 372–385.

108. Mongush Ch. M. On decomposition of a binary context without losing formal concepts / Ch. M. Mongush, V. V. Bykova // Journal of Siberian Federal University. Mathematics and Physics. — 2019. — Vol. 12. — № 3. — P. 323–330.

109. Moon J. W. On cliques in graphs / J. W. Moon, L. Moser // Journal Mathematics. — 1965. — № 3. — P. 23–28.

110. Naidenova X. A. Good Classification Tests as Formal Concepts / X. A. Naidenova // 10th International Conference on Formal Concept Analysis: proceedings. — Belgium, Leuven, 2012. — P. 211–226.

111. Neznanov A. A. FCART: A New FCA-based System for Data Analysis and Knowledge Discovery / A. A. Neznanov, D. A. Ilvovsky, S. O. Kuznetsov // 11th International Conference on Formal Concept Analysis: proceedings. — Germany: TU Dresden, 2013. — P. 31–44.

112. Nourine L. A fast algorithm for building lattices / L. Nourine, O. Raynaud // Information Processing Latters. — 1999. — № 71. — P. 199–204.

113. Poelmans J. Formal concept analysis in knowledge processing: a survey on applications / J. Poelmans, D. I. Ignatov, S. O. Kuznetsov, G. Dedene // Journal of Mathematical Sciences. — 2013. — Vol. 40.— № 16. — P. 6538-6560.

114. Poelmans J. Formal concept analysis in knowledge processing: a survey on models and techniques / J. Poelmans, D. I. Ignatov, S. O. Kuznetsov, G. Dedene // Journal of Mathematical Sciences. — 2013. — Vol. 40. — № 16. — P. 6601-6623.

115. Pottosina S. Finding maximal complete bipartite subgraphs in a graph / S. Pottosina, Y. Pottosin, B. Sedliak // Journal Applied Mathematics. — 2008. — Vol. 1. — № 1. — P. 75–81.

116. Priss U. FCAStone — FCA file format conversion and interoperability software / U. Priss // Conceptual Structures Tool Interoperability Workshop. — 2008. — P. 33–43.

117. Priss U. Formal Concept Analysis in Information Science / U. Priss // Annual Review of Information Science and Technology. — 2006. — № 40(1). — P. 521–543.

118. Prisner E. Bicliques in graphs I: bounds on their number / E. Prisner // Combinatorica. — 2000. — Vol. 20. — P. 109–117.

119. Qian T. Decomposition methods of formal contexts to construct concept lattices / T. Qian, L. Wei, J. Qi // International Journal of Machine Learning and Cybernetics. — 2017. — Vol. 8. — № 1. — P. 95–108.

120. Salton G. Automatic Information Organization and Retrieval / G. Salton — New York: McGraw-Hill, 1968. — 514 p.

121. Simon A. A. Best-of-Breed approach for designing a fast algorithm for computing fixpoints of Galois Connections / A. A. Simon // Information Sciences. — 2015. — Vol. 295. — № 2. — P. 633–649.

122. Turney P. D. From Frequency to Meaning: Vector Space Models of Semantics / P. D. Turney, P. Pantel // Journal of Artificial Intelligence Research. — 2010. — Vol. 30. — № 1. — P. 141–188.

123. Valtchev P. Galicia: an open platform for lattices, in using conceptual structures / P. Valtchev, D. Grosser, C. Roume, M. R Hacene // 11th International Conference on Conceptual structures : proceedings. — Aachen, Germany: Shaker Verlag, 2003. — P. 241–254.

124. Vania M. F. Generating bicliques of a graph in lexicographic order / M. F. Dias Vania, M. H. de Figueiredo Celina, L.S. Jayme // Theoretical Computer Science. — 2005. — № 337. — P. 240–248.

125. Vlasov D. V. The methods of forming the theoretical concepts / D. V. Vlasov // Journal of the Buryat State University. — 2009. — № 6. — P. 37–41.

126. Wille R. Restructuring lattice theory: an approach based on hierarchies of concepts / R. Wille // 7th International Conference on Formal Concept Analysis : proceedings. — Darmstadt, Germany: Springer-Verlag, 2009. — P. 314–339.

127. Wood D. R. On the Maximum Number of Cliques in a Graph / D. R. Wood // Graphs and Combinatorics. — 2007. — № 23. — P. 1–16.

УТВЕРЖДАЮ
Ректор ТувГУ
Хомушку Ольга Матпаевна
_____
«17» 09 2019 г.

АКТ
использования научных результатов диссертационной работы Монгуш Ч.М. «Разработка метода и средств фрагментации и дефрагментации формальных контекстов»

Комиссия ФГБОУ ВО «Тувинский государственный университет» в составе Бавуу-Сюрюн Миры Викторовны (канд. филол. наук, директора научно-образовательного центра «Тюркология»), Далаа Сергея Монгушевича (канд. физ.-мат. наук, доцента), Салчак Аэлиты Яковлевны (канд. филол. наук, доцента) рассмотрела вопрос об использовании результатов диссертационной работы Монгуш Чодураа Михайловны «Разработка метода и средств фрагментации и дефрагментации формальных контекстов» на соискание ученой степени кандидата физико-математических наук по специальности 05.13.17 – Теоретические основы информатики, выполненной в Сибирском федеральном университете.

Результаты диссертационной работы Монгуш Ч.М. обладают актуальностью и представляют интерес для электронного корпуса текстов тувинского языка. Программы для ЭВМ «Программа FCACorpus концептуального моделирования тувинских текстов методами анализа формальных понятий», «Программа формирования контекстов в корпусе тувинского языка», разработанные Монгуш Ч.М. в рамках диссертации, переданы в научно-образовательный центр «Тюркология» Тувинского государственного университета для встраивания в корпус тувинского языка.

Переданные программы рекомендованы к использованию в научно-образовательном центре «Тюркология» для проведения научных исследований в рамках мероприятий по выполнению проекта Госзадания №34.3876.2017/ПЧ Министерства науки и высшего образования РФ.

Директор НОЦ «Тюркология» ТувГУ, канд. филол. наук _____ М.В. Бавуу-Сюрюн

Старший научный сотрудник НОЦ «Тюркология» ТувГУ, канд. физ.-мат. наук _____ С.М. Далаа

Старший научный сотрудник НОЦ «Тюркология» ТувГУ, канд. филол. наук _____ А.Я. Салчак

УТВЕРЖДАЮ

Ректор ТувГУ

Хомушку Ольга Матпаевна

_____

«_01_» _____08_____ 2019 г.

АКТ

о внедрении в учебный процесс Федерального государственного бюджетного образовательного учреждения высшего образования «Тувинского государственного университет» научных результатов диссертационной работы Монгуш Ч.М. «Разработка метода и средств фрагментации и дефрагментации формальных контекстов»

Программы для ЭВМ «Программа FCACorpus концептуального моделирования тувинских текстов методами анализа формальных понятий», «Программа формирования контекстов в корпусе тувинского языка», разработанные Монгуш Чодураа Михайловной, и теоретические результаты кандидатской диссертации «Разработка метода и средств фрагментации и дефрагментации формальных контекстов», выполненной в Сибирском федеральном университете, внедрены в учебный процесс на кафедре информатики и ИКТ. Эти материалы используются при подготовке бакалавров по специальности 02.03.02 – «Фундаментальная информатика и информационные технологии», при изучении дисциплины «Введение в анализ данных» и выполнении выпускных квалификационных работ.

Заведующий кафедрой информатики
и ИКТ ТувГУ, канд. пед. наук, доцент _____Д.О. Куулар