Федеральное государственное бюджетное учреждение науки
Институт динамики систем и теории управления имени В. М. Матросова
Сибирского отделения Российской академии наук (ИДСТУ СО РАН)

На правах рукописи

Шигаров Алексей Олегович

Методы и инструментальные средства автоматизации процессов извлечения данных из таблиц электронных документов неструктурированного формата

2.3.5 – Математическое и программное обеспечение вычислительных систем, комплексов и компьютерных сетей

ДИССЕРТАЦИЯ
на соискание ученой степени
доктора технических наук

Научный консультант
академик, д-р техн. наук
Бычков Игорь Вячеславович

Иркутск – 2025

Оглавление

Введение

Актуальность. Таблицы являются способом представления реляционных данных. Когда они заключены в электронных документах *неструктурированного формата* (т. е. без схемы данных), например в растровых изображениях, печатно-ориентированных описаниях страниц (PDF[1], PostScript[2] и др.), веб-страницах (HTML[3]) или рабочих книгах (Excel[4], Sheets[5] и др.), в научной литературе их часто называют *документными таблицами*[6]. К настоящему времени в мире накоплен большой массив такой информации. Например, по некоторым оценкам, опубликованным в научной литературе, количество только HTML-таблиц с реляционными данными в открытой части «Всемирной паутины» исчисляется по крайней мере сотнями миллионов. Предполагается, что из них можно извлечь сотни миллиардов фактов. Объем всех документных таблиц в мире неизвестен, но, вероятно, значительно превосходит указанные оценки.

Документные таблицы являются ценным источником информации в различных приложениях, в том числе системах поддержки принятия решений, интеллектуальном анализе данных, конструировании баз знаний и информационном поиске. Для того чтобы табличная информация могла быть проиндексирована, запрошена и применена в перечисленных приложениях, прежде всего она должна быть приведена к структурированному представлению (базам данных или графам знаний). Последнее согласуется с целью *извлечения информации*, которая состоит в получении структурированных данных, а именно сущностей и связей между ними, из неструктурированных источ-

[1] `https://pdfa.org/resource/iso-32000-pdf`

[2] `https://www.adobe.com/jp/print/postscript/pdfs/PLRM.pdf`

[3] `https://www.w3.org/html`

[4] `https://www.microsoft.com/en-us/microsoft-365/excel`

[5] `https://www.google.ru/intl/ru/sheets/about`

[6] Перевод с англ. «*Document table*».

ников. Однако применить существующий инструментарий *извлечения информации*, ориентированный главным образом на текст, напрямую к таблицам, как правило, невозможно. Поскольку, в отличие от текста, рассматриваемая информация выражена не только с помощью естественного языка, но также и посредством размещения ее частей в структуре ячеек.

В литературе сложившуюся проблематику извлечения структурированных данных из документных таблиц принято называть *автоматизированным пониманием таблиц*[7] (далее АПТ). Базовая задача АПТ может быть сформулирована следующим образом. Имеется документная таблица t, доступная как часть неструктурированного источника d, такая что представленные в ней данные составляют один или несколько *наборов записей* $R_1, \ldots, R_n$ со схемами соответственно $S_1, \ldots, S_n$, описывающими их атрибуты. Будем называть *канонической формой* набора записей R_i такую таблицу, в которой заголовок с именами атрибутов схемы S_i занимает первую строку, каждая запись из набора R_i полностью размещается в одной из последующих строк (в одной строке — одна запись), каждому атрибуту схемы S_i соответствует отдельный столбец (в нем ячейка заголовка содержит его имя, а остальные ячейки — его значения, характеризующие соответствующие записи). Требуется извлечь наборы записей $R_1, \ldots, R_n$ в канонической форме, доступной в машиночитаемом формате, из неструктурированного источника d. Расширенная формулировка может также предусматривать дальнейшие действия, направленные на нормализацию извлеченных данных и сопоставление их с внешними словарями и схемами.

Сложность АПТ обусловлена двумя основными факторами: во-первых, наличием большого разнообразия способов компоновки, форматирования и наполнения таблиц; во-вторых, ограниченностью форматов их представления. Являясь частью неструктурированного источника, документная табли-

[7]Перевод с англ. «*Automated table understanding*».

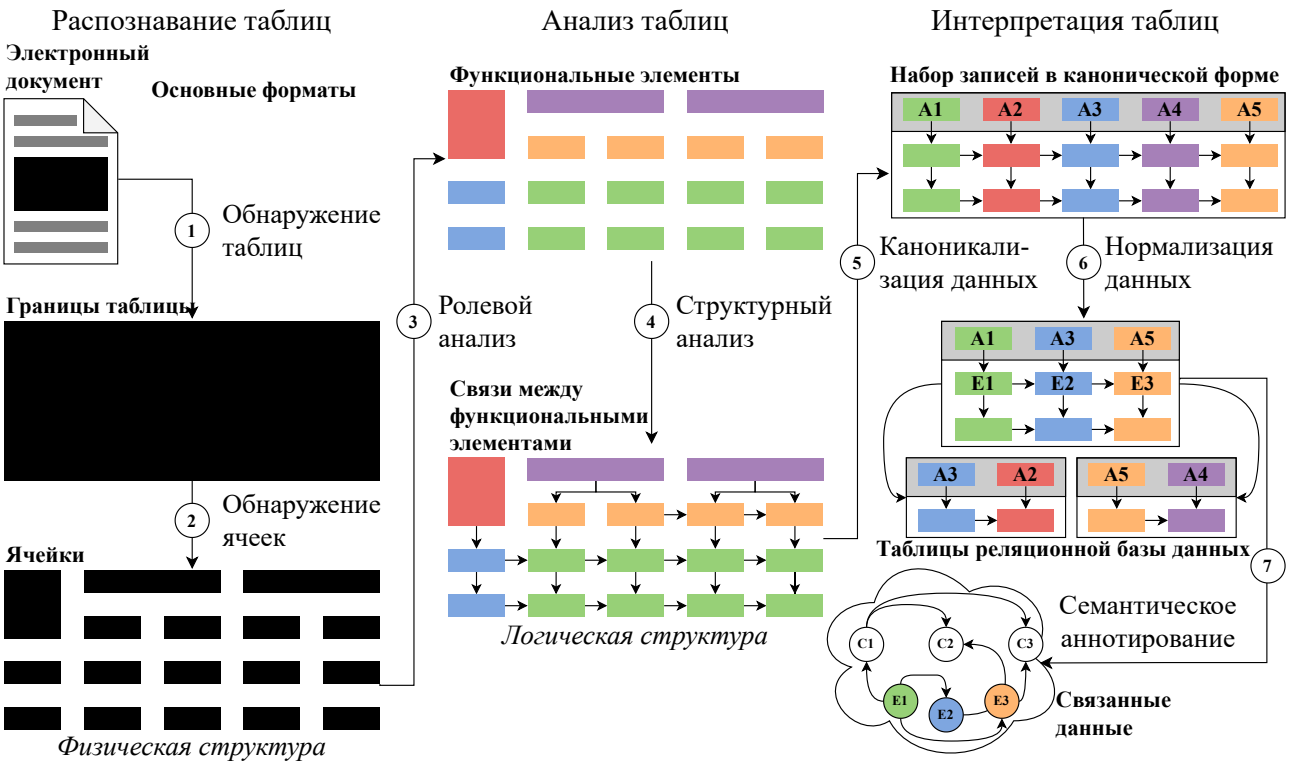


Рисунок 1 — Основные этапы АПТ

ца обычно не сопровождается формальной моделью, позволяющей интерпретировать представленную там информацию в соответствии со смыслом, заложенным в нее автором. Обычно неизвестно местоположение документной таблицы внутри источника, структура ячеек, логический порядок чтения данных и пр. В общем случае процесс АПТ включает следующие этапы: *распознавание* — обнаружение местоположения таблицы и ее ячеек в источнике; *анализ* — восстановление логического порядка чтения данных; *интерпретация* — восстановление соответствующего ей набора записей в канонической форме, приведение его к некоторой совокупности отношений в терминах реляционной модели данных и сопоставление их с внешними словарями и схемами (рис. 1). При текущем уровне развития информационных технологий данные процессы в общем случае не могут выполняться без участия человека. Поэтому автоматизация этих процессов нацелена на сокращение операций, производимых человеком.

Исследование вопросов данной проблематики началось еще в середине 1990-х. За три последних десятилетия были защищены десятки диссертаций, опубликованы тысячи научных статей и зарегистрированы по крайней мере сотни патентов. Значительный вклад в исследование вопросов, связанных с ней, внесли отечественные (В. Л. Арлазаров, М. Ю. Богатырев, И. В. Бычков, В. Э. Вольфенгаген, К. В. Воронцов, О. Ю. Гавенко, В. И. Городецкий, В. В. Грибова, К. А. Зуев, С. В. Зыкин, С. В. Зыков, Л. А. Калиниченко, М. Р. Когаловский, С. Д. Кузнецов, С. В. Кулешов, Н. В. Лукашевич, А. Г. Марчук, В. В. Миронов, Д. И. Муромцев, Б. А. Новиков, Г. М. Ружников, А. М. Федотов, А. Е. Хмельнов, А. А. Хорошилов и др.) и зарубежные (C. Bhagavatula, K. Braunschweig, T. M. Breuel, D. Burdick, M. Cafarella, Z. Chen, E. Crestan, J. Cunha, A. Dengel, A. C. e Silva, J. Eberius, V. Efthymiou, D. W. Embley, M. Erwig, J. Fang, W. Gatterbauer, V. Govindaraju, S. Gulwani, A. Halevy, T. Hassan, M. Hurst, T. Kieninger, E. Koci, M. S. Krishnamoorthy, O. Lehmberg, W. Lehner, G. Limaye, Y. Liu, D. Lopresti, N. Milosevic, V. Mulwad, G. Nagy, R. Rastan, S. Roldan, S. Schreiber, S. Seth, F. Shafait, P. Szekely, M. Thiele, X. Wang, Y. Wang, S. Yang, R. Zanibbi, Z. Zhang и др.) ученые.

Несмотря на продолжительную историю, она до сих пор остается открытой, нуждаясь в выработке общих теоретических основ и создании технологических решений, применимых для различных сред и форматов представления табличной информации. Наблюдаемый рост количества публикаций по данной проблематике показывает, что интерес к ней со стороны научного сообщества продолжает усиливаться (рис. 2). Последнее десятилетие ознаменовалось всплеском публикаций, предлагающих решения задач АПТ на основе новых техник глубокого обучения, векторного представления слов и сущностей, а также связанных открытых данных. В рамках ведущих кон-

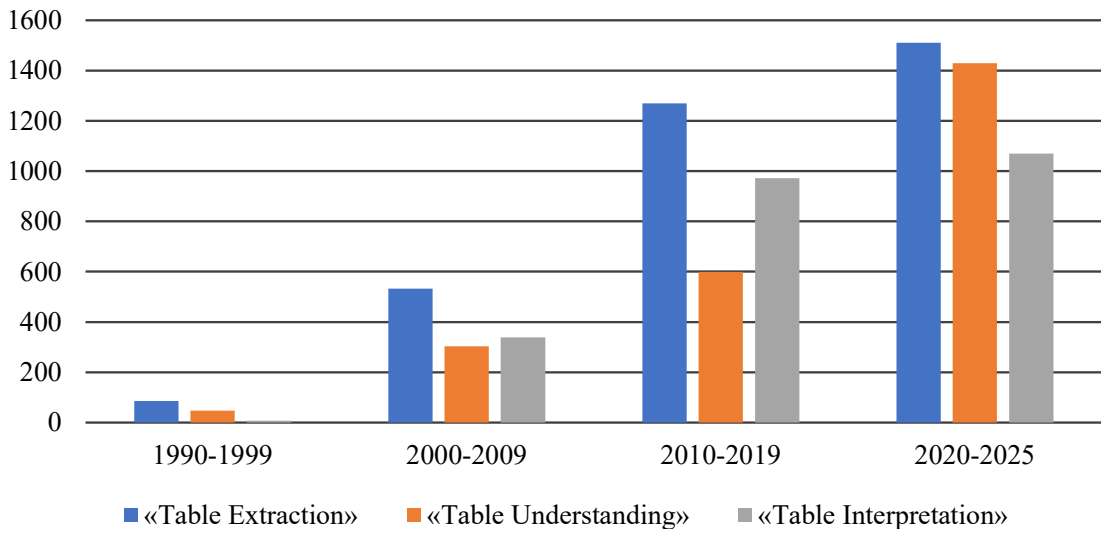


Рисунок 2 — Количество публикаций по ключевым словам АПТ (по данным сервиса поиска научной литературы «Google Scholar» на апрель 2025 года)

ференций (ICDAR[8], ISWC[9], NeurIPS[10], VLDB[11] и др.) стали проводиться тематические секции, семинары и соревнования по отдельным задачам АПТ с участниками со всего мира. Сегодня некоторые элементы АПТ уже представлены в популярных сервисах извлечения данных из документов от крупнейших технологических компаний: «Amazon Textract»[12], «IBM Smart Document Understanding»[13], «Google Document AI»[14], «Microsoft Azure AI Document Intelligence»[15] и др.

Современные обзоры тематической литературы показывают, что все известные решения автоматизации извлечения данных из документных таблиц являются частными. Их общим ограничением является отсутствие поддержки произвольности структуры документных таблиц. Известные методы полагаются на *типичную компоновку* (т. е. небольшое количество — от одного

[8] https://icdar.org

[9] https://semanticweb.org

[10] https://nips.cc

[11] https://vldb.org

[12] https://aws.amazon.com/ru/textract

[13] https://cloud.ibm.com/docs/discovery?topic=discovery-sdu

[14] https://cloud.google.com/document-ai

[15] https://azure.microsoft.com/en-us/products/ai-services/ai-document-intelligence

до пяти — компоновочных типов, наиболее распространенных в открытой части «Всемирной паутины»), атомарность ячеек и плоскую структуру заголовков, игнорируя большое количество случаев, когда эти предположения не выполняются. Таким образом, автоматизация процессов извлечения данных из документных таблиц произвольной структуры является актуальной научной проблемой. В частности, ее решение имеет важное хозяйственное значение для массовой обработки таких источников табличной информации, как *нередактируемые печатно-ориентированные документы* (далее НПОД), представленные в форматах языков описания страниц (далее PDL): PDF, PostScript и др., а также *рабочие книги* (далее РК), представленные в форматах табличных процессоров: Excel, Sheets и др.

Цель исследования состоит в создании методов АПТ и комплекса инструментальных средств на их основе для упрощения разработки прикладного программного обеспечения извлечения данных из документных таблиц с машиночитаемым текстовым содержимым, представленных в неструктурированном виде, за счет поддержки произвольности табличной структуры. Для достижения поставленной цели были решены следующие **задачи**: проанализирована совокупность задач АПТ; выявлены ограничения текущего исследования вопросов АПТ и сформулированы актуальные направления его развития; предложен метод автоматизации распознавания таблиц в НПОД, т. е. конвертирования их в редактируемый формат РК; предложен метод автоматизации анализа и интерпретации таблиц РК, т. е. извлечения из них наборов записей в канонической форме; разработаны инструментальные средства на основе предлагаемых методов; выполнена оценка производительности реализованных решений и сравнение их с аналогами; изучены возможности практической применимости предлагаемых методов и инструментальных средств.

Объектом исследования является совокупность процессов АПТ, а его **предметом** — методы и программные средства автоматизации процессов из-

влечения данных из произвольных документных таблиц с машиночитаемым текстовым содержимым, представленных в НПОД/РК.

Методы исследования основаны на применении, в том числе эвристик, машинного обучения, исполнения правил, моделей данных, формальных грамматик, трансляции программ и тестах производительности.

Положения, выносимые на защиту:

1. Усовершенствованная структура совокупности задач АПТ, в рамках которой была согласована терминология АПТ, сложившаяся в родственных исследовательских направлениях: компьютерном зрении, управлении данными, информационном поиске и «Семантической паутине».
2. Метод автоматизации распознавания таблиц НПОД на основе правил анализа компоновки страниц, использующих свойства PDL-представления. Разработаны соответствующие методики сегментации страниц, обнаружения и сегментации таблиц НПОД.
3. Модель представления табличной структуры в процессах АПТ, не ограниченная предопределенной компоновкой, атомарностью ячеек и плоскими заголовками.
4. Метод автоматизации анализа и интерпретации таблиц РК на основе исполнения правил. Обеспечена поддержка произвольности структуры таблицы, а именно свободной компоновки, структурированности ячеек и иерархичности заголовков.
5. Проблемно-ориентированный язык правил анализа и интерпретации таблиц, обеспечивающий разработку пользовательских программ извлечения данных из документных таблиц.
6. Комплекс инструментальных средств (далее КИС), разработанный на основе предложенных теоретических положений для автоматизации основных процессов извлечения данных из произвольных таблиц с ма-

шиночитаемым текстовым содержимым, представленных в форматах НПОД/РК.

Научная новизна:

1. По сравнению с имеющимися формулировками АПТ, актуализированная структура совокупности задач АПТ является более релевантной за счет согласованной терминологии и многоуровневой декомпозиции.
2. В отличие от аналогов предлагаемый метод автоматизации распознавания таблиц базируется на использовании особенностей представления таблиц в НПОД. Впервые показано, как можно задействовать PDL-специфичную информацию для сегментации страниц и распознавания таблиц НПОД с целью улучшения качества их результатов.
3. В отличие от моделей документных таблиц, применяемых конкурентными решениями, созданная модель ориентирована на представление произвольной табличной структуры, что позволяет ей быть применимой к более широкому кругу сценариев АПТ.
4. В отличие от аналогов предлагаемый метод автоматизации анализа и интерпретации таблиц РК основан на пользовательском программировании правил. Впервые обеспечена поддержка произвольности структуры таблицы, а именно свободной компоновки, структурированности ячеек и иерархичности заголовков.
5. Впервые создан проблемно-ориентированный язык правил анализа и интерпретации документных таблиц. В отличие от языков общего назначения, он позволяет сфокусироваться исключительно на реализации логики соответствующих этапов АПТ, упрощая тем самым разработку целевого программного обеспечения (далее ПО).
6. По сравнению с другими доступными инструментами АПТ разработанный КИС в части распознавания таблиц НПОД дополнен анализом

компоновки страниц и фильтрацией кандидатных случаев, а в части извлечения данных из таблиц РК отделяет правила их анализа и интерпретации от моделей представления и алгоритмов обработки.

Теоретическая значимость основных результатов состоит в том, что в совокупности они составляют теоретические основы решения проблемы упрощения разработки прикладного программного обеспечения извлечения данных из документных таблиц неструктурированного формата (НПОД/РК) за счет поддержки произвольности табличной структуры. Их **практическая значимость** обосновывается созданием технологии автоматизации процессов распознавания, анализа и интерпретации документных таблиц НПОД/РК. Разработанные инструментальные средства нашли применение в пяти научных и трех индустриальных проектах при решении прикладных задач анализа документов (технических, финансовых и научных), интеграции данных (государственной статистики и медиапланирования), конструирования онтологии предметной области и кросс-контекстного обмена бизнес-документами. (Из них пять проектов выполнено сторонними коллективами.)

Достоверность полученных результатов подтверждается представленными в диссертационной работе экспериментальными данными, качественным и количественным сравнением с имеющимися аналогами, а также программной реализацией. Разработанный КИС и материалы для воспроизведения проведенных экспериментов опубликованы в открытом доступе под свободными лицензиями[16].

Апробация. Основные результаты работы представлялись на международных и всероссийских научных мероприятиях: «Информационные и математические технологии в науке и управлении» (ИМТ 2007–2009, 2013); «Pattern recognition and image analysis» (PRIA 2008, 2010); «Matematičke i

[16] `https://tabbydoc.github.io`

informacione tehnologije» (MIT 2010); «Data analytics and management in data intensive domains» (DAMDID/RCDL 2014–2016); «Information and software technologies» (ICIST 2015, 2016, 2018–2021); «ACM Document engineering» (DocEng 2016, 2018); «Information systems architecture and technology» (ISAT, 2019); «Information and communication technology, electronics and microelectronics» (MIPRO 2019, 2020, 2022); «Information, computation, and control systems for distributed environments» (ICCS-DE 2023), «Марчуковские научные чтения» (МНЧ 2024) и др. Они также обсуждались на ряде совещаний и семинаров ИДСТУ СО РАН (Иркутск), ОНИТ СО РАН (Новосибирск), Института национального развития Монголии (Улан-Батор, Монголия), Харбинского политехнического университета (Харбин, Китай) и др.

Публикации. Основные результаты опубликованы в 70 работах [1–70] включая 12 статей [1–12] в журналах, рекомендованных Высшей аттестационной комиссией (ВАК) для опубликования основных научных результатов, из которых все 12 изданий относятся к категории К1[17], а четыре из них [3, 5, 7, 8] — к первому уровню «Белого списка»[18]. Получено шесть свидетельств о государственной регистрации программ для ЭВМ [13–18].

Соответствие паспорту специальности. Полученные результаты согласуются со следующими направлениями исследований: пункт паспорта 1 — разработанные модели, методы и методики могут использоваться при проектировании программных систем извлечения данных из документных таблиц (см. защищаемые положения 2–5); пункт паспорта 2 — созданные инструментальные средства обеспечивают разработку прикладного ПО извлечения данных из таблиц РК на основе проблемно-ориентированного языка правил (см. защищаемые положения 5 и 6); пункт паспорта 3 — представленные модели, методы и методики позволяют организовать взаимодействие программ

[17] См. «Итоговое распределение журналов Перечня ВАК по категориям К1, К2, К3 в 2023 году».

[18] `https://journalrank.rcsi.science/ru`

распознавания, анализа и интерпретации документных таблиц (см. защищаемые положения 1–6).

Структура и объем. Диссертация состоит из введения, шести глав, заключения, списка литературы и пяти приложений. Объем диссертации составляет 291 страницу, с приложениями — 312; представлено 116 рисунков и 17 таблиц; процитировано 436 источников. Основное содержание изложено в следующем порядке. Первая глава дает обзор современного состояния исследований вопросов АПТ. Вторая глава представляет метод автоматизации распознавания таблиц, а третья — метод автоматизации анализа и интерпретации таблиц. Четвертая глава посвящена описанию инструментальных средств АПТ. В пятой главе приводятся результаты оценки производительности реализованных решений и сравнение их с имеющимися аналогами. В шестой главе рассматриваются случаи практического применения предложенных методов и разработанных на их основе инструментальных средств АПТ.

Личный вклад автора. Все выносимые на защиту положения получены соискателем лично. Из совместных работ, в том числе [4–7, 9, 11, 12], в диссертацию включены только те результаты, которые принадлежат непосредственно автору, включая постановку задач, разработку предлагаемых методов, моделей, языковых и инструментальных средств для АПТ, а также планирование и проведение экспериментов. В неделимом соавторстве с А. А. Алтаевым, А. И. Бондаревым, А. А. Михайловым, В. В. Парамоновым, Е. В. Рожковым, В. В. Христюком и И. А. Черепановым выполнена программная реализация и оценка производительности предлагаемых методов. Совместно с А. А. Ветровым, Н. О. Дородных, В. В. Парамоновым и А. Ю. Юриным реализованы примеры их практического применения, с И. В. Бычковым, Г. М. Ружниковым и А. Е. Хмельновым определены направления и методы исследования.

Благодарности. Автор выражает глубокую благодарность И. В. Бычкову, Г. М. Ружникову и А. Е. Хмельнову за научное консультирование и обмен идеями. Настоящая работа выполнена при поддержки грантов научных фондов, в том числе: «Методология и инструментальная платформа разработки систем извлечения данных из произвольных электронных таблиц» (РНФ № 18-71-10001, 2018–2023 гг.), «Методы интеграции неструктурированных табличных данных» (РФФИ № 15-37-20042, 2015–2016 гг.), «Методы извлечения табличной информации из неструктурированных текстовых источников» (Совет по грантам Президента РФ № СП-3387.2013.5, 2013–2015 гг.), «Интеллектуальная система извлечения информации из слабоструктурированных таблиц со сложной компоновкой» (РФФИ № 12-07-31051, 2012–2013 гг.), а также гранта Министерства науки и высшего образования РФ на выполнение крупного научного проекта по приоритетным направлениям научно-технологического развития (проект «Фундаментальные исследования Байкальской природной территории на основе системы взаимосвязанных базовых методов, моделей, нейронных сетей и цифровой платформы экологического мониторинга окружающей среды», № 075-15-2024-533, 2024–2025 гг.).

Глава 1

Современное состояние исследований

В данной главе представлен обзор проблематики АПТ и современного состояния исследований, посвященных ее вопросам. Вводятся основные понятия документных таблиц (раздел 1.1). Дается характеристика проблематики (раздел 1.2. Рассматриваются постановки задач и существующие методы решения (раздел 1.3), а также доступные тесты производительности и коллекции данных (раздел 1.4). Очерчивается область применения существующих решений (раздел 1.5). Обсуждаются ограничения текущих исследований и предлагаются актуальные направления развития (раздел 1.6). В конце главы делаются выводы (раздел 1.7).

1.1. Основные понятия документных таблиц

H. Hinterberger [210] дает энциклопедическое определение *таблицы*, с одной стороны, как средства визуальной коммуникации, размещающего информацию в ячейках двумерной решетки, а с другой — как структуры данных для организации кортежей некоторого отношения. Таблицу принято называть *подлинной*, если она содержит реляционные данные, и *неподлинной* в противном случае [392]. Когда подлинная таблица является частью некоторого электронного документа, будем называть ее *документной*. В зависимости от функции представленных в ней данных, она может быть *сущностной* или *многомерной*.

Сущностная таблица представляет либо некоторый набор однотипных сущностей, либо соединение нескольких наборов. Рассмотрим первый случай (рис. 1.1). Обозначим через *e сущность*, описывающую некоторую вещь ре-

Оборудование и его элементы	Справочные данные		Паспортные данные	
Трансформатор Т-1	Тип	DEP-1600/5	Дата изготовления	1940
	Заводской номер	406034	Ввод в эксплуатацию	1953
	Завод-изготовитель	GARBE LAHMEYER Германия	Вес полный	6680
	Мощность	1600 кВА	Дата последнего к.р.	2004
	Система охлаждения	естественное маслянное	Дата последнего т.р.	2006
	Данные о ремонтах	в соответствии с графиком	Периодичность к.р.	по испытан.
	Данные испытаний	соответствует требованиям РД34.45-51.300-97	Периодичность т.р.	1 раз в год
	Данные о дефектах	отсутсвуют	**ПЕРИОДИЧНОСТЬ**	
			ВВИ	1 раз в 3 года
			ХАРГ	2 раза в год
			Хим. анализ масла	1 раз в год

— группы атрибутов — атрибуты — значения данных

Рисунок 1.1 — Сущностная таблица, представляющая один набор однотипных сущностей

ального мира. Пусть D — множество допустимых значений некоторого атрибута a, называемое *доменом*, тогда будем говорить, что сущность e описывается данным атрибутом a, если ей сопоставлено некоторое значение домена D. Будем говорить, что сущности $e_1, \ldots, e_n$, обладающие общими атрибутами $a_1, \ldots, a_m$, составляют *набор однотипных сущностей* E. Если $D_1, \ldots, D_m$ — домены атрибутов $a_1, \ldots, a_m$ соответственно, тогда каждая сущность $e : e \in E$ описывается одним значением в каждом из ее атрибутов:

$$e = (v_1, \ldots, v_m \mid v_1 \in D_1, \ldots, v_m \in D_m).$$

Набор однотипных сущностей E соответствует отношению в терминах реляционной модели данных [134], в котором каждой сущности $e : \ e \in E$ сопоставлен ровно один кортеж $\langle v_1, \ldots, v_m \mid v_1 \in D_1, \ldots, v_m \in D_m \rangle$, а каждому атрибуту $a : a \in A$ — единственное поле.

Рассмотрим случай *соединения нескольких наборов однотипных сущностей* (рис. 1.2). Пусть наборы однотипных сущностей E_1 и E_2 обладают наборами атрибутов соответственно A_1 и A_2, такими что найдется по крайней

Отдел	Сотрудник	Журнал (Индексация)		
Динамики систем	Иванов И.И.	ЖВТ	РИНЦ	Scopus
		Differential Equations	WOS	Scopus
		SowfwareX	Scopus	
	Петров П.П.	System Dynamics Review	WOS	Scopus
	Сидоров С.С.	ESWA	WOS	
		Известия ИГУ	РИНЦ	
Теории управления	…	…	…	

— атрибуты — значения данных

Рисунок 1.2 — Сущностная таблица, представляющая соединение нескольких наборов однотипных сущностей

мере один общий атрибут a для пары сущностей $(e_1, e_2 \mid e_1 \in E_1, e_2 \in E_2)$, тогда любое подмножество декартова произведения этих наборов $J \subseteq E_1 \times E_2$ будем называть их *соединением*. Аналогично и само соединение J может быть объединено с третьим набором однотипных сущностей или другим соединением в том случае, когда они также имеют по крайне мере один общий атрибут. Таким образом, соединение в общем случае может являться подмножеством декартова произведения нескольких наборов однотипных сущностей $J \subseteq E_1 \times \cdots \times E_n$ с общим множеством атрибутов $A = A_1 \cup \cdots \cup A_n$. Пусть общее множество A состоит из атрибутов $a_1, \ldots, a_m$, которым соответствуют домены $D_1, \ldots, D_m$, тогда соединение J может быть представлено в виде отношения, в котором каждому кортежу связанных сущностей $\langle e_1, \ldots, e_n \mid e_1 \in E_1, \ldots, e_n \in E_n \rangle$ сопоставлен ровно один кортеж значений их атрибутов $\langle v_1, \ldots, v_m \mid v_1 \in D_1, \ldots, v_m \in D_m \rangle$, а каждому атрибуту $a : a \in A$ — единственное поле.

Многомерная таблица представляет данные, характеризуемые совместно значениями двух или более *измерений*. Под измерением понимается множество однотипных значений: литеральных или категориальных. *Литеральные* — числа, символы и пр. *Категориальные* — упоминания сущностей, они

DATA	SALES CHANNEL	FISCAL YEAR	CURRENCY	PRODUCT (GROUP)	PRODUCT (DETAIL)
11200	retail	2016	u.s. dollars	electronics	phones
23700	retail	2017	u.s. dollars	electronics	phones
12600	catalog	2016	u.s. dollars	electronics	phones
32200	catalog	2017	u.s. dollars	electronics	phones
89900	retail	2016	u.s. dollars	electronics	computers
...	...	...	...	...	...

—значения измерений — атрибуты

—значения набора многомерных данных

Рисунок 1.3 — Многомерная таблица с односторонней компоновкой

могут быть атомарными или составными. Под *атомарным* понимается значение единственного атрибута, а *составным* — кортеж значений нескольких атрибутов. Пусть $D_1, \ldots, D_n$ — измерения, характеризующие совместно набор данных Q, тогда можно говорить, что многомерная таблица представляет функцию t, которая отображает некоторое подмножество декартова произведения значений этих измерений $V \mid V \subseteq D_1 \times \cdots \times D_n$ на набор данных Q:

$$t : V \to Q.$$

Данная функция соответствует отношению следующим образом: каждый кортеж имеет вид $\langle v_1, \ldots, v_n, q \rangle$, где $v_1, \ldots, v_n \mid v_1 \in D_1, \ldots, v_n \in D_n$ — совместная характеристика значения $q \mid q \in Q$; каждому измерению соответствует единственное поле, в совокупности эти поля составляют ключ, и еще одно поле, функционально зависимое от данного ключа, соответствует набору данных Q. В общем случае, данное отношение является ненормализованным, поскольку оно допускает присутствие составных значений в любом из его полей, а также функциональные зависимости между полями измерений.

Многомерная таблица может иметь одностороннюю (рис. 1.3) или многостороннюю (рис. 1.4) компоновку. При *односторонней компоновке* значения данных и всех соответствующих измерений размещаются в виде вертикального или горизонтального списка. При *многосторонней компоновке* строки по-

	Retail Sales		Catalog Sales	
	FY2016	FY2017	FY2016	FY2017
	(thousands of dollars)			
Electronics	114,5	259,5	106,2	326,4
– Phones	11,2	23,7	12,6	32,2
– Computers	89,9	203,1	81,9	204,1
– TV	13,4	32,7	11,7	90,1
Books	12,3	21,6	11,8	24,5
…	...	...	...	...

—значения измерений —значения набора многомерных данных

Рисунок 1.4 — Многомерная таблица с многосторонней компоновкой

мечаются значениями одной части измерений, а столбцы — другой; ячейка в пересечении каждой строки и столбца содержит некоторое значение данных, характеризуемое этими измерениями.

Принято различать физическую и логическую части структуры документной таблицы [221, 397]. *Физическая структура* состоит из ячеек, характеризуемых своими координатами в пространстве строк и столбцов, форматированием и содержимым. *Логическая структура* представлена функциональными элементами таблицы и связями между ними. *Функциональный элемент* может определяться либо на уровне ячеек, либо на уровне данных: в первом случае он состоит из одной или нескольких ячеек; во втором — представлен атомарным значением внутри содержимого одной ячейки. Функциональный элемент выполняет некоторую функцию (роль) внутри таблицы, например представляет атрибут или значение. *Связь* — упорядоченная пара двух функциональных элементов, которая выражает некоторое логическое отношение, например атрибут–значение или родитель–потомок. Следует отметить, что некоторые исследователи [247, 248, 327] рассматривают функциональные части таблицы (например, шапка, боковик и тело) и выводят связи между ними. Будем называть такие части *функциональными регионами*.

1.2. Характеристика научной проблематики

Приводятся некоторые оценки объема информации, представленной в виде документных таблиц (раздел 1.2.1). Обсуждается сложность ее обработки (раздел 1.2.2). Дается историческая ретроспектива тематических исследований (раздел 1.2.3). Предлагается иерархическая структура совокупности задач АПТ (раздел 1.2.4).

1.2.1. Объем табличной информации

Таблицы использовались для хранения и обмена информацией с древних времен. В качестве примера можно привести некоторые находки археологов. C. Proust [326] описывает таблицы из южной Месопотамии, датированные ок. 2600–2500 гг. до н. э. (рис. 1.5, *а*). P. Tallet и G. Marouard [379] сообщают о находке производственного отчета в виде таблиц, записанных на папирусе, датированном ок. 2500 г. до н. э. (древнейшем папирусе из найденных к настоящему моменту) (рис. 1.5, *б*). M. Campbell-Kelly и др. [110] замечают, что "*таблицы являются почти исторической необходимостью, вышедшей из двухмерности письменной поверхности, общей для всех цивилизаций*".

В настоящее время таблицы являются общепринятым способом представления реляционных данных в различных источниках, как неструктуриро-

а *б*

Рисунок 1.5 — Две древние таблицы: шумерская глиняная табличка с таблицей площадей, известная как VAT 12593 (*а*); отчет на папирусе, связанный со строительством пирамиды Хеопса (*б*)

ванных, так и структурированных. Очевидно, что в мире существует большой объем такой информации. Приведем некоторые оценки, подтверждающие это утверждение. Современные исследования открытой части «Всемирной паутины» [105, 140, 160, 264] собрали сотни миллионов веб-таблиц. L. Lautert и др. [263] указывают, что «*Всемирная паутина является крупнейшим хранилищем данных, содержащим более 150 миллионов высококачественных таблиц, содержащих реляционные данные*» (на 2013 год). C. S. Bhagavatula и др. [95] отмечают, что одна только Википедия содержит 1,6 миллиона реляционных таблиц (на 2015 год). Недавно сотни тысяч документных таблиц были извлечены из научных статей, опубликованных в открытом доступе в репозиториях arXiv[1] [270] и PubMed[2] [432].

Очевидно, что значительное количество табличной информации накапливается в РК (Excel, Sheets и др.) и плоских файловых базах данных (CSV, DBF и др.). В исследованиях [90, 117] собраны сотни тысяч таких примеров. J. Mitlöhner и др. [296] установили, что 10 % всех ресурсов порталов открытых данных помечены именно как табличные. Проект GitTables[3] недавно собрал 1,7 миллиона реляционных таблиц с ресурсов GitHub[4]; предполагается, что этот набор данных может быть увеличен как минимум до 20 миллионов таблиц. Косвенно об объеме табличной информации можно судить по исследованию C. Scaffidi и др. [353], которое оценивает, что только в США в 2012 году должно было быть около 55 миллионов конечных пользователей, использующих табличные процессоры и системы управления базами данных.

Объем, доступность и реляционная природа делают их ценным источником данных для создания вопросно-ответных систем [226, 421], генерации текста [124,315], автозаполнения форм [108,255,420], наполнения баз знаний [212,

[1] `https://arxiv.org`

[2] `https://pubmed.ncbi.nlm.nih.gov`

[3] `https://gittables.github.io`

[4] `https://github.com`

291, 340, 421], интеллектуального анализа научной литературы [193, 256, 407] и автоматического реферирования [309, 422]. Однако трудности, неизбежно возникающие при извлечении и интеграции такого рода информации, часто препятствуют ее интенсивному использованию в реальных приложениях.

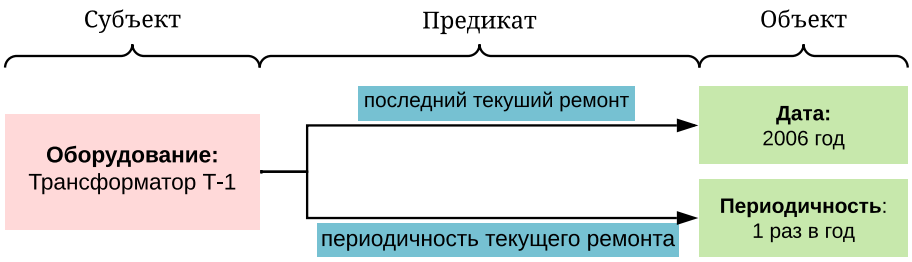
1.2.2. Сложность обработки табличной информации

Как правило, документные таблицы предназначены для восприятия и понимания человеком, но не для интерпретации компьютерной программой. В них отсутствует явная семантика (структура), необходимая для того чтобы компьютерные программы интерпретировали их так, как было задумано авторами или как того требует приложение. Иллюстративный пример приводится на рис. 1.6. Предметный эксперт может принимать некоторые решения, прочитав и проанализировав необходимые ему сведенья об электротехническом оборудовании, представленные в виде документной таблицы (рис. 1.6, *а*), в частности, спланировать его ремонт. Однако исходная информация, доступная компьютерной программе, в лучшем случае является структурой ячеек в редактируемом формате (Excel, HTML или др.), а в худшем — растром, т. е. набором пикселей, не отделенных явным образом от остального изображения страницы документа. Автоматизация принятия экспертных решений, опирающихся на подобные источники данных, нуждается в предварительном извлечении всех необходимых фактов (рис. 1.6, *б*).

Для того чтобы табличная информация могла быть проиндексирована, запрошена и проанализирована компьютерными программами, в общем случае, она должна быть преобразована от неструктурированного к структурированному представлению (базе данных или связанным данным). Несложно заметить, что такое преобразование согласуется с целью *извлечения информации*, которая состоит в получении фактов о имеющихся сущностях и отно-

Оборудование и его элементы	Справочные данные		Паспортные данные	
Трансформатор Т-1	Тип	DEP-1600/5	Дата изготовления	1940
	Заводской номер	406034	Ввод в эксплуатацию	1953
	Завод-изготовитель	GARBE LAHMEYER Германия	Вес полный	6680
	Мощность	1600 кВА	Дата последнего к.р.	2004
	Система охлаждения	естественное масляное	Дата последнего т.р.	2006
	Данные о ремонтах	в соответствии с графиком	Периодичность к.р.	по испытан.
	Данные испытаний	соответствует требованиям РД34.45-51.300-97	Периодичность т.р.	1 раз в год
	Данные о дефектах	отсутсвуют	**ПЕРИОДИЧНОСТЬ**	
			ВВИ	1 раз в 3 года
			ХАРГ	2 раза в год
			Хим. анализ масла	1 раз в год

а



б

Рисунок 1.6 — Иллюстративный пример: имеются паспорта электротехнического оборудования, представленные в виде документных таблиц ($а$); планирование текущего ремонта данного оборудования требует извлечения необходимых фактов из таких таблиц ($б$)

шений между ними из неструктурированного текста [352]. Однако в отличие от такого рода источников, извлечение информации из документных таблиц требует автоматического «понимания» как естественного языка, так и пространственного размещения самого текста в структуре ячеек.

Задаваясь вопросом: будет ли эффективно применение развитого арсенала инструментов извлечения информации напрямую к таблицам, недавние исследования [295, 350] дают отрицательный ответ. N. Milosevic и др. [295] утверждают, что существующие инструменты *интеллектуального анализа текста* непригодны для работы с таблицами, и делают вывод, что следует развивать специализированные средства *интеллектуального анализа табличной информации*. J. C. Roldán и др. [350] отмечают, что средства извлечения информации из веб-страниц общего назначения неэффективны для

веб-таблиц, поэтому предпочтительнее создавать проблемно-ориентированные инструменты.

D. Burdick и др. [104] указывают, что сложность извлечения информации из документных таблиц обусловлена главным образом двумя факторами — разнообразием форм представления и ограниченностью форматов данных. Сложность конкретных реализаций решений АПТ во многом зависит от исходных и целевых форм и форматов представления табличной информации. В одних случаях структура таблиц не доступна вовсе, в других возможно декодировать по крайней мере некоторые ее части, и хотя она может оказаться некорректной, но иногда это позволяет упростить реализацию АПТ. Целевым представлением извлекаемой информации обычно является либо база данных, либо граф знаний. Если в первом случае можно обойтись без привлечения внешних словарей, то в последнем полученную информацию, как правило, требуется сопоставить со связанными открытыми данными[5].

1.2.3. История исследований

В 1992 году S. Srihari и др. [369] впервые определили АПТ как извлечение реляционных данных (отношений) из документных таблиц. Первые исследования по данной тематике стартовали в середине 1990-х [157, 190, 320]. За три последних десятилетия в рамках проблематики АПТ были защищены десятки диссертаций, опубликованы тысячи научных статей и зарегистрированы по крайней мере сотни патентов. Они реферировались, сравнивались и обобщались в ряде обзоров [3, 98, 138, 164, 200, 230, 283, 286, 348, 363, 415, 421].

Наиболее ранние работы рассматривались в обзорах [164, 286, 363, 415]. D. Lopresti и G. Nagy [286] впервые охарактеризовали обработку документных таблиц. D. W. Embley и др. [164] сформулировали основные вопросы, совокуп-

[5] `https://lod-cloud.net`

ность идей и понятий научного направления *обработки таблиц*, включающего АПТ. R. Zanibbi и др. [415] изучили основную литературу по *распознаванию таблиц*, опубликованную с 1990 по 2002 г. A. C. e Silva и др. [363] представили развернутое описание этапов АПТ [222], охватив основную литературу, опубликованную до 2004 г. B. Couasnon и A. Lemaitre [138] проследили дальнейший прогресс, проделанный до 2011 г. Исследования по распознаванию таблиц в PDF-документах были отражены в обзорах [136, 236].

Можно сделать вывод, что два первых десятилетия внимание исследователей в основном уделялось распознаванию таблиц. Вопросы анализа и интерпретации изучались в меньшей степени. Следует отметить, что исследования, проводившиеся до 2010 г., ограничивали этап интерпретации таблиц восстановлением наборов записей без сопоставления с внешними словарями [164, 184, 221, 363]. Первые шаги в сторону семантической интерпретации были предприняты в работах [221, 384]. M. Hurst [221] предложил распознавать онтологические отношения, такие как «*is-a*», «*part-of*», «*unit-is*», и «*quantity-is*». Y. Tijerino и др. [384] впервые начали конструировать онтологии из веб-таблиц.

Последующие достижения отражены в недавних обзорах [98,200,230,283, 348,421] и руководствах [104,150,328]. S. Zhang и K. Balog [421] фокусируются на работах, посвященных сопоставлению веб-таблиц с внешними базами знаний в приложениях информационного поиска, вопрос-ответных систем и конструирования графов знаний. J. C. Roldán и др. [348] обобщают и сравнивают большое количество работ, посвященных извлечению данных из веб-таблиц. S. Bonfitto и др. [98] дают всесторонний обзор современного состояния исследований по данной тематике, включив в рассмотрение также инструменты очистки и трансформации таблиц PK и баз данных.

Начиная с основополагающей работы M. Cafarella и др. [105], значительные усилия стали направляться на развитие методов распознавания и интер-

претации веб-таблиц [348, 421]. Пионерские работы [273, 377] открыли новое направление — семантическую интерпретацию таблиц, при которой извлеченные реляционные данные сопоставляются внешним базам знаний. Последнее десятилетие, с развитием открытых связанных данных, исследователи стали активно применять именно такую интерпретацию. Современные точки зрения [98, 104] рассматривают ее как завершающий этап АПТ.

В последнее время активно развивающиеся языковые модели, в том числе большие[6], визуально-языковые[7] и мультимодальные[8], открыли новые возможности для обработки табличных данных на основе двух подходов: либо настройки предобученных моделей, либо пользовательских «промптов», т. е. инструкций на естественном языке. Обзоры подходов и методов применения таких моделей к задачам АПТ представлены в недавних работах [85, 152, 284, 287, 372, 426]. Вероятно, именно это многообещающее направление составляет сегодня наибольший интерес для новых исследований [102, 130, 153].

1.2.4. Структура совокупности задач

Основополагающая работа M. Hurst [222] выделяет пять этапов АПТ следующим образом: *обнаружение* — определение местоположения таблицы в источнике; *распознавание структуры* — обнаружение ее ячеек; *ролевой анализ* — сопоставление ячейкам функциональных ролей; *структурный анализ* — восстановление связей между ячейками; *интерпретация* — вывод структурированных данных. Приведенная последовательность впоследствии уточнялась в целом ряде более поздних исследований [3, 7, 8, 98, 104, 184, 294, 295, 335, 339, 348, 350, 363]. По сути, они предлагают свои собственные варианты этой концепции.

[6] англ. «*Large Language Models*»

[7] англ. «*Visual Language Models*»

[8] англ. «*Multimodal Large Language Models*»

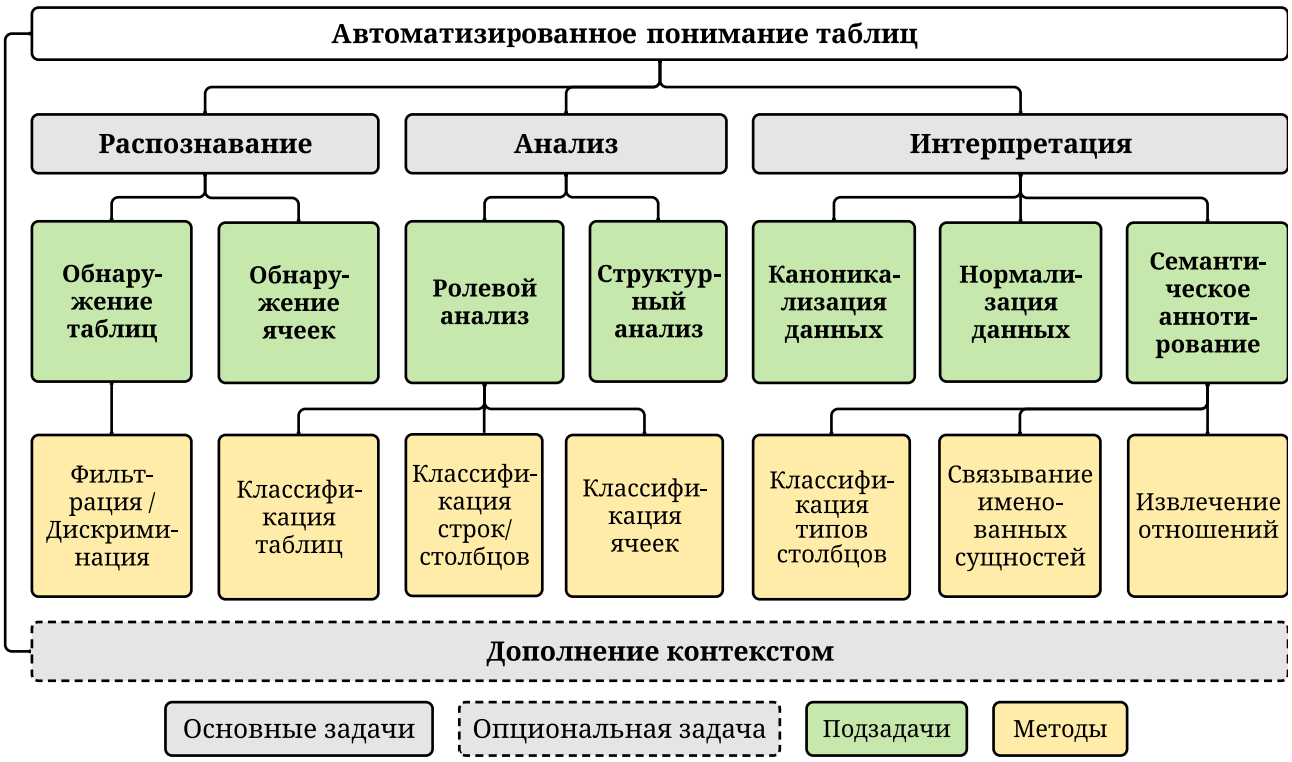


Рисунок 1.7 — Структура совокупности задач АПТ

Следует отметить, что в литературе нет единой точки зрения на структуру совокупности задач АПТ, а используемая терминология по-прежнему нуждается в согласовании и унификации. Некоторые исследователи [222,348,363] рассматривают линейную последовательность задач, другие [104,184,339,421] выделяют два уровня. При этом одни [184, 421] совмещают распознавание и анализ в одну общую задачу, а другие [104, 339], напротив, различают их. В пользу последней точки зрения свидетельствует то, что в действительности, распознавание таблицы имеет самостоятельную ценность, поскольку используется в ряде приложений независимо от анализа. С другой стороны, в некоторых случаях, когда источник предоставляет изолированные и структурно размеченные таблицы, процесс АПТ может вовсе обходиться без распознавания, начинаясь непосредственно с анализа.

Придерживаясь точки зрения [104,339], в настоящем исследовании предлагается собственный вариант иерархической структуры совокупности задач

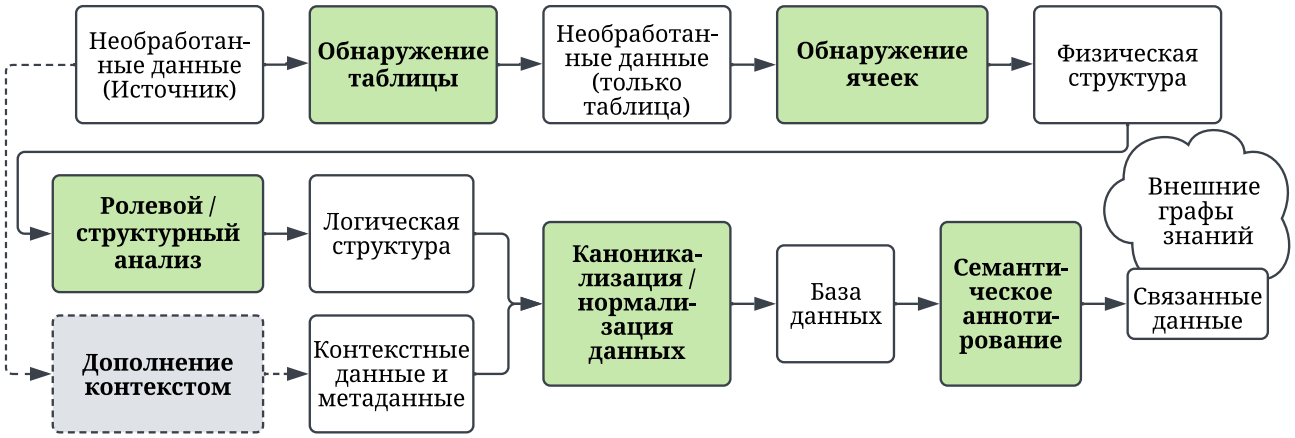


Рисунок 1.8 — Последовательность этапов АПТ: от неструктурированного источника до структурированного представления

АПТ (рис. 1.7). В отличие от постановки M. Hurst [222], предлагаемая структура имеет два уровня: задач и подзадач. На верхнем уровне выделяются следующие задачи: *распознавание* — восстановление физической структуры таблицы; *анализ* — восстановление логической структуры таблицы; *интерпретация* — генерация структурированного представления; *дополнение контекстом* — связывание таблицы с окружающими данными/метаданными в том же источнике. Первые четыре этапа последовательности M. Hurst [222]: *обнаружение*, *распознавание структуры*, *ролевой* и *структурный анализ*, отражены в предлагаемой схеме на уровне подзадач. Пятый этап, *интерпретация*, включен в уровень задач и детализируется вложенной частью, состоящей из *каноникализации/нормализации данных* и *семантического аннотирования*.

Одна из возможных последовательностей этапов АПТ, соответствующая предложенной структуре совокупности задач, показана на рис. 1.8. Этап *обнаружения таблицы* позволяет выявить и считать из источника ту часть необработанных данных, которые представляют исходную таблицу. Этап *обнаружения ячеек*, получая эту часть необработанных данных, восстанавливает физическую структур таблицы. В результате она становится доступ-

ной для редактирования. Этап *анализа* разбирает физическую структуру, для того чтобы воссоздать соответствующую ей логическую часть. Тем самым обеспечивается чтение ее данных в логическом порядке. Этап *дополнения контекстом* опционально обогащает логическую структуру информацией об авторстве, тематике, единицах измерения и пр. Этап *каноникализации/нормализации данных* по логической структуре строит соответствующие исходной таблицы реляционные отношения. Этап *семантического аннотирования* сопоставляет последние с внешними словарями; в результате генерируются графовые представления целевых данных. Перечисленные задачи и методы их решения подробно рассматриваются в разделе 1.3.

1.3. Задачи и методы их решения

Рассматриваются основные задачи АПТ и методы их решения: распознавания (раздел 1.3.1), анализа (раздел 1.3.2), интерпретации (раздел 1.3.3) и дополнения контекстом (раздел 1.3.4).

1.3.1. Распознавание таблиц

Постановка задачи: имеется неструктурированный источник, потенциально содержащий одну или несколько таблиц; требуется извлечь из него физическую структуру (координаты, форматирование и содержимое ячеек) каждой из них. В общем случае, обеспечивается возможность представления результатов в редактируемом формате: HTML, Excel или др.

Рассматривается две взаимосвязанные подзадачи: *обнаружение таблиц* и *обнаружение ячеек*. Первая из них формулируется как выделение той части источника, которая представляет единственную таблицу и ничего другого. Вторая состоит в определении всех частей источника, которые представляют отдельные ячейки одной таблицы. Качество результатов первой подзадачи

может быть улучшено за счет последующего распознавания ее ячеек, и, напротив, предварительное обнаружение таблиц позволяет улучшить результаты второй подзадачи.

Конкретные постановки обеих подзадач зависят от исходного формата данных. Они существенно различаются для нередактируемых (форматов языков описания страниц: PDF, PostScript и др.) и редактируемых (форматов текстовых процессоров: Word, RTF и др.) документов, веб-страниц (HTML) и РК (форматов табличных процессоров: Excel, Sheets и др.). Следует отметить, что любые источники могут быть преобразованы в растровые изображения. Потенциально это позволяет обобщить постановку. Однако неизбежная при таком преобразовании потеря информации потенциально может снизить качество результатов. Рассмотрим методы распознавания таблиц в источниках наиболее популярных форматов.

В нередактируемых документах

НПОД с машиночитаемым текстовым содержимым часто не сопровождаются метаданными о местоположениях и структурах таблиц. Так, неразмеченный PDF-документ физически предоставляет только PDL-инструкции отрисовки текста и графики. Следует отметить, что современная версия формата PDF позволяет аннотировать таблицы с помощью HTML-подобной разметки, но на практике это используется далеко не всегда [308, 387]. Можно выделить три основных подхода к работе с такими НПОД: конвертирование к редактируемому представлению; конвертирование к растровым изображениям; прямой доступ к НПОД без конвертирования.

При реализации первого подхода НПОД автоматически конвертируются в редактируемое представление: ASCII-текст, HTML-разметку или др. Преимущество такого подхода связано с тем, что с помощью имеющегося ПО

конвертирования PDF-документов (например, pdf2text[9], pdf2html[10]) можно перейти к редактируемому представлению, обеспечивающему более простое декодирование по сравнению с исходным форматом. Однако в общем случае такое конвертирование приводит к потере PDL-специфичной информации, которая могла бы использоваться при принятии решений в процессе анализа компоновки страниц НПОД. Более того такое конвертирование может вносить ошибки в целевое представление, например некорректно сформированные слова или абзацы.

Второй подход предполагает предварительную растеризацию НПОД и дальнейшее извлечение таблиц уже из полученных изображений. Следует отметить, что из трех перечисленных подходов этот является наиболее общим и делает возможным применение к задаче сегментации документов более общие техники компьютерного зрения. Однако растеризация приводит к неизбежной потере информации. В отличие от НПОД полученный растр нуждается в OCR-обработке и распознавании характеристик форматирования текста [148].

При третьем подходе текст и графика извлекаются напрямую из исходных НПОД. Для того чтобы получить доступ к низкоуровневым PDL-инструкциям отрисовки графики и текста НПОД могут использоваться различные инструментальные средства декодирования и воспроизведения PDF-содержимого (например, PDFBox[11], iText[12]). Преимуществом этого подхода является возможность использовать PDL-специфичную информацию, включая порядок отрисовки текста в графическом контексте и следы перемещения пера [36, 41]. Основной недостаток данного подхода состоит в ограниченной применимости: его невозможно использовать в случаях, когда таблица пред-

[9] `https://www.xpdfreader.com/pdftotext-man.html`

[10] `http://pdftohtml.sourceforge.net`

[11] `https://pdfbox.apache.org`

[12] `https://github.com/itext`

ставлена в виде растра или с помощью некорректных кодировок встроенных глифов.

Известные методы, в том числе [89, 112, 114, 192, 233, 289, 290, 293, 310, 320, 354, 383, 385, 403, 417, 435], преимущественно ориентированы на изображения документов; в меньшей степени представлены имеющие дело с НПОД (PDF, PostScript и др.) — [36, 41, 77, 172, 198, 203, 211, 280, 312, 313, 319, 334, 335, 337–339, 409, 413]. Традиционно они базировались на правилах и машинном обучении. В настоящее время основное направление их развития связано с применением глубокого обучения, в том числе моделей обнаружения объектов на изображениях, семантической сегментации изображений, а также генеративно-состязательных моделей. Среди основных применяемых подходов можно выделить следующие: нисходящий, восходящий и глубокое обучение. Рассмотрим их подробнее.

Нисходящий подход состоит в том, что сперва выполняется обнаружение ограничивающей рамки некоторой таблицы внутри страницы документа, а затем разделение ее на строки, столбцы и ячейки. Известные методы, имеющие дело с нередактируемыми документами, преимущественно следуют именно этому подходу [202,314,325,354,358,360–362,429,435]. Полагаясь на наличие разграфки, некоторые из них [93,113,172,224] предлагают распознавать линейки, отделяющие таблицу от остального содержимого страницы документа, а также разделяющие ее на ячейки. Однако в общем случае разграфка может быть неполной или вовсе отсутствовать. Поэтому также предлагается анализировать размещение текста и/или пробельного пространства. Для чего одни методы [93,113,172,224] используют правила анализа выравнивания блоков текста, а другие [112, 244, 398] предпочитают «X-Y Cut»-алгоритм [225], который строит проекционные профили блоков текста на осях X и Y.

Восходящий подход, напротив, предполагает, что сперва выполняется обнаружение отдельных ячеек, строк и столбцов напрямую на странице до-

кумента, а затем уже они комбинируются в таблицу [314]. Восходящие методы можно разделить на две части: первые [146, 203, 238, 358, 409] собирают таблицы из столбцов, вторые [197, 279, 313, 321, 363] — из строк. В первом случае сперва собираются блоки текста на основе эвристик [238] или машинного обучения [89, 93], а затем из них формируются столбцы на основе правил анализа выравнивания текста внутри таблицы. Во втором случае строки текста классифицируются на табличные и прочие на основе алгоритмов машинного обучения, в частности, скрытой марковской модели [363], условных случайных полей [321] и метода опорных векторов [279]. Затем табличные строки группируются в таблицы на основе правил анализа выравнивания текста внутри таблицы [197].

Глубокое обучение используется для создания специализированных искусственных нейронных сетей (далее ИНС), выполняющих как обнаружение [78, 80, 111, 187, 205, 213, 217, 234, 272, 314, 325, 343, 354, 360, 374], так и распознавание структуры таблиц [128, 201, 235, 314, 325, 330, 332, 354, 360, 361, 382, 429] в изображениях документов. Они реализуют различные современные архитектуры ИНС, в частности, обнаружение объектов [80, 111, 187, 201, 217, 272, 325, 332, 354, 374, 429], семантическую сегментацию [205, 234, 314], деформируемые свертки [78, 360, 361], «полностью» сверточные [314, 354, 360], графовые [128, 213, 330, 343], генеративно-состязательные [269], рекуррентные [235], расширенные свертки [382], Encoder-Decoder [148] и Encoder-Dual-Decoder модели [432].

Наиболее востребованными среди перечисленных являются архитектуры ИНС обнаружения объектов [79, 282, 414], такие как «Faster R-CNN» [342], «Mask R-CNN» [206], «Cascade Mask R-CNN» [109], YOLO [341], RetinaNet [275] и SSD [281]. Они разработаны для целей определения наличия объекта некоторого класса на изображении и нахождения его границ в виде ограничивающей рамки, ключевых точек или контура. При обнаружении объектов на

изображениях для извлечения признаков используются сверточные нейронные сети, в том числе таких архитектур как VGG [366], ResNet [204], DarkNet [341], ResNeXt [401] и EfficientNet [380]. Их базовые версии предобучены на больших массивах фотографических изображений, таких как ImageNet [147], MS-COCO [274] и «Pascal VOC» [170]. Будучи предназначенными для обнаружения и классификации фотоизображений объектов реального мира, изначально они мало эффективны для работы с изображениями документов. Для того чтобы адаптировать их к рассматриваемой задаче, предлагается использовать технику трансферного обучения, называемую *тонкой настройкой* [187,354]. Примеры адаптации различных архитектур можно найти в следующих работах: «Faster R-CNN» — [80,187,272,354,374], YOLO — [111,217], «Mask R-CNN» — [111], «Cascade Mask R-CNN» — [325], SSD — [111] и RetinaNet — [111].

Подобные архитектуры обеспечивают возможность тонкой настройки весов отдельных слоев предобученных моделей, не изменяя большую часть их слоев. Например, «Faster R-CNN» позволяет дообучить два последних полносвязных слоя, один из которых отвечает за уточнение координат ограничивающей рамки, а другой — за классификацию объекта внутри рамки. Такая тонкая настройка может быть выполнена на небольших наборах данных, включающих от сотен до десятков тысяч примеров изображений страниц документов с эталонной разметкой представленных в них таблиц: UW3/UNLV [359], Marmot [173], ICDAR-2013 [189], «ICDAR-2017 POD» [181], ICDAR-2019 [182], SciTSR [128] и др. Обучение целевых нейросетевых моделей также можно выполнить с помощью более крупных наборов данных, включающих от сотен тысяч до миллионов примеров: TableBank [272], PubLayNet [431], PubTabNet [432] и др.

Последние достижения связаны с применением мультимодальных языковых моделей. Для решения рассматриваемой задачи могут использоваться

мультимодальные модели общего назначения, такие как «GPT-4 Vision» [13], LLaVA[14] и др., а также проблемно-ориентированные мультимодальные модели, в том числе TableVLM [127] и Table-LLaVA [430], способные распознавать структуры ячеек таблиц напрямую в изображениях документов.

В редактируемых документах

Редактируемые форматы дают возможность автоматически выбрать таблицу и считать ее физическую структуру, если она представлена соответствующим объектом в модели документа. Однако может оказаться, что таблица имеет некорректную разметку или вовсе не имеет ее, являясь в действительности вставкой растра или неформатированного текста. В таких случаях, она не может быть декодирована стандартным образом. Основной подход к извлечению таблиц в подобных случаях состоит в конвертировании источников к нередактируемом формату (например, растровым изображениям или расширенным метафайлам [59]).

Неформатированный текст продолжает использоваться в ПО с интерфейсом командной строки, в частности, для вывода табличных данных. Существует ряд методов распознавания таблиц в неформатированном тексте [214, 215, 237, 238, 245, 268, 307, 321, 329, 386]. В основном предлагается распознавать табличные строки [214, 307, 321]. В отличие от изображений, неформатированный текст не требует OCR-обработки. Тем не менее, в тех случаях, когда разграфка представлена с помощью символов псевдографики, иногда оказывается целесообразным сперва выполнить его растеризацию, а затем уже извлекать таблицы из полученного растра.

Следует отметить, что, будучи представленными в виде неформатированного текста, такие таблицы могут обрабатываться с помощью больших

[13] `https://openai-docs.ru/docs/guides/vision`

[14] `https://llava-vl.github.io`

языковых моделей общего назначения: «ChatGPT»[15], «DeepSeek»[16] и др. В частности, они позволяют трансформировать структуру исходных ячеек в форматы JSON/CSV, подходящие для последующего анализа.

В веб-страницах

Веб-страница является документом «Всемирной паутины», распространяемым в формате HTML. Он обеспечивает возможность разметить таблицу специальными тегами: `<table>`, `<tr>`, `<td>` и др. Корректное применение упомянутых тегов потенциально обеспечивает автоматическое декодирование HTML-таблиц, включенных в веб-страницу. Однако на практике далеко не во всех случаях доступная разметка является корректной. Во-первых, упомянутые теги часто используются не как способ кодирования *подлинных таблиц* с реляционными данными, но как средство компоновки веб-страницы [140]. Во-вторых, веб-таблица может быть представлена не специальной разметкой, ориентированной на таблицы, а тегами, обозначающими списки и блоки: `<ul>`, `<li>`, `<div>`, `<span>` и др. [267, 348]. Более того, таблица может оказаться вставкой растра или неформатированного текста. В-третьих, физическая структура может быть неявной, например, таблица может содержать вложенные таблицы, часть ячеек может отсутствовать [183]. J. C. Roldán [348] указывает, что обработка подобных случаев требует их предварительной корректировки. Следует отметить, что веб-таблицы также могут быть представлены в виде информационных ресурсов: CSV-файлов [166, 303] и вики-ресурсов [94, 298]. В отличие от HTML-таблиц, встроенных в веб-страницы, такие изолированные ресурсы обычно могут быть декодированы стандартными средствами. Можно выделить три подхода к распознаванию HTML-таблиц: с помощью объектной модели, рендеринга и растеризации веб-страницы.

[15] `https://openai.com/index/chatgpt`

[16] `https://www.deepseek.com`

В первом случае, предполагается, что HTML-таблица сигнализируется явным образом с помощью HTML-элемента `<table>`. Обнаружение таблицы сводится к установлению ее подлинности: действительно ли она является представлением реляционных данных, а не просто средством компоновки [106, 368, 390]. Распознавание структуры HTML-таблицы обычно пропускается, так как предполагается, что она имеет корректную разметку тегами, обозначающими строки — `<tr>` и ячейки — `<td>`.

Методы, следующие первому подходу, обычно включают: *фильтрацию* на основе правил и *дискриминацию* на основе бинарной классификации. *Фильтрация таблиц* исключает заведомо некорректные случаи: недостаточное количество строк или столбцов, наличие вложенных таблиц, отсутствие текстового содержимого, несоответствие шаблону, непохожесть ячеек и др. [116, 140, 231, 240, 318, 350, 406]. *Дискриминация таблиц* разделяет таблицы на два класса: подлинные (с данными) и неподлинные (без данных) [135,231,368,390]. Известные решения базируются на правилах [106] и техниках машинного обучения, в том числе наивном байесовском классификаторе [135, 390], методе опорных векторов [391], линейном классификаторе Winnow [135], деревьях решений [231].

В некоторых исследованиях [140,160] предлагается использовать детализированные таксономии типов таблиц, которые включают один или несколько подтипов подлинных и неподлинных таблиц. Такой метод называется *классификацией типов таблиц* и служит как для распознавания, так и для анализа таблиц. Более подробно он рассматривается в разделе 1.3.2.

Второй подход предполагает, что веб-таблицы следует распознавать в визуальном представлении веб-страницы, формируемом в результате рендеринга ее исходного кода HTML/CSS[17] [183, 184, 243, 260]. Данная постановка подходит для тех случаев, когда используется некорректная HTML-разметка.

[17]Каскадные таблицы стилей (англ. Cascading Style Sheets).

По сравнению с первым подходом, второй является более общим, поскольку не ограничивается HTML-разметкой представления таблиц `<tr>`/`<td>`.

Методы, следующие второму подходу [135, 179, 183, 184, 267], полагаются на визуальные признаки. Они сперва оценивают ограничивающие рамки элементов веб-страницы, включая списки `<ul>`/`<li>` и блоки `<div>`/`<span>`; затем предсказывают местоположения и физические структуры таблиц, визуально представленных такими элементами.

В третьем подходе предлагается извлекать веб-таблицы из растрового изображения веб-страницы [243]. В таком случае в обработку можно включить веб-таблицы, являющиеся неформатированным текстом или растром [243]. Данный подход исследуется в единственной работе [243].

В рабочих книгах

Рабочие книги являются документами *табличных процессоров*: Excel, Sheets и др. [178, 317]. Они предоставляют *рабочие листы* — пространства ячеек, типизацию данных (числовые, денежные, текстовые и прочие типы ячеек), формулы, фильтры, ссылки, правила условного форматирования и другие функции. В общем случае, диапазон ячеек конкретной таблицы внутри рабочего листа неизвестен, так как она может размещаться где угодно вместе с текстом, рисунками и другими объектами. Более того, физическая структура может быть некорректной [24, 29]. Например, две или более соседних физических ячеек могут визуально представлять одну логическую ячейку и, наоборот, одна физическая ячейка может быть прочитана человеком как несколько логических. Описанные особенности препятствуют автоматическому распознаванию физической структуры таблицы.

Обнаружение таблиц в рабочих листах сводится к распознаванию однотипных диапазонов ячеек и их взаимосвязей [151, 158, 159, 249, 252, 253]. По

сути, выполняется анализ всего рабочего листа, в значительной степени пересекающийся с задачей анализа таблиц, рассматриваемой в разделе 1.3.2. Распознавание структуры таких таблиц состоит в коррекции тех случаев, когда их физическая структура и визуальное представление ячеек не соответствуют друг другу [24,29]. Методы, ориентированные на данный вид источников, мало представлены в литературе. Существует ряд методов обнаружения таблиц в рабочих листах на основе правил [158], машинного [248, 249, 251] и глубокого обучения [151, 159], генетических алгоритмов [252], а также метод коррекции специфических случаев некорректной физической структуры на основе правил [24, 29].

1.3.2. Анализ таблиц

Данная задача направлена на восстановление отсутствующей логической структуры таблицы по доступной физической структуре [166, 299, 302, 355]. Как правило, исходная физическая структура таблицы записана в виде HTML-разметки, PK Excel или CSV-файла. Следует отметить, что язык HTML обеспечивает функциональную разметку таблиц, позволяющую пометить ячейки как `<th>` — заголовки и `<td>` — данные, а также группы строк как `<thead>` — шапку, `<tbody>` — тело, `<tfoot>` — подвал. J. C. Roldán [348] указывает, что на практике такая разметка используется редко и, как правило, любая ячейка маркируется тегом `<td>`.

В отличие от HTML, форматы PK и вовсе не предоставляют никаких стандартных возможностей описания логической структуры таблиц. В лучшем случае из источника можно считать типы данных ячеек и функции (формулы, фильтры и пр.), которые могут помочь распознать логическую структуру таблицы. Кроме того, ни HTML, ни форматы PK не позволяют кодировать структуру ячейки и связи между функциональными элементами.

Из недавних обзоров [98, 348] следует, что в целом исследователи пренебрегают специфичными свойствами этих форматов, полагаясь только на наличие структуры строк, столбцов и ячеек без функциональной разметки.

Очевидно, что анализ плоских (реляционных) таблиц может сводиться к ряду тривиальных трансформаций. В случае же многомерных таблиц с многосторонней компоновкой, где могут присутствовать иерархии заголовков, вложенные таблицы, структурированные ячейки и пр., предсказание логической структуры становится одним из наиболее сложных вызовов АПТ. Кроме того, последнее требует использования более подходящего формата выходных данных, который может быть основан на известных нотациях логической структуры таблицы [7, 223, 294, 397]. Анализ таблиц принято делить на две подзадачи: *ролевой* и *структурный*. Рассмотрим их подробнее.

Ролевой анализ

Традиционная постановка данной подзадачи предполагает, что каждой ячейке сопоставляется один из заданных типов [7, 76, 117, 174, 339, 350, 356]. Некоторые исследования [117,339,356] предлагают делить таблицу на несколько функциональных регионов, таких как шапка, боковик, тело и подвал. Другие [76, 174] присваивают функциональные типы (заголовков и данных) строкам и столбцам. Еще одна работа [350] предлагает функциональное разделение ячеек на два типа — метаданные и данные.

Многие методы ролевого анализа [99, 116, 165, 166, 184, 231, 240, 276, 292, 294,300–303,399,399,406] основаны на правилах. В частности, они используют следующие способы: заданные шаблоны структуры таблицы [99, 166, 184, 240, 294] (например, первая строка является заголовком [99]); заданные свойства структуры таблицы (например, наличие так называемых *критических ячеек*, определяющих координаты функциональных табличных регионов в про-

странстве строк и столбцов) [165, 166, 292, 300–303, 356]; функциональную разметку (теги `<th>`, `<thead>` и пр.) HTML-таблиц [231,276,294,399]; оценку сходства соседних строк/столбцов по их визуальному представлению [406] и содержимому (строкам, числам, именованным сущностям) [116, 240, 266], в том числе с использованием метрик семантического сходства [240]; оценку сходства соседних ячеек [294]; сопоставление содержимого с шаблонами, ключевыми словами и регулярными выражениями [231]. При этом методы могут использовать как предварительно заданные параметры [240, 294, 406, 411], так и настраиваемые пользователями [116, 231, 240, 276]. Основные методы ролевого анализа включают следующие: классификация типов ячеек, строк/столбцов и таблиц.

Классификация типов ячеек [185, 247, 394] позволяет распознать группы однотипных ячеек [159, 186, 248, 375], каждая из которых образует некоторый функциональный регион (например, боковик, шапку, тело и подвал). Известные методы классификации типов ячеек основаны на эвристиках [74], деревьях решений (CART, C4.5, случайный лес) [247, 251], методе опорных векторов [247, 251], рекуррентных ИНС (архитектуры LSTM/Bi-GRU) [159, 185, 186], графовых ИНС [159], нейросимволических [375] и языковых моделях BERT [394]. Следует отметить, что в случае РК классификация типов ячеек также может использоваться для аннотирования рабочего листа в целом, а распознанные группы однотипных ячеек — для обнаружения таблиц [248, 249, 252, 253] (раздел 1.3.1).

Классификация типов строк/столбцов присваивает один из заданных типов (например, *header*, *data* и *blank*) строкам/столбцам [117, 174]. Методы классификации типов строк/столбцов базируются на эвристиках [76, 406], вероятностных моделях [411], деревьях решений [174], методе опорных векторов [174], условных случайных полях [117]. Данная классификация позволяет обнаружить заголовки таблицы как последовательности соседних строк или

	Lake	Area (km2)	Area (sq mi)
1	Windermere	14.73	25324
2	Rutland Water	31686	43556
3	Kielder Water	10.00	31472
4	Ullswater	33451	16132
5	Grafham Water	46539	15373
6	Bassenthwaite Lake	12540	44714
7	Derwent Water	12540	44714

а

Government	
• Type	Strong mayor–council
• Body	New York City Council
• Mayor	Eric Adams (D)
Area	
• Total	472.43 sq mi (1,223.59 km2)
• Land	300.46 sq mi (778.18 km2)
• Water	171.97 sq mi (445.41 km2)
Elevation	33 ft (10 m)

б

Handedness / Sex	Right-handed	Left-handed	Total
Male	43	9	52
Female	44	4	48
Total	87	13	100

в

Рисунок 1.9 — Компоновочные типы таблиц K. Braunschweig [100]: *список* (*а*), *запись* (*б*) и *матрица* (*в*); примеры таблиц взяты из Википедии

столбцов [166, 174, 231, 303]. В некоторых постановках предлагается сегментация таблицы на два или более функциональных региона [166, 300, 327].

Классификация типов таблиц нацелена на сопоставление таблице одного из заданных классов [140, 160, 263, 264]. Литература предлагает несколько таксономий типов таблиц [100, 140, 160, 263, 264, 349]. Их сравнительный обзор представлен в работе [21]. Известные таксономии включают от 3 до 13 типов таблиц, например, K. Braunschweig [100] предлагает следующие типы: *список*, *запись* и *матрица*, как показано на рис. 1.9. При этом реляционные таблицы, в зависимости от ориентации записей, могут быть *вертикальными* или *горизонтальными* [264]. Следует отметить, что некоторые классификации являются многоуровневыми [140, 160]. На верхнем уровне они включают два типа, а именно подлинные и неподлинные случаи. На последующих вводятся соответствующие им подтипы. Как упоминалось в разделе 1.3.1, подобные таксономии могут также использоваться для обнаружения веб-таблиц.

Классификация типов таблиц [140,160,194,263,264,311,350] позволяет вывести функциональные типы ячеек на основе правил. Например, реляционная таблица с горизонтальной ориентацией по классификации O. Lehmberg [264] обладает такой компоновкой, при которой верхняя строка является заголовком, а все последующие строки — записями. Вывод функциональных типов ячеек в таком случае тривиален. Известные методы классификации типов

таблиц основаны на правилах [294, 406], деревьях решений [140, 160, 263, 264], кластерном анализе [350], ИНС [194, 311, 350].

Следует отметить, что современные ИНС, в том числе: TaBERT [410], TaPas [209] и TUTA [394], предназначенные для классификации типов ячеек и таблиц, представляются многообещающими. Однако среди свободно доступных из них нет таких, которые могли бы извлекать данные из любых таблиц одинаково эффективно. Как правило, они настраиваются с использованием доступных источников данных (например, таблиц Википедии [261] или финансовых отчетов [416]). В результате некоторые из них экспериментально показывают высокую эффективность при работе с таблицами англоязычной части Википедии, а другие хорошо справляются с финансовыми отчетами. Так, значение макросредней F_1-меры, показанное TUTA [394] — лучшим решением на основе глубокого обучения — на задаче классификации типов ячеек, достигает не более 79–92 % на разных тестовых коллекциях [416]. На практике требуется дополнительная настройка подобных моделей для работы с источниками данных, специфичных для решаемой задачи.

Структурный анализ

Данная подзадача нацелена на извлечение связей между функциональными элементами таблицы (рис. 1.10). В литературе выделяются следующие варианты ее постановки: выводятся связи между разнотипными регионами таблицы [294,295,327,375]; извлекаются иерархические связи внутри ячеек заголовка [117,394]; выводятся связи между атомарными значениями ячеек [7]. Результат структурного анализа — восстановленный логический порядок чтения данных таблицы [335, 339], как показано на рис. 1.10.

Большая часть известных методов структурного анализа ориентирована на широко распространенные типы таблиц [348]. Предполагается, что,

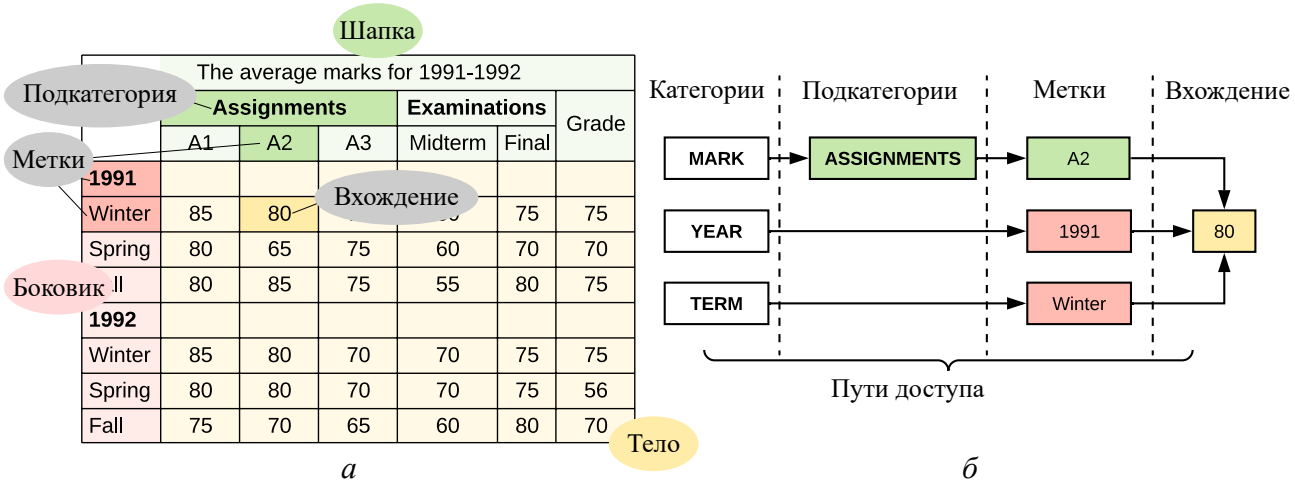


Рисунок 1.10 — Восстановление логического порядка чтения: исходная таблица (скопирована из работы [397]) (*а*); порядок чтения, соответствующий значению данных «*80*», составлен из набора так называемых *путей доступа* [335, 339] (*б*)

зная тип таблицы, можно автоматически вывести связи между разнотипными ячейками или функциональными элементами на основе правил [294, 350, 406]. В случае плоских таблиц, автоматический вывод связей между ячейками или регионами может быть тривиальным. Например, предполагается, что в таблице типа запись (рис. 1.9, *б*) с вертикальной ориентацией левый столбец — заголовок, правый столбец — единственная запись, в этом случае вывод связей сведется к построчному сопоставлению ячеек. Подобные, так называемые наивные, решения приводятся в целом ряде работ [99, 116, 129, 163, 276, 399]. В частности, они ориентированы на сущностные таблицы [116], списки с горизонтальной [99, 129, 163, 276, 399] и вертикальной ориентацией записей [99, 276].

Более сложные случаи могут включать чередования строк/столбцов разных функциональных типов [411], структурированные ячейки [406], многостороннюю компоновку [294, 350], иерархию заголовка (боковика) [336, 339]. Соответствующие работы предлагают дополнительно использовать шаблоны структуры таблиц [411], шаблоны структуры заголовков на основе описания свойств компоновки, форматирования и содержимого [336, 339], шабло-

ны структуры ячейки на основе описания ее естественно-языкового наполнения [406].

1.3.3. Интерпретация таблиц

Данная задача направлена на интерпретацию логической структуры таблицы, в результате которой исходная таблица может быть преобразована в целевое структурированное представление. В литературе рассматривается три связанных подзадачи: *каноникализация* и *нормализация данных*, *семантическое аннотирование таблиц.* Рассмотрим их подробнее.

Каноникализация данных

Предполагается, что таблица должна быть преобразована к некоторому набору записей в *канонической форме* [167,384], соответствующей отношению в терминах реляционной модели данных [157, 221]. Решение этой подзадачи тривиально в том случае, когда исходная таблица уже является реляционной. Однако в других случаях такое преобразование нуждается в интерпретации логической структуры таблицы. Ряд исследований [7, 8, 165, 166, 301, 335, 339] предлагают методы такой интерпретации, ориентированные на многомерные таблицы с многосторонней компоновкой.

Наиболее проработанный способ каноникализации данных многомерной таблицы с многосторонней компоновкой обеспечивается известной моделью X. Wang [397]. Будучи изначально предназначенной для создания и редактирования именно таких таблиц, данная модель получила широкое применение в АПТ [7, 8, 165, 166, 301, 335, 339] как способ представления логической структуры таблицы, приводимой к канонической форме.

Модель X. Wang [397] предполагает, что таблица может иметь иерархическую схему данных с разделением категорий на подкатегории. Пример

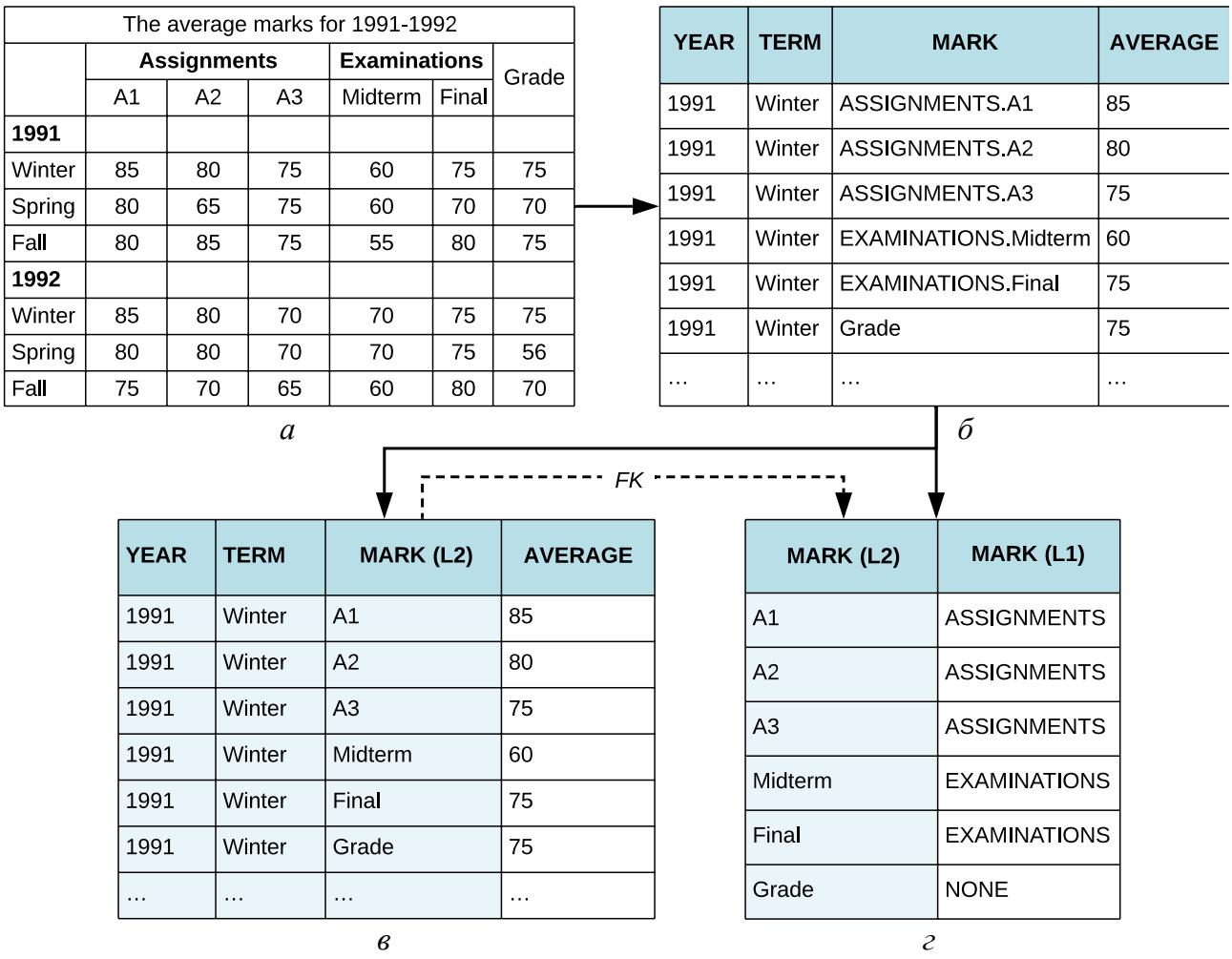


Рисунок 1.11 — Каноникализация данных: исходная таблица (скопирована из работы [397]) (*а*) и ее каноническая форма (*б*). Пример нормализации данных: приведение к третьей нормальной форме (*в*, *г*)

такого случая показан на рис. 1.11, *а*; соответствующая каноническая форма — на рис. 1.11, *б*. В данном примере имеется три категории верхнего уровня: *YEAR*, *TERM* и *MARK*; категория *MARK* включает две подкатегории: *ASSIGNMENTS* и *EXAMINATIONS*. Интерпретация логической структуры подобных случаев должна по крайней мере обеспечить разделение значений измерений на категории и подкатегории.

Модель X. Wang [397] предполагает, что значения измерений, называемые *метками*, принадлежат категориям, а значения данных, называемые *вхождениями*, адресуются метками. Набор *путей доступа* от категорий верх-

него уровня до меток задает логический порядок чтения данных [335, 339]. Каждое вхождение адресуется единственным путем в каждой категории верхнего уровня. Например, в таблице, показанной на рис. 1.11, *а*, доступ к вхождению *80* обеспечивается только тремя путями, а именно *YEAR.1991*, *TERM.Winter* и *MARK.ASSIGNMENTS.A2*. Логическая структура таблицы формально определяется как функция δ: $A \to E$, где A — набор путей доступа, E — набор вхождений. В приведенном примере функция δ отображает пути доступа к вхождениям следующим образом:

$$\delta(\{YEAR.1991, TERM.Winter, MARK.ASSIGNMENTS.A1\}) = 85,$$
$$\delta(\{YEAR.1991, TERM.Winter, MARK.ASSIGNMENTS.A2\}) = 80,$$
$$...$$
$$\delta(\{YEAR.1992, TERM.Fall, MARK.Grade\}) = 70.$$

Данное отображение может быть выражено в виде канонической формы (рис. 1.11, *б*). Следует отметить, что категория не обязательно является именованной, метки могут группироваться в анонимные категории [7, 166, 339]. Опционально каноникализация данных может задействовать внешние словари для именования таких категорий [8].

Следует отметить, что каноникализация данных также обеспечивается некоторыми средствами интерактивной первичной обработки данных, в том числе: Wrangler [232], Foofah [229] и TableCanoniser [402]. Они применяют пользовательское программирование операции трансформации таблиц с помощью проблемно-ориентированных языков. Некоторые из них предлагают автоматический синтез таких программ из пользовательских примеров ручной трансформации [91, 191, 199, 229]. Современные большие языковые модели общего назначения (ChatGPT, DeepSeek и др.) и проблемно-ориентированные (Table-GPT [271], TableLLM [428] и др.) также способны выполнять

такие операции трансформации на основе пользовательских «промптов» на естественном языке, дополненных примерами исходных и целевых данных.

Нормализация данных

Данная подзадача состоит в преобразовании ненормализованной реляционной таблицы к *нормальной форме* [167]. Пример нормализации данных показан на рис. 1.11, *б–г*. Исходные реляционные таблицы часто являются ненормализованными. Такая таблица может описывать несколько взаимосвязанных сущностей одновременно [99]. Однако семантическая интерпретация [99] требует того, чтобы данные соответствовали *третьей нормальной форме*. Методы извлечения функциональных зависимостей [395] и классификации столбцов [99] могут применяться для автоматической нормализации таблицы. Braunschweig K. [99] предлагает использовать методы семантического аннотирования таблиц (раздел 1.3.3), для того чтобы идентифицировать подмножества столбцов, каждое из которых описывает сущности отдельного класса.

Более того, исходные данные могут содержать служебную, ошибочную и избыточную информацию, следовать разным стандартам и соглашениям именования. Часто нормализация должна включать очистку данных, в том числе удаление вспомогательной разметки, исключение дубликатов записей, разделение составных атрибутов, унификацию единиц измерения [139, 297, 346]. Наиболее применяемым подходом к очистке данных является пользовательское программирование [232, 333]. Реализация данного подхода может состоять в том, что пользователь обеспечивается предметно-ориентированным языком, позволяющим ему запрограммировать правила очистки данных и их трансформации к нормальной форме, например OpenRefine GREL [389] или TranSheet [220]. Ряд исследований предлагает методы по автоматическому

синтезу программ очистки данных и трансформации таблиц по пользовательским примерам [82, 91, 191, 199, 229, 232, 333]. Подобные решения предлагают интерактивные пользовательские интерфейсы для коррекции таблиц после трансформации в полуавтоматическом режиме [232, 333].

Семантическое аннотирование

Данная постановка ставит своей целью сопоставление схемы и данных таблицы с внешними графами знаний [273,377,418,419], например, DBpedia[18] [139, 346], Wikidata[19] [162], YAGO[20] [95], Probase[21] [396]. *Граф знаний* (далее ГЗ) является представлением коллекции взаимосвязанных сущностей. Введем следующие обозначения: *ГЗ-сущность* — описание некоторого объекта реального мира, события или абстрактного понятия; *ГЗ-класс* — множество сущностей, обладающих общими свойствами; *ГЗ-экземпляр* — ГЗ-сущность, принадлежащая некоторому ГЗ-классу; *ГЗ-тип данных* — тип литеральных значений (чисел, строк, ссылок и др.); *ГЗ-свойство* — предикат, описывающий некоторое отношение субъекта (ГЗ-сущности) и объекта (ГЗ-сущности или литерала). Пример семантического аннотирования таблицы с помощью открытой графовой базы знаний Wikidata показан на рис. 1.12.

Постановка данной подзадачи предполагает, что она применима только к реляционным таблицам однотипных сущностей [99,427]. Такая таблица соответствует отношению в третьей нормальной форме. Каждая запись представляет единственную сущность одного класса, каждое поле — некоторый атрибут сущности. Более того, она обязательно имеет так называемый *предметный столбец*, содержащий упоминания сущностей одного класса [99,265,427]. Следует отметить, что данная концепция близка понятию *набора сущностей*

[18] `https://www.dbpedia.org`

[19] `https://www.wikidata.org`

[20] `https://yago-knowledge.org`

[21] `https://concept.research.microsoft.com`

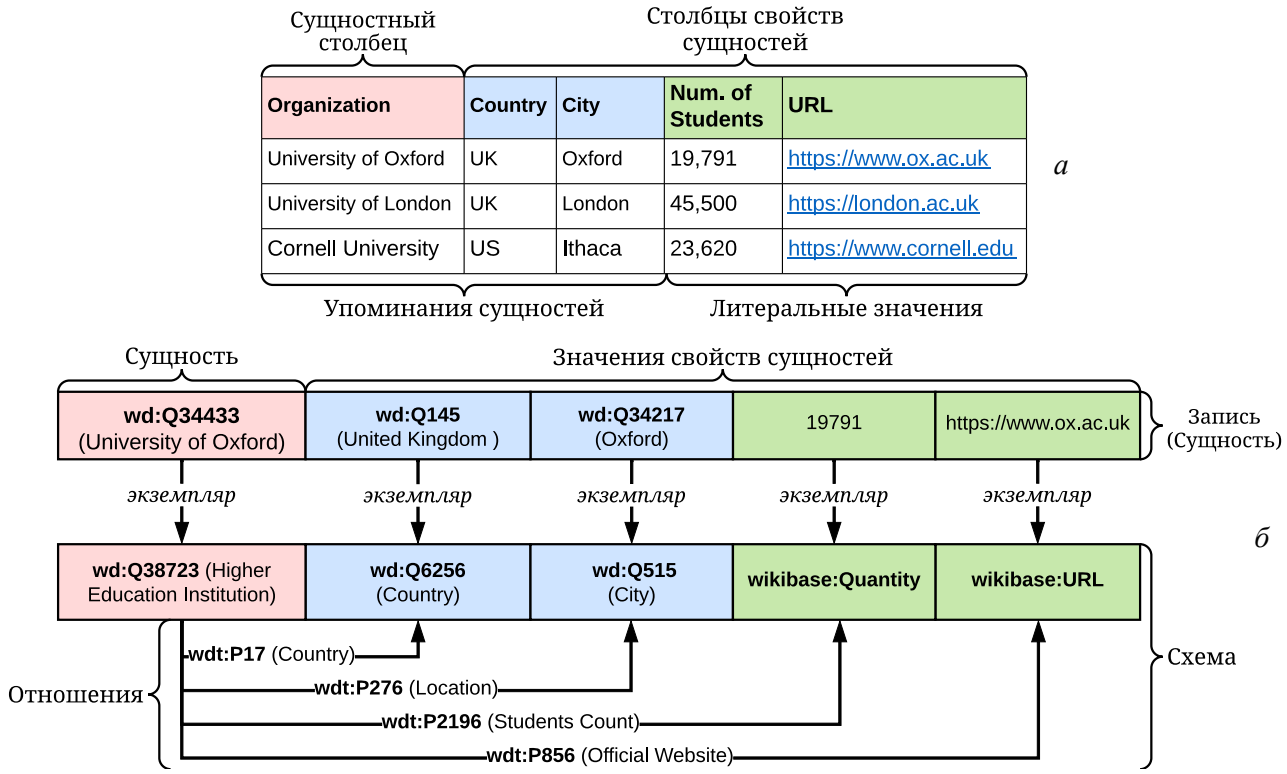


Рисунок 1.12 — Семантическое аннотирование таблицы: реляционная таблица однотипных сущностей ($а$); сущности, классы и свойства из базы знаний Wikidata, сопоставленные значениям, полям и отношениям в данной таблице ($б$)

в модели сущность–связь P. Chen [115]. Согласно S. Zhang и K. Balog [421] методы семантического аннотирования включают следующие: связывание именованных сущностей, идентификация типа столбца и извлечение отношений. Рассмотрим их более подробно.

Связывание именованных сущностей в контексте рассматриваемой задачи состоит в сопоставлении упоминаний сущностей из таблицы с соответствующими ГЗ-сущностями [95]. Одна сущность может быть выражена в тексте разными упоминаниями и, напротив, одно упоминание может соответствовать разным сущностям. Связывание именованных сущностей включает устранение неоднозначности упоминаний сущностей, используя их контекст.

Классификация типов столбцов направлена на сопоставление схемы исходной таблицы с базой знаний. Предполагается, что столбцу, содержащему упоминания сущностей (личностей, географических мест, организаций и др.),

должен быть сопоставлен некоторый ГЗ-класс. Так, предметный столбец всегда ассоциируется только с ГЗ-классом [259, 419]. Когда столбец хранит литеральные значения (числа, даты, координаты и др.), он привязывается к соответствующему ГЗ-типу данных [378].

Извлечение отношений нацелено на отображение каждой пары столбцов в ГЗ-свойства, представляющие субъектно-объектные отношения между ГЗ-сущностями в базе знаний [168, 388]. При этом рассматриваются только пары столбцов вида (S, P), где S является предметным, а P — атрибутом, функционально зависимым от S.

Таким образом, связывание именованных сущностей позволяет аннотировать данные, а идентификация типов столбцов и извлечение отношений — схему данных. Формируемая в результате семантическая аннотация может сохраняться в виде связанных данных в форматах RDF/OWL [81, 139, 297]. Извлеченные отношения выводятся в виде RDF-триплетов — выражений вида $\langle s, p, o \rangle$, где субъект s — сущность, упомянутая в предметном столбце S, предикат p — это свойство сущности s, сопоставленное паре столбцов (S, P), а объект o — значение свойства p сущности s, извлеченное из столбца P. В таком формате извлеченные данные могут подвергаться дальнейшей высокоуровневой интерпретации, например, SPARQL[22]-запросам.

Известные методы семантического аннотирования таблиц основаны на применении следующих техник: поиск сущностей в графе знаний; векторное представление сущностей; отображение онтологий. Рассмотрим их подробнее.

Поиск сущностей в графе знаний выполняется на основе SPARQL-запросов, передаваемых для обработки поисковым сервисам, таким как DBpedia SPARQL Endpoint[23] или Wikidata Query Service[24]. Поисковый запрос состав-

[22] `https://www.w3.org/TR/sparql11-query`

[23] `https://dbpedia.org/sparql`

[24] `https://query.wikidata.org`

ляется из упоминаний сущностей, содержащихся в текстовом наполнении ячеек. Результатом выполнения SPARQL-запроса является выборка релевантных ГЗ-экземпляров. Поскольку каждое упоминание необходимо связать с единственным ГЗ-экземпляром, среди релевантных выбираются те, которые будут сопоставлены упоминаниям сущностей. Известные методы устранения неоднозначности [95, 162, 347, 423] следуют итеративному подходу к выбору ГЗ-экземпляров, при котором на каждом шаге выбор уточняется до полного согласования ГЗ-экземпляров, сопоставленных упоминаниям как записей, так и схемы. Для аннотирования схемы таблицы некоторые исследователи предлагают также задействовать заранее обученные классификаторы типов столбцов, например, ColNet [122, 123].

Векторное представление сущностей позволяет закодировать некоторый набор ГЗ-экземпляров в виде векторной модели с использованием известных алгоритмов, таких как RDF2Vec [345], KGloVe [133] или Wikipedia2Vec [404]. Сформированная векторная модель позволяет использовать метрики семантического сходства [433, 434] и родства сущностей [323], чтобы ранжировать кандидатные ГЗ-экземпляры по релевантности к связываемому упоминанию сущности в ячейки исходной таблицы. Примеры использования векторных представлений сущностей можно найти в ряде исследований [162, 288, 378, 436].

Отображение онтологий предполагает, что из реляционной таблицы генерируется онтология, которая затем сопоставляется с целевой онтологией [161]. Сопоставление онтологий может выполняться с помощью известных техник, таких как LogMap [227] или PARIS [371]. Данный подход позволяет аннотировать как записи, так и схему [162].

Методы также могут быть гибридными, т. е. совмещающими две или более техники. V. Efthymiou и др. [162] экспериментально показали, что гибрид-

ный подход, в котором комбинируются сервисы поиска сущностей и векторные представления ГЗ, позволяет получить наиболее эффективное решение.

Вопросы семантического аннотирования многомерных таблиц в литературе практически не рассматриваются. Можно привести единичные работы [154, 155]. Очевидно, что многомерная таблица (рис. 1.11, *а*), представленная как одна или несколько связанных реляционных таблиц (рис. 1.11, *б*–*г*), может подвергнуться семантической аннотации с использованием внешних словарей. Когда метки содержат упоминания сущностей, их можно связать с соответствующими ГЗ-сущностями. Категории таких меток могут быть связаны с ГЗ-классами. Категории литеральных меток и значений данных соответствуют ГЗ-типам данных. Можно сопоставить функциональные зависимости между ключом и столбцами значений данных с ГЗ-свойствами. Несмотря на то, что качество извлечения фактов из многомерных таблиц может быть улучшено за счет подобной семантической аннотации, случай многомерных таблиц фактически остается неохваченным текущими исследованиями.

1.3.4. Дополнение контекстом

Некоторые источники данных предоставляют не только таблицы, но и их контекст (авторство, назначение и пр.), скрытый в окружающем тексте (заголовках, примечаниях и пр.). Данная задача направлена на извлечение дескриптивных метаданных, например, тематики, единиц измерения, времени и места, к которым относятся табличные данные [86, 277]. Предполагается, что подобные метаданные должны дополнять извлекаемые таблицы, делая их более подходящими для целей АПТ.

В настоящий момент вопросы извлечения метаданных о таблицах из окружающего контекста остаются малоизученными. В литературе удалось найти лишь несколько методов [196, 242, 277, 412, 418, 419]. M. Yoshida и др.

[412] предлагают метод поиска единиц измерения в страницах Википедии и сопоставление их с данными вики-таблиц. Z. Zhang [418, 419] использует семантическую разметку (аннотации RDFa/Microdata), которая изначально вставляется в веб-страницы для поддержки семантического индексирования в поисковых системах. В частности, такие аннотации могут предоставлять дополнительные атрибуты сущностей, перечисленных в веб-таблицах. D. H. Kim и др. [242] сопоставляют заголовок таблицы с окружающим текстом на основе правил с использованием техник обработки естественного языка. B. Hancock и др. [196] собирают описание HTML-таблицы из заголовков разделов — `<h>` тегов, подписи — `<caption>` тег, а также заголовков столбцов — `<th>` тегов.

1.4. Тесты производительности и коллекции данных

Решения распознавания таблиц в НПОД могут оцениваться и сравниваться количественно с помощью известных методик [173,188,216,359,365,393] и тестов производительности: SciTSR[25], TableBank[26] [272], PubTabNet[27] [432]. На их основе регулярно проводятся соревнования академических и коммерческих решений [131, 132, 182, 189, 431, 432]. Соответствующие конкурсные наборы данных включают в себя две части: одну для предварительной настройки параметров соревнующихся решений (например, могут быть обучены модели, выбраны алгоритмы, изменены пороги и т. д.), другую для их тестирования. Следует отметить, что наиболее современные из доступных наборов эталонных данных [149,431,432] собраны в основном из научных статей, в том числе узкой предметной направленности, например, биомедицинской литературы по неврологическим заболеваниям [75]. Несмотря на наличие устоявшихся соревновательных тестов производительности, их развитие до сих пор остается

[25] `https://github.com/Academic-Hammer/SciTSR`

[26] `https://doc-analysis.github.io/tablebank-page`

[27] `https://github.com/ibm-aur-nlp/PubTabNet`

актуальным. В частности, Z. Zhang и др. [424] экспериментально показывают необходимость совершенствования самих методик оценки производительности.

Вопросы оценки производительности методов анализа таблиц остаются малоизученными. Исследования сфокусированы на создании корпусов веб-таблиц, собранных из архивных срезов «Всемирной паутины», в том числе WDC[28] и DWTC[29], и коллекций PK, в том числе WEB[30] и Fuse[31]. Подобные корпуса и коллекции могут использоваться для создания тестов производительности методов анализа таблиц. Однако существующее предложение ограничено наборами эталонных данных, опубликованными в качестве материалов специализированных методов анализа таблиц: Troy200[32] [166], SAUS[33] [117,119], DECO[34] [250], TabbyXL[35] (на основе SAUS[36]) [6], CIUS/DeEx/SAUS[37] [185]. Перечисленные материалы предоставляют специфическую аннотацию, ориентированную на постановку задачи того исследования, которое они сопровождают, например, разметку критических ячеек в терминах работы [166] или строк рабочих листов вида: *BLANK*, *TITLE*, *HEADER* и *DATA* [117,119]. Они мало пригодны для сравнительной оценки разных решений анализа таблиц.

Исследователи в области семантического аннотирования таблиц разрабатывают так называемые *золотые стандарты* сопоставления веб-таблиц с графами знаний общего назначения: DBpedia, Wikidata и др., например,

[28]Web Data Commons – Web Table Corpora, `http://webdatacommons.org/webtables`

[29]Dresden Web Table Corpus, `https://wwwdb.inf.tu-dresden.de/misc/dwtc`

[30]Senbazuru Spreadsheet Dataset, `http://dbgroup.eecs.umich.edu/project/sheets/datasets`

[31]Fuse Spreadsheet Corpus, `https://static.barik.net/fuse`

[32]`https://tc11.cvc.uab.es/datasets/Troy_200_1`

[33]`https://dbgroup.eecs.umich.edu/project/sheets/datasets`

[34]`https://github.com/ddenron/deco_dataset`

[35]`https://github.com/tabbydoc/tabbyxl/wiki/performance-evaluation`

[36]`https://github.com/tabbydoc/tabbyxl/wiki/SAUS`

[37]`https://github.com/majidghgol/TabularCellTypeClassification`

G. Limaye [273], T2Dv2[38] [347], T2 [144], WikipediaGS[39] [162]. В ряде исследований оценка качества семантического аннотирования проводится на основе корпусов таблиц, таких как WDC [347], WikiTables[40] [95] и GitTables[41] [218]. Начиная с 2019 года, в рамках конференции ISWC ежегодно проводится соревнование SemTab[42] [145], сравнивающее современные решения отдельных этапов, а именно семантического аннотирования ячеек, столбцов и пар столбцов на синтетических [228] и реальных примерах [144, 218].

В настоящее момент не существует тестов производительности комплексных решений АПТ, но предлагаются некоторые методики и наборы эталонных данных, предназначенные для оценки производительности решений отдельных задач: распознавания, анализа и семантической интерпретации таблиц, ориентированных на основные источники данных. S. Bonfitto и др. [98] заключают, что сегодня существует необходимость разработки общего теста производительности комплексных решений АПТ. Также особый интерес представляет развитие тестовых коллекций с учетом национальной (например, русскоязычной части Википедии [177]), предметной (например, корпуса нормативных документов [87]) и организационной (например, электронной переписки корпорации Enron [207]) специфики.

1.5. Области применения

Задачи АПТ имеют несколько важных приложений. Связи между ними схематично изображены на рис. 1.13. Приложения обработки документов [97], включая анализ компоновки, ввод и доступность, тесно связаны с процессом

[38] http://webdatacommons.org/webtables/goldstandardV2.html

[39] https://doi.org/10.6084/m9.figshare.5229847.v1

[40] http://websail-fe.cs.northwestern.edu/TabEL

[41] https://gittables.github.io

[42] https://www.cs.ox.ac.uk/isg/challenges/sem-tab

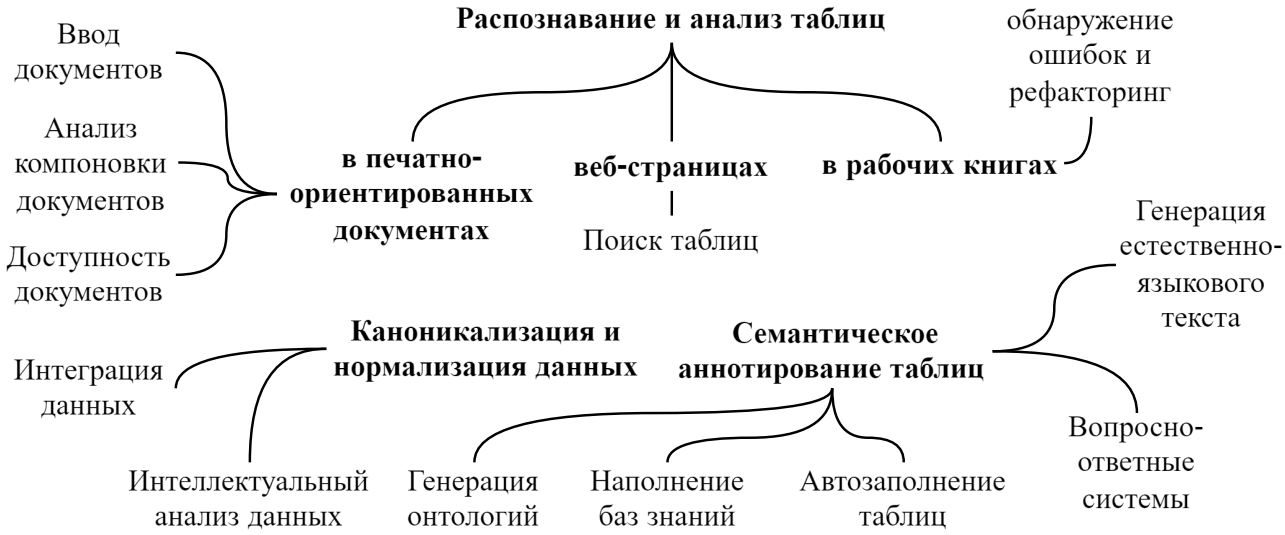


Рисунок 1.13 — Области применения АПТ

распознавания таблиц [97, 393, 431]. Они требуют конвертирования нередактируемых документов (скан-копий, PDF и др.) к редактируемому формату (HTML/XML). В результате документные таблицы становятся доступными для редактирования и отображения на различных устройствах [175]. Последующий анализ делает их доступными для считывания данных в логическом порядке, понятном человеку. В частности, документная доступность обеспечивает автоматическое речевое озвучивание; в такой форме они могут восприниматься незрячими пользователями [176].

Поиск таблиц выбирает таблицы из «Всемирной паутины» и ранжирует их по релевантности пользовательскому запросу. Данное приложение прибегает к распознаванию и анализу веб-таблиц [125, 306, 309]. Например, известное исследование WebTables [105] нашло применение в поисковых сервисах компании Google [88]. Другим примером является поиск бизнес-информации в РК [137], а также сбор и индексация метаданных в электронных библиотеках [277, 278].

Пользовательское программирование, применяемое в табличных процессорах, тесно связано с задачами распознавания и анализа документных таблиц. В частности, они востребованы при автоматизации обнаружения ошибок

в РК [92, 156, 169, 285, 324], рефакторинга формул [84, 208, 246, 254], очистки и нормализации данных [91, 141–143, 191, 219, 229, 232, 367], а также автозаполнении [108, 255, 420]. Например, R. Abraham и M. Erwig [73, 74] полагаются на классификацию ячеек и обнаружение заголовка для автоматической коррекции ошибочных единиц измерений; X. Chen и др. [126] синтезируют пропущенные формулы на основе анализа агрегаций (итогов и подытогов).

Интеграция данных может опираться на решения распознавания, анализа и интерпретации таблиц [118, 119, 165, 166, 301, 376]. Они приобретают особенно важное значение в масштабах «Всемирной паутины» [107], предоставляющей доступ к большому массиву открытых табличных данных в полуструктурированном виде. Подобные решения [118, 119, 165, 166, 301] могут помочь повысить уровень открытости в терминах *5-звездочной схемы развертывания открытых данных*[43] [136]. Открытые табличные данные могут подвергаться дальнейшей интеграции [331].

Наполнение графа знаний может быть основано на семантической интерпретации таблиц. Будучи сопоставленной на уровне схемы, таблица может использоваться для добавления новых сущностей целевого графа знаний или расширения имеющихся дополнительными атрибутами [212,291,340,421]. Данное приложение родственно *генерации онтологий* [83, 180, 322, 351, 381, 384] и *генерации связанных данных* [195,257,262,405]. Набор однотипных сущностей таблицы может быть представлен в виде онтологии или связанных данных (RDF-триплетов с URI[44]-ссылками на внешние словари).

Ряд приложений требуют совместного применения решений распознавания, анализа и интерпретации документных таблиц, в частности, *вопрос-ответные системы* [226, 373], *генерация естественно-языковых текстов* [124, 315], *интеллектуальный анализ научной литературы* [193, 256, 407].

[43] `https://5stardata.info`

[44] Унифицированный идентификатор ресурса (англ. Uniform Resource Identifier).

1.6. Актуальные направления развития

Результатом литературного обзора стало выявление ряда ограничений современных исследований АПТ и перспективных путей их преодоления. В частности, современные решения извлечения таблиц из НПОД пренебрегают возможностями улучшения качества результатов за счет использования особенностей PDL-представления. Общим местом современных исследований по анализу и интерпретации таблиц является то, что они полагаются на заданность компоновки, атомарность ячеек и плоскую структуру заголовков, игнорируя большое количество случаев, когда эти предположения не выполняются. Таким образом, можно указать два актуальных направления развития: первое — изучение возможности применения PDL-специфичной информации для распознавания таблиц в НПОД; второе — поддержка произвольности структуры, включая свободную компоновку, структурированность ячеек и иерархичность заголовков в процессе анализа и интерпретации таблиц. Рассмотрим их подробнее.

1.6.1. Применения PDL-специфичной информации

Сегодня основное направление исследований связано с растровыми изображениями. Действительно, НПОД можно растеризовать, сведя таким образом задачу к более общей. Однако данный процесс неизбежно сопровождается потерей информации. По сравнению с растром, PDL-форматы (PDF, PostScript и др.) предоставляют дополнительную информацию (текст, шрифты, линейки и пр.), которая может улучшить качество результатов в некоторых случаях. В частности, она может использоваться на этапе предобработки для сегментации страницы НПОД (обнаружения заголовков, колонок и абзацев текста), а также на этапе постобработки для фильтрации кандидатных случаев.

Предобработка состоит в предварительном анализе компоновки страниц документа, включающем распознавание колонок текста, колонтитулов, заглавий, рисунков и пр. [316, 357], а также самих таблиц [96, 408, 431]. Из НПОД считываются PDL-инструкции отрисовки текста и графики. Во многих случаях порядок их проигрывания в графическом контексте соответствует структуре слоев документа, а также порядку чтения текста внутри каждого слоя. Данная информация может использоваться для того, чтобы распознать блоки текста, соответствующие отдельным словам, строкам и абзацам, а также ячейкам таблицы [4, 36, 41]. Несмотря на то что информация о компоновке документа может улучшить качество результатов распознавания таблиц, вопросы предобработки именно НПОД остаются малоизученными.

Постобработка позволяет выбрать релевантные случаи среди кандидатных таблиц, а также уточнить их границы и структуру [104]. Границы кандидатных таблиц могут быть некорректными. Например, ограничивающие рамки, предсказанные ИНС, могут пересекаться, захватывать по несколько действительных таблиц одновременно или, напротив, только их части. Постобработка может выполнять следующие действия: объединение соседних кандидатных таблиц [358, 393]; уточнение границ и структуры таблицы за счет добавления и удаления строк или столбцов [146, 203]; уточнение ограничивающей рамки с помощью нейросетевых моделей [187, 360].

Для того чтобы исключить ложные случаи из дальнейшей обработки, используются методы фильтрации таблиц, основанные на правилах исключения списков, рисунков и пр. [28], оценке схожести столбцов [239], вероятностных [393] и нейросетевых моделях [187,360]. Некоторые методы [214,344] обеспечивают поиск лучших вариантов среди кандидатных случаев с пересечением ограничивающих рамок на основе динамического программирования [214] и алгоритмов на графах [344]. За исключением единственного примера [28],

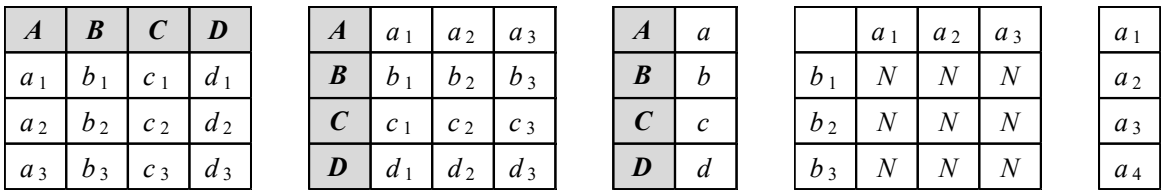
перечисленные решения постобработки рассчитаны на изображения документов. Случай PDF-источников в литературе почти не отражен [28].

1.6.2. Поддержка произвольности структуры

Известные решения анализа и интерпретации документных таблиц вебстраниц и РК ограничиваются типичной компоновкой. При этом многие приемы оформления остаются неохваченными. Важным свойством, которое игнорируется абсолютным большинством современных исследований, является структурированность самой ячейки, когда она содержит несколько значений одновременно. Практически все известные решения ориентируются на атомарность ячеек, полагая, что любая из них всегда соответствует единственному значению. Другим общим допущением является игнорирование иерархических заголовков, но именно такой прием часто применяется при компоновке многомерных таблиц. Очевидно, что такие решения не являются полными относительно всего разнообразия всевозможных способов оформления документных таблиц.

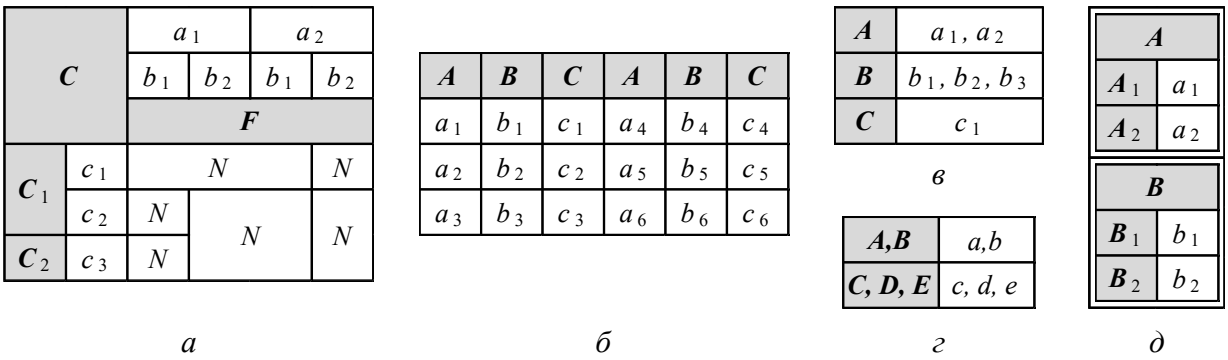
Свободная компоновка

Современные исследования в основном фокусируются на небольшом количестве заданных компоновочных типов таблиц; основные из них показаны на рис. 1.14. Каждый из них характеризуется некоторыми отличительными свойствами компоновки. Таксономии типов таблиц предлагаются в ряде работ [140,160,264]. Обычно они базируются на изучении архивных срезов «Всемирной паутины». С одной стороны, такое ограничение позволяет классифицировать таблицы и применять к ним методы, зависящие от их компоновочного типа. С другой стороны, очевидно, что не охватываются всевозможные случаи, с которыми можно столкнуться на практике, тем более что существу-



а *б* *в* *г* *д*

Рисунок 1.14 — Компоновочные типы таблиц таксономии E. Crestan и P. Pantel [140]: *вертикальный список* (*а*); *горизонтальный список* (*б*); *A/V-запись* (*в*); *матрица* (*г*); *перечисление* (*д*). Используется следующая нотация: $A, B, C, \ldots$ — имена атрибутов/категорий; $a_i, b_i, c_i, \ldots$ — значения атрибутов/категорий; N — значения характеризуемых данных (количеств)



а *б* *в* *г* *д*

Рисунок 1.15 — Варианты компоновки, представленные L. Lautert и др. [263]: *объединение ячеек* (*а*); *разрыв* (*б*); *список значений внутри ячейки* (*в, г*); *вложенность таблиц* (*д*). Ост. пояснения см. рис. 1.14

ющие решения имеют дело преимущественно только с наиболее распространенным материалом.

Ни одна из известных таксономий типов таблиц [140, 160, 263, 264] не является исчерпывающей. Многие свойства компоновки по-прежнему остаются без внимания. Часть таких свойств перечисляется в работах [263, 348]. Они соответственно показаны на рис. 1.15 и 1.16. Это перечисление можно дополнить следующими случаями: иерархической компоновкой заголовка [117], структурированными ячейками вида ключ–значение [406] и вычислимыми значениями данных [247–249, 285], как показано на рис. 1.17. Более того, различаются именно компоновочные свойства, но не функциональные. Хотя, на-

пример, сводная таблица может иметь, как одностороннюю, так и многостороннюю компоновку.

Структурированность ячеек

Текущие постановки задач анализа и интерпретации таблиц ограничиваются так называемыми *атомарными ячейками* [348]. Предполагается, что любая ячейка может содержать одно-единственное атомарное значение. Однако в действительности одна ячейка может включать несколько атомарных значений, выполняющих общую или разные функциональные роли (рис. 1.18). Примером служат единицы измерений (*$*, *%*, *℃* и др.), которые часто сопровождают количества. Такие ячейки принято называть *структурированными* [348].

J. C. Roldán [348] указывает, что существует лишь пара методов, учитывающих наличие структурированных ячеек [135, 406]. Первый [406] допускает, что пара ключ–значение может быть помещена в одну и ту же ячейку. Для того чтобы извлечь такие пары, используются символьные разделители (двоеточие, знак равенства и др.) [241]. Второй [135] поддерживает ячейки, содержащие вложенные таблицы или списки. Подход, предлагаемый в работе [135], состоит в том, чтобы предварительно избавиться от структурированных ячеек, разделив их на атомарные. Очевидно, что эти методы являются частными; в действительности, структура наполнения ячеек не обязательно укладывается в приведенные случаи.

Иерархичность заголовков

Многомерные таблицы часто содержат иерархии заголовков [117, 118]. Особый интерес представляет боковик, где используются различные приемы компоновки, отражающие иерархичность заголовков (рис. 1.19). В подобных

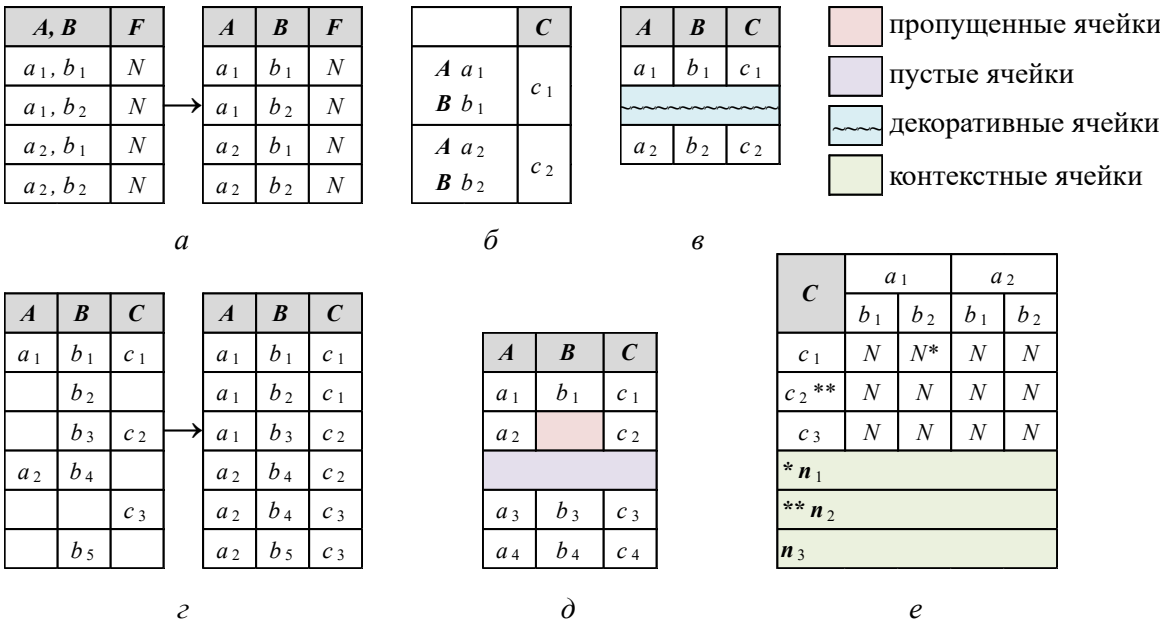


Рисунок 1.16 — Варианты компоновки, представленные J. Roldán [348]: *соединенные ячейки* (*а*); *структурированные ячейки* (*б*); *декоративные ячейки* (*в*); *факторизованные ячейки* (*г*); *пропущенные ячейки* (*д*); *контекстные ячейки* (*е*). Ост. пояснения см. рис. 1.14

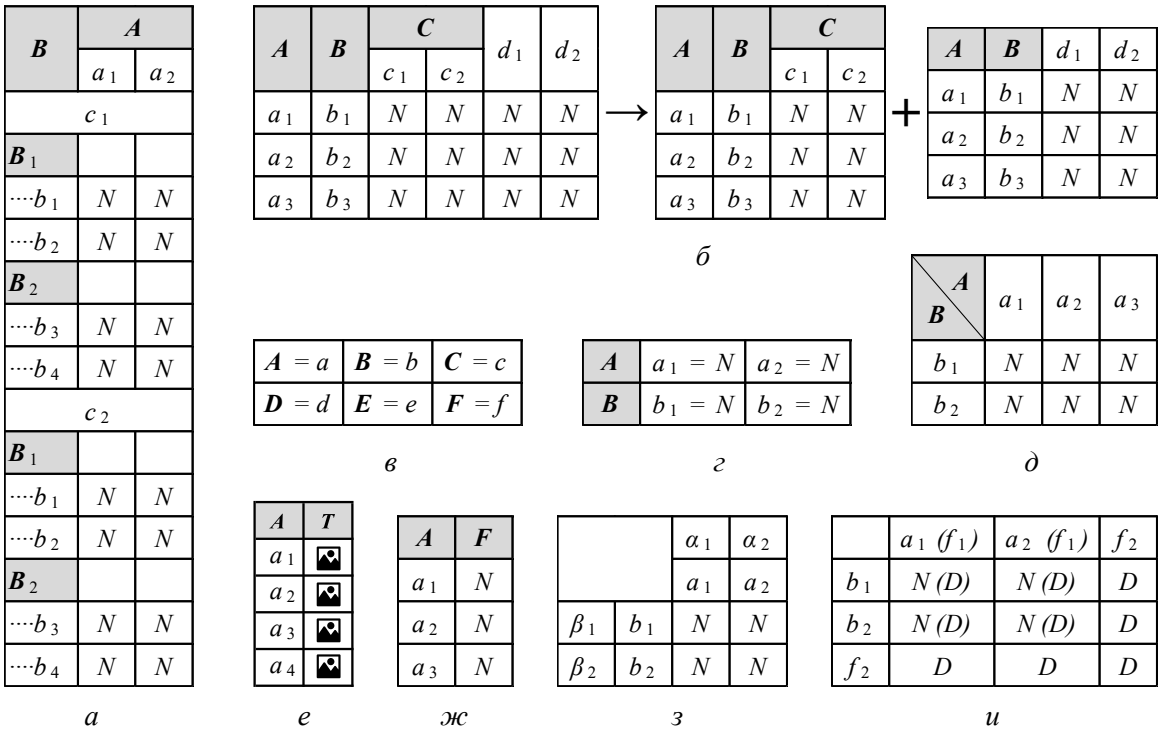
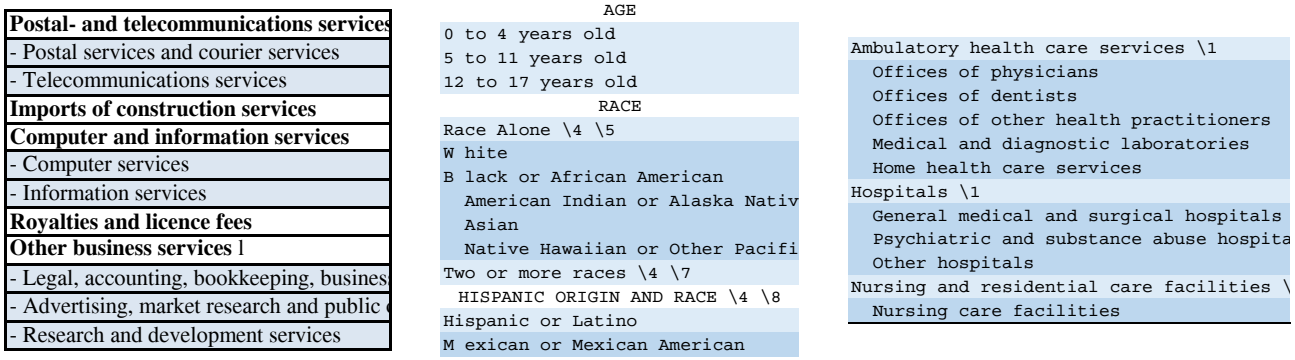


Рисунок 1.17 — Дополнительные варианты компоновки: иерархия заголовка (*а*); объединение многомерных таблиц (*б*); ключ–значение внутри ячейки (*в*, *г*); диагональные ячейки (*д*); изображение внутри ячейки» (*е*); частотное распределение (*ж*); многоязычная дупликация (*з*); вычислимые данные (обозначены как D) (*и*). Ост. пояснения см. рис. 1.14

STATE ABBREVIATION	2006			2007		
	Democratic (1,000)	Republican (1,000)	Percent for leading party	Democratic (1,000)	Republican (1,000)	Percent for leading party
AL	502	628	R-55.0	718	1 121	R-60.4
AK	94	133	D-56.6	143	159	R-50.1

Рисунок 1.18 — Структурированные ячейки на примере фрагмента статистической таблицы из коллекции SAUS: в первом случае одна ячейка (*Republican (1,000)*) содержит две метки (*Republican* и *1,000*), во втором случае (*R-55.0*) — метку (*R* — республиканская партия) и вхождение (*55.0*)



а *б* *в*

Рисунок 1.19 — Некоторые приемы компоновки иерархий заголовков боковика: символьное выделение (*а*); выравнивание по центру (*б*); отступ от левого края (*в*)

случаях интерпретировать табличные данные необходимо с учетом наличия иерархии. Тем не менее вопрос извлечения иерархий заголовков до сих пор не получил должного внимания со стороны исследовательского сообщества. В литературе представлены единичные работы, в которых предлагаются методы распознавания родительско-дочерних связей внутри иерархических заголовков [119, 120].

Z. Chen и M. Cafarella [119, 120] извлекают иерархию заголовка (боковика) на основе алгоритмов машинного обучения. Выполняется бинарная классификация всех пар ячеек, составляющих боковик, на два класса: родительско-дочерние связи и остальные. Предлагаемый ими классификатор обучается с помощью метода опорных векторов. В качестве признаков вы-

бираются компоновочные, стилевые и содержательные свойства отдельных ячеек и целых строк. Данное решение может быть дополнено автоматическим определением наличия иерархии заголовка [121]. Подробное исследование диссертационной работы Z. Chen [120] показало, что точно воспроизвести представленный метод невозможно в виду неполного изложения.

1.7. Выводы

Представленный обзор современного состояния исследований научной проблематики АПТ дополняет упомянутые ревизии тематической литературы [98, 104, 150, 328, 348, 421]. Здесь обобщаются известные постановки задач; согласовываются их терминологии. В результате уточнены формулировка и структура совокупности задач АПТ; выявлены пробелы в изучении некоторых вопросов, которым не уделялось должного внимания в прошлом; предложены направления перспективных исследований, нацеленных на преодоление текущих ограничений.

Исходя из представленного обзора, можно сделать следующие выводы. Главным образом предшествующие исследования были адресованы задачам распознавания и интерпретации таблиц. Наименее проработанными остаются вопросы их анализа. Подтверждается заключение, сделанное в предшествующих работах [104, 348]: ни одно из известных решений в настоящий момент не реализует АПТ в полной мере. В основном предлагаются только частные решения отдельных задач и подзадач, в частности, распознавания таблиц в изображениях научных статей, анализа веб-таблиц распространенных компоновочных типов, а также интерпретации таблиц однотипных сущностей, соответствующих третьей нормальной форме. В литературе представлено лишь небольшое количество решений полного цикла (охватывающих все эта-

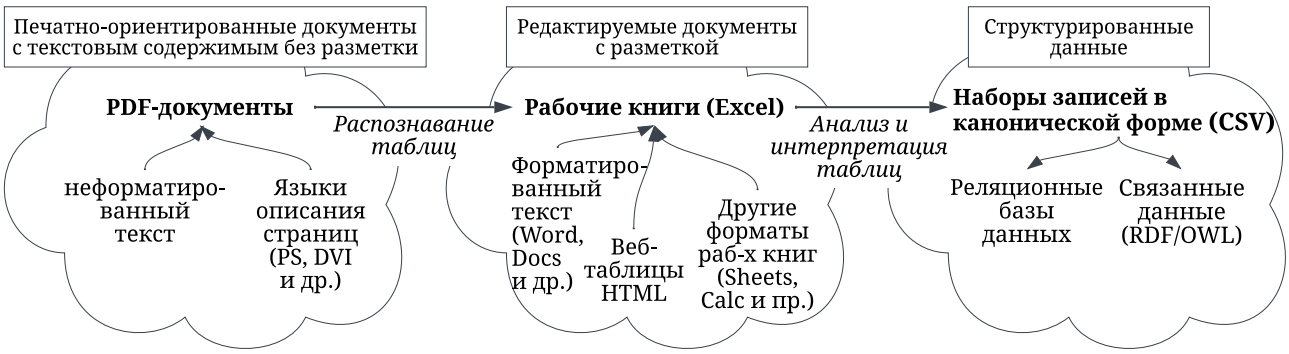


Рисунок 1.20 — Возможные цепочки преобразования данных в процессе АПТ

пы АПТ), а именно [295, 339, 350], но каждое из них подходит лишь для ограниченного количества случаев исходных данных.

В целом, известные решения имеют дело только с простыми приемами компоновки таблиц, тем самым значительно ограничивая диапазон обрабатываемых случаев. Известные решения предназначены для обработки таблиц предопределенных компоновочных типов, ограничиваясь обычно несколькими наиболее распространенными типами: от одного до пяти. Важным свойством, которое игнорируется абсолютным большинством текущих исследований, является структурированность самой ячейки, когда одна ячейка представляет несколько значений одновременно. Практически все известные решения ориентируются на атомарность ячеек. Другим общим допущением является игнорирование визуально выраженных иерархий заголовков, но именно такой прием часто применяется при компоновке многомерных таблиц. Очевидно, что такие решения не являются полными относительно всего разнообразия всевозможных компоновочных типов и свойств.

Вопросы АПТ активно исследуются на протяжение трех последних десятилетий. При этом интерес к данной научной проблеме продолжает только усиливаться. Это стало особенно заметно в последнее десятилетие с появлением новых техник, таких как глубокое обучение, связанные открытые данные, а также векторное представление слов и сущностей. Актуальны направ-

ления исследований, расширяющие возможности автоматизации процессов распознавания, анализа и интерпретации документных таблиц. В частности, имеется ряд нерешенных вызовов: улучшение качества результатов распознавания таблиц в НПОД; поддержка произвольности структуры, а именно свободной компоновки, структурированности ячеек и иерархичности заголовков при анализе и интерпретации таблиц. Настоящая диссертационная работа адресована перечисленным актуальным направлениям развития.

Предлагаемый в диссертации метод автоматизации распознавания таблиц применим к НПОД, приводимым к формату PDF. Следует отметить, что PDF является одним из наиболее общеупотребимых способов представления НПОД. Источники, представленные с помощью других языков описания страниц (PostScript, DVI[45] и пр.), а также форматов текстовых процессоров (Word, RTF и пр.), могут быть приведены к PDF с помощью имеющихся в свободном доступе виртуальных принтеров или конвертеров. Аналогично метод автоматизации анализа и интерпретации таблиц применим к источникам, приводимым к наиболее популярным форматам РК Excel/Sheets. Так, таблицы, представленные с помощью форматов: HTML, CSV, Word и пр., могут быть приведены к ним с помощью имеющихся в свободном доступе конвертеров. Таким образом, общность предлагаемого подхода обеспечивается имеющимися возможностями конвертирования документов между форматами. Возможные цепочки преобразования данных в процессе АПТ демонстрируются на рис. 1.20.

Результат, представленный в данной главе

Усовершенствованная структура совокупности задач АПТ [3], в которой согласована терминология, сложившаяся в разных направлениях родственных исследований: распознавании документов, анализе РК и семантической ин-

[45] Аппаратно независимый (англ. DeVice Independent) формат издательской системы TeX.

терпретации таблиц. По сравнению с имеющимися формулировками АПТ, предлагаемая структура является более полной за счет уточнения задачи интерпретации таблиц.

Публикации

Представленный результат опубликован в ряде рецензируемых изданий, в том числе в журнале [3], а также материалах всероссийских и международных мероприятий [21, 22, 33].

Глава 2

Автоматизация распознавания таблиц

Глава посвящена вопросам автоматизации распознавания таблиц НПОД. Вводятся определения, составляющие модель представления страницы (раздел 2.1). Рассматривается постановка задачи (раздел 2.2). Предлагается метод ее решения (раздел 2.3). Подробно излагаются реализующие его методики подготовки (раздел 2.4.1) и сегментации страниц (раздел 2.4), обнаружения (разделы 2.5 и 2.6) и сегментации таблиц (раздел 2.7). В конце главы делаются выводы (раздел 2.8).

2.1. Модель страницы документа

Модель страницы НПОД предназначена для представления PDL-специфичной информации, участвующей в процессе распознавания таблиц. Для определения модели введем следующие понятия.

Нотация

Будем использовать следующую нотацию: запись $o = (x_1, \ldots, x_n)$ означает, что некоторый объект o состоит из свойств $x_1, \ldots, x_n,\ n \in \mathbb{N}$. При этом отдельные значения свойства объекта o будем записывать следующим образом:

$$x_1, \ldots, x_n \mid x_1 = x_1(o), \ldots, x_n = x_n(o).$$

Обозначим через ε некоторую малую, а через θ — пороговую величину.

Система координат

Используется прямоугольная система вещественных координат, в которой значения по оси X возрастают слева направо, а по оси Y — сверху вниз.

Ограничивающая рамка

Пусть некоторый объект o ограничен рамкой, заданной координатами ее сторон: левой — x_l, верхней — y_t, правой — x_r и нижней — y_b, тогда определена функция

$$\text{bbox}(o) = (x_l, y_t, x_r, y_b) \mid x_l = x_l(o), y_t = y_t(o), x_r = x_r(o), y_b = y_b(o).$$

Проекция

Под проекцией некоторой ограничивающей рамки $\text{bbox}(o)$ на ось X будем понимать упорядоченную пару x-координат его левой и правой сторон:

$$p_x(o) = (x_l, x_r) \mid x_l = x_l(o), x_r = x_r(o),$$

а под проекцией на ось Y — упорядоченную пару y-координат его верхней и нижней сторон:

$$p_y(o) = (y_t, y_b) \mid y_t = y_t(o), y_b = y_b(o).$$

Символьная позиция

Пусть s — некоторый символ, напечатанный на странице документа, тогда ему соответствует символьная позиция (рис. 2.1), определяемая следующим образом:

$$s = (\text{code}, \text{font}, \text{bbox}, \text{idx}),$$

где code — код символа в некоторой кодировке; font — шрифт, характеризуемый семейством (Arial, Times и пр.), размером и начертанием (обычным, полужирным, курсивом и пр.); bbox — ограничивающая рамка; idx — индекс (номер) PDL-инструкции отрисовки по порядку ее выполнения в графическом контексте.

Текстовый компонент

Под текстовым компонентом понимается некоторое напечатанное «слово»

Рисунок 2.1 — Символьная позиция ($а$), текстовый компонент ($б$) и блок ($в$)

или его часть (рис. 2.1). Определим его следующим образом:

$$w = (S, \text{font}, \text{bbox}),$$

где $S = \{s_1, \ldots, s_n\}$, $n \in \mathbb{N}$ — упорядоченная последовательность соседних непробельных символьных позиций, таких что:

$$\text{font}(w) = \text{font}(s_1) = \cdots = \text{font}(s_n),$$

$$x_r(s_1) = x_l(s_2) - \varepsilon_1, \ldots, x_r(s_{n-1}) = x_l(s_n) - \varepsilon_{n-1}$$

$$\text{bbox}(w) = (x_l, y_t, x_r, y_b) \mid$$

$$x_l = x_l(s_1), y_t = \min_{0 \leqslant i \leqslant n} y_t(s_i), x_r = x_r(s_n), y_b = \max_{0 \leqslant i \leqslant n} y_b(s_i).$$

Следует отметить, что высоты ограничивающих рамок любых символьных позиций, отрисованных с помощью одного шрифта (наименования, начертания и размера) совпадают так, что для текстового компонента w выполняется следующее условие:

$$p_y(w) = p_y(s_1) = \cdots = p_y(s_n).$$

Для каждого текстового компонента можно вычислить начальный и конечный индексы в порядке отрисовки принадлежащих ему символьных позиций соответственно:

$$\text{idx}_s(w) = \min_{0 \leqslant i \leqslant n} \text{idx}(s_i) \text{ и } \text{idx}_e(w) = \max_{0 \leqslant i \leqslant n} \text{idx}(s_i).$$

Блок текста

Блок текста обычно соответствует напечатанной строке или абзацу. Фактически, один или несколько текстовых компонентов могут быть сгруппированы в блок (рис. 2.1) следующим образом:

$$b = (W, \mathrm{bbox}),$$

где $W = \{w_1, \ldots, w_n\}$, $n \in \mathbb{N}$ — упорядоченная последовательность текстовых компонентов, таких что их x_l-координаты возрастают слева направо:

$$x_l(w_i) < x_l(w_{i+1}),$$

а при равенстве последних их y_t-координаты возрастают сверху вниз:

$$x_l(w_i) = x_l(w_{i+1}), y_t(w_i) < y_t(w_{i+1});$$

ограничивающая рамка блока заключает все его текстовые компоненты:

$$\mathrm{bbox}(b) = (x_l, y_t, x_r, y_b) \mid$$
$$x_l = \min_{0 \leqslant i \leqslant n} x_l(w_i), y_t = \min_{0 \leqslant i \leqslant n} y_t(w_i), x_r = \max_{0 \leqslant i \leqslant n} x_r(w_i), y_b = \max_{0 \leqslant i \leqslant n} y_b(w_i).$$

Для каждого блока можно вычислить начальный и конечный индексы в порядке отрисовки символьных позиций, принадлежащих его текстовым компонентам, соответственно:

$$\mathrm{idx}_s(b) = \min_{0 \leqslant i \leqslant n} \mathrm{idx}_s(w_i) \text{ и } \mathrm{idx}_e(b) = \max_{0 \leqslant i \leqslant n} \mathrm{idx}_e(w_i).$$

Линейка разграфки

Представляет прямую линию с некоторой толщиной. Подобно ограничивающей рамке, она задается двумя парами начальных и конечных координат следующим образом:

$$u_r = (x_l, y_t, x_r, y_b) \mid x_l = x_l(u_r), y_t = y_t(u_r), x_r = x_r(u_r), y_b = y_b(u_r).$$

След перемещения пера

Представляет невидимый след перемещения пера (предполагается, что перо, которым выполняется рисование в документе, может перемещаться по странице, ничего не рисуя). Подобно ограничивающей рамке, он задается двумя парами начальных и конечных координат следующим образом:

$$u_p = (x_l, y_t, x_r, y_b) \mid x_l = x_l(u_p), y_t = y_t(u_p), x_r = x_r(u_p), y_b = y_b(u_p).$$

Страница документа

Представляет входные данные процесса распознавания таблиц. Она определяется тройкой следующего вида:

$$p = (S, U_r, U_p),$$

где S — множество символьных позиций, U_r — множество линеек разграфки, U_p — множество следов перемещения пера.

Кандидатная таблица

Представляет выходные данные процесса распознавания таблицы. Она задается парой следующего вида:

$$t = (\mathrm{bbox}, \phi),$$

где bbox — ограничивающая рамка, задающая местоположение таблицы внутри страницы документа, а ϕ — ее физическая структура. Последняя определяется следующим образом. Пусть B — множество блоков текста, расположенных внутри ограничивающей рамки $\mathrm{bbox}(t)$. Пусть таблица t состоит из n строк и m столбцов, тогда любая ее ячейка характеризуется четырьмя координатами:

$$c_l, r_t, c_r, r_b \mid c_l \geqslant 1,\ r_t \geqslant 1,\ c_r \geqslant c_l,\ r_b \geqslant r_t,\ c_r \leqslant m,\ r_b \leqslant n,$$

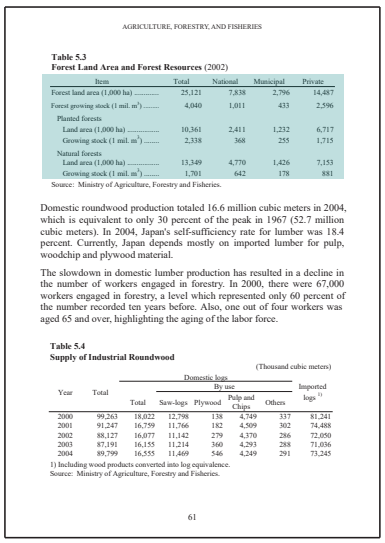
где c_l — левый столбец, r_t — верхняя строка, c_r — правый столбец и r_b — нижняя строка. Пусть C — множество ячеек в пространстве строк и столбцов таблицы t. Тогда будем говорить, что физическая структура $\phi(t)$ отображает блоки текста из B в ячейки из C. Следует отметить, что одна ячейка может включать несколько блоков текста. При этом любой блок может принадлежать только одной ячейке.

2.2. Постановка задачи распознавания таблиц

Примем, что любая исходная таблица представлена в НПОД исключительно PDL-инструкциями отрисовки текста и графики (т. е. неизвестно ни ее местоположение на странице, ни ее физическая структура), необходимо распознать ее по доступным данным, сделав ее машиночитаемой. Рассматриваемая задача может быть сформулирована следующим образом:

- Имеется страница НПОД p с машиночитаемым набором символьных позиций S, часть из которых может визуально представлять одну или несколько таблиц: $t_1, \ldots, t_n$.
- Необходимо распознать их физические структуры: $\phi(t_1), \ldots, \phi(t_n)$.

Источник должен удовлетворять следующим ограничениям: страница имеет одноколонную манхэттенскую компоновку, любая непустая ячейка может содержать только текст; страница содержит таблицу целиком (в случае, когда одна «логическая» таблица размещена на нескольких соседних страницах, каждая ее часть рассматривается как отдельная «физическая» таблица, которую требуется распознать). Восстановленная физическая структура должна предоставлять координаты ячеек в пространстве строк и столбцов, а также их текстовое содержимое. В результате она может быть представлена в редактируемом формате данных HTML/Excel (рис. 2.2).



AGRICULTURE, FORESTRY, AND FISHERIES

Table 5.3
Forest Land Area and Forest Resources (2002)

Item	Total	National	Municipal	Private
Forest land area (1,000 ha)	25,121	7,838	2,796	14,487
Forest growing stock (1 mil. m^3)	4,040	1,011	433	2,596
Planted forests				
Land area (1,000 ha)	10,361	2,411	1,232	6,717
Growing stock (1 mil. m^3)	2,338	368	255	1,715
Natural forests				
Land area (1,000 ha)	13,349	4,770	1,426	7,153
Growing stock (1 mil. m^3)	1,701	642	178	881

Source: Ministry of Agriculture, Forestry and Fisheries.

Domestic roundwood production totaled 16.6 million cubic meters in 2004, which is equivalent to only 30 percent of the peak in 1967 (52.7 million cubic meters). In 2004, Japan's self-sufficiency rate for lumber was 18.4 percent. Currently, Japan depends mostly on imported lumber for pulp, woodchip and plywood material.

The slowdown in domestic lumber production has resulted in a decline in the number of workers engaged in forestry. In 2000, there were 67,000 workers engaged in forestry, a level which represented only 60 percent of the number recorded ten years before. Also, one out of four workers was aged 65 and over, highlighting the aging of the labor force.

Table 5.4
Supply of Industrial Roundwood

(Thousand cubic meters)

Year	Total	Domestic logs					Imported logs 1)
		Total	By use				
			Saw-logs	Plywood	Pulp and Chips	Others	
2000	99,263	18,022	12,798	138	4,749	337	81,241
2001	91,247	16,759	11,766	182	4,509	302	74,488
2002	88,127	16,077	11,142	279	4,370	286	72,050
2003	87,191	16,155	11,214	360	4,293	288	71,036
2004	89,799	16,555	11,469	546	4,249	291	73,245

1) Including wood products converted into log equivalence.
Source: Ministry of Agriculture, Forestry and Fisheries.

61

```
<table>
    <tr>
        <td>Item</td>
        <td>Total</td>
        <td>National</td>
        ...
    </tr>
    <tr>
        <td>Forest land area (1,000 ha) ...</td>
        <td>25,121</td>
        <td>7,838</td>
        ...
    </tr>
    ...
</table>
```

а *б*

Рисунок 2.2 — Пример распознавания таблицы: исходное ($а$) и целевое представление ($б$)

2.3. Метод автоматизации распознавания таблиц

Для решения поставленной задачи предлагается метод на основе применения PDL-специфичной информации. За основу берется нисходящий подход, т. е. последовательное выполнение сперва обнаружения таблицы внутри источника, а затем ее ячеек. В рассматриваемом случае он может быть сформулирован следующим образом:

- Найти непересекающиеся наборы символьных позиций $S_1 \subset S, \ldots, S_n \subset S$, принадлежащие размещенным на странице p таблицам $t_1, \ldots, t_n$ соответственно.
- Найти для каждой таблицы t_i непересекающиеся наборы символьных позиций $S_{i1} \subset S_i, \ldots, S_{im} \subset S_i$, принадлежащие ее ячейкам $c_1, \ldots, c_m$ соответственно.
- Пусть каждая символьная позиция $s : s \in S_i$ сопоставлена с некоторой ячейкой $c : c \in C_i$, тогда восстановлены физические структуры таблиц: $\phi(t_1), \ldots, \phi(t_n)$.

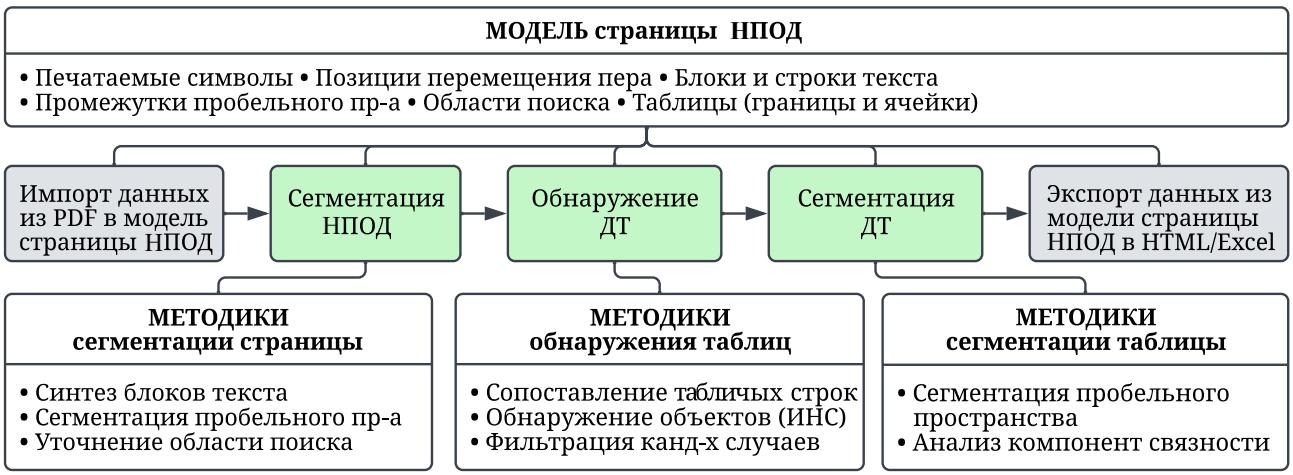
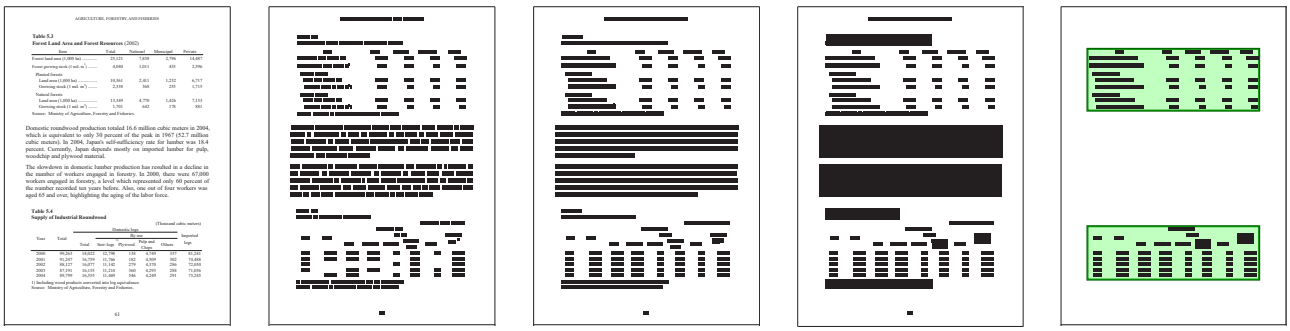


Рисунок 2.3 — Состав и этапы метода автоматизации распознавания таблиц НПОД

Сперва выполняется обнаружение ограничивающей рамки таблицы внутри источника (обеспечивается выбор набора символьных позиций S_i), а затем сегментация выделенной части на отдельные ячейки (обеспечивается отображение: $S_i \to C_i$). Первая часть включает предварительную сегментацию страницы документа на основе синтеза блоков текста и последующее предсказание ограничивающих рамок таблиц посредством либо попарного сопоставления кандидатных табличных строк, либо применения искусственных нейронных сетей (далее ИНС) обнаружения объектов на изображениях. Вторая часть выполняется за счет либо сегментации пробельного (т. е. не занятого текстом) пространства, либо анализа компонент связности блоков текста. Состав и этапы предлагаемого метода схематично показаны на рис. 2.3. Он может быть конкретизирован разными комбинациями методик сегментации страницы документа, обнаружения и сегментации таблицы. Далее излагаются соответствующие методики.

2.4. Сегментация страницы документа

Подзадача сегментации страницы НПОД состоит в том, чтобы наиболее правдоподобно восстановить отдельные части текста, соответствующие



а *б* *в* *г* *д*

Рисунок 2.4 — Синтез блоков текста: напечатанные символы (*а*); слова (*б*); строки (*в*); абзацы (*г*). Обнаружение ограничивающих рамок таблиц (*д*)

отдельным словам, строкам и абзацам, а также промежутки пробельного пространства, отделяющие друг от друга соседние колонки текста и таблиц. Предлагается методика решения данной подзадачи на основе встроенных правил. Ее особенностью является использование PDL-специфичной информации. Сегментация страницы документа выполняется снизу вверх: доступные изначально напечатанные символы объединяются в текстовые компоненты — «слова» (рис. 2.4, *а, б*), те в свою очередь группируются в многокомпонентные блоки текста — либо однострочные, соответствующие условно «строкам» (рис. 2.4, *в*), либо многострочные, соответствующие условно «абзацам» (рис. 2.4, *г*). По восстановленным блокам сегментируется пробельное пространство. Рассмотрим предлагаемую методику подробнее в следующем порядке: подготовка страницы документа (раздел 2.4.1), группирование символьных позиций в текстовые компоненты (раздел 2.4.2), группирование текстовых компонентов в однострочные (раздел 2.4.3) и многострочные блоки (раздел 2.4.4) и сегментацию пробельного пространства (раздел 2.4.5). Следует отметить, что оба варианта синтеза блоков текста, а именно в виде «строк» (раздел 2.4.3) и «абзацев» (раздел 2.4.4), могут использоваться для последующего распознавания таблиц.

2.4.1. Подготовка страницы документа

Подготовка страницы документа состоит в формировании символьных позиций, линеек разграфки и следов перемещения пера из доступных данных источника. Рассмотрим подробнее представление и чтение исходных данных НПОД (PDF).

Представление исходных данных в источнике

Будучи ориентированным на визуальное воспроизведение, НПОД часто не предоставляет никакой информации о своей структуре: заголовках, абзацах, таблицах и пр. Так, PDF-файл содержит поток инструкций на языке описания страниц, выполняющих отрисовку текста, примитивов векторной графики и растра. Процессор вывода PDF — стековый: сначала параметры PDL-инструкции помещаются на стек; затем когда она вызывается, ее параметры оттуда снимаются. Для определения параметров вывода используется графический контекст. Он содержит настройки шрифта, пера, заливки, матрицы преобразования и пр. Например, следующий фрагмент PDF-файла (табл. 2.1) приведет к выводу заданного текста в указанных координатах. PDL-инструкция `Tf` указывает, что будет применен шрифт, именованный как `F15` в объекте ресурсов, где определены все детали шрифта (название, ширины глифов, кодировка). PDL-инструкция `Tj` содержит список кодов (числовых или символьных значений), идентифицирующих глифы.

Следует отметить, что два визуально идентичных НПОД могут быть представлены содержательно разными PDF-файлами. Различные виртуальные принтеры или конвертеры генерируют разные наборы PDL-инструкций из одного и того же источника. Так, текст строки может быть получен в результате выполнения разных последовательностей PDL-инструкций, как показано на рис. 2.5, *а*–*г*. PDL-инструкция может соответствовать одному сим-

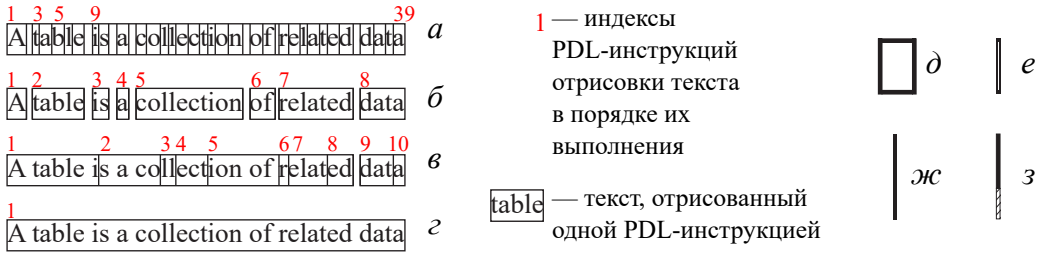


Рисунок 2.5 — Представление текста и графики в PDF-источниках: каждый символ отрисован отдельной PDL-инструкцией (*а*); каждая последовательность непробельных символов строки отрисована отдельной PDL-инструкцией (*б*); строка разделена на части, каждая из которых отрисована отдельной PDL-инструкцией (*в*); строка отрисована одной PDL-инструкцией (*г*); прямоугольник, отрисованный одной PDL-инструкцией, соответствует четырем (*д*) и одной линейке (*е*); линейка состоит из одного (*ж*) и двух сегментов (*з*), отрисованных отдельными PDL-инструкциями

волу (рис. 2.5, *а*), последовательности непробельных символов (рис. 2.5, *б*), подстроке (рис. 2.5, *в*) или целой строке (рис. 2.5, *г*). При этом пробелы могут быть сформированы различными способами, например: явно как пробельный символ, т. е. не имеющий видимого глифа (рис. 2.5, *а*); неявно как промежуток между соседних непробельных символов, т. е. имеющих видимые глифы (рис. 2.5, *б*).

Линейки, в том числе составляющие разграфку таблиц, могут быть представлены PDL-инструкциями отрисовки линий или прямоугольников, как по-

Таблица 2.1 — Пример PDL-инструкций для отрисовки текста в НПОД PDF

PDL-инструкция	**Действие**
`BT`	Начало текста
`/F15 10 Tf`	Выбрать шрифт F15 и установить размер шрифта 10
`150 300 Td`	Переместить x, y-координаты отрисовки относительно текущих
`(A table is) Tj`	Отрисовать текст «A table is»
`ET`	Конец текста

казано на рис. 2.5, *д*–*з*. При этом в одном случае прямоугольник может представлять четыре отдельных линейки, каждая из которых соответствует одной из его границ (рис. 2.5, *д*). В другом узкий и длинный прямоугольник образует только одну линейку с толщиной, равной его короткой стороне (рис. 2.5, *е*). Одна линейка может быть отрисована одной (рис. 2.5, *ж*) или несколькими линиями (рис. 2.5, *з*). PDL-инструкции отрисовки линий и прямоугольников могут также использоваться для подчеркивания, перечеркивания или маркерного выделения текста. Некоторые PDF-генераторы дополнительно записывают в PDF-файлы PDL-инструкции перемещения пера, которые могут соответствовать границам ячеек таблиц.

Чтение исходных данных из источника

Интерпретация PDL-инструкций отрисовки текста и графики зависит от графического контекста, который устанавливает различные параметры, задающие текущий шрифт, контур, заливку, преобразования в выбранной системе координат и пр. Считывание текста и линеек из PDF-файла требует его интерпретации, аналогичной проигрыванию НПОД на графическом устройстве. При этом требуется дополнительная обработка, в том числе отдельные сегменты визуально единой линейки должны объединяться, линии подчеркивания и перечеркивания текста должны трансформироваться в стилевые характеристики текста.

Для того чтобы считать исходные данные, предлагается использовать инструментальные средства обработки PDF-документов — PDFBox. В процессе проигрывания исходного НПОД (PDF) они позволяют выполнить PDL-инструкции, чтобы сформировать символьные позиции, линейки и следы перемещения пера.

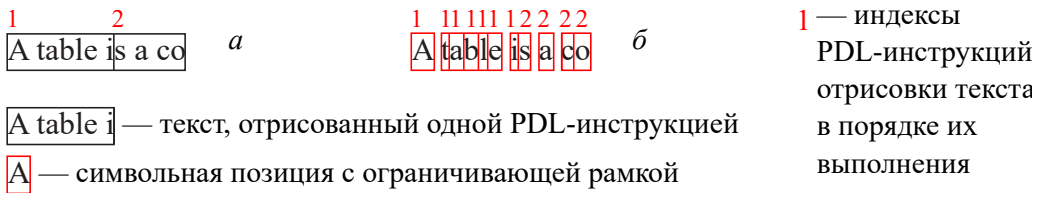


Рисунок 2.6 — Чтение текста из PDL-источника: исходное представление (*а*); полученные символьные позиции (*б*)

В результате выполнения PDL-инструкций отрисовки текста извлекаются символьные позиции, соответствующие только видимым непробельным символам; индекс каждой из них считается по порядку выполнения исходной PDL-инструкции (рис. 2.6). При этом можно отфильтровать те из них, которые относятся к псевдографике, т. е. фактически служат не для представления текста, но для разграфки. Следует отметить, что подобная псевдографика часто используется в таблицах, сгенерированных в виде ASCII-текста (рис. 2.7, *а*).

Исключение псевдографики из текста выполняется следующим образом. Рассматриваются только те символьные позиции, коды которых удовлетворяют известным диапазонам символов псевдографики. Если ограничивающие рамки нескольких символьных позиций примыкают друг к другу верхними и нижними сторонами, то из них формируется вертикальная линейка разграфки. Аналогично, если они примыкают друг к другу боковыми сторонами, то из них формируется горизонтальная линейка разграфки. Таким образом, восстановленные линейки добавляются в множество видимых линий обрабатываемой страницы документа. При этом сами символьные позиции псевдографики исключаются из текста и в дальнейшем не используются (рис. 2.7, *б*, *в*).

Линейки формируются в результате выполнения PDL-инструкций отрисовки видимых линий и прямоугольников. Каждая отрисованная линия преобразуется в линейку. Каждый отрисованный прямоугольник, ширина или высота которого не превышает заданного порога (т. е. визуально он представ-

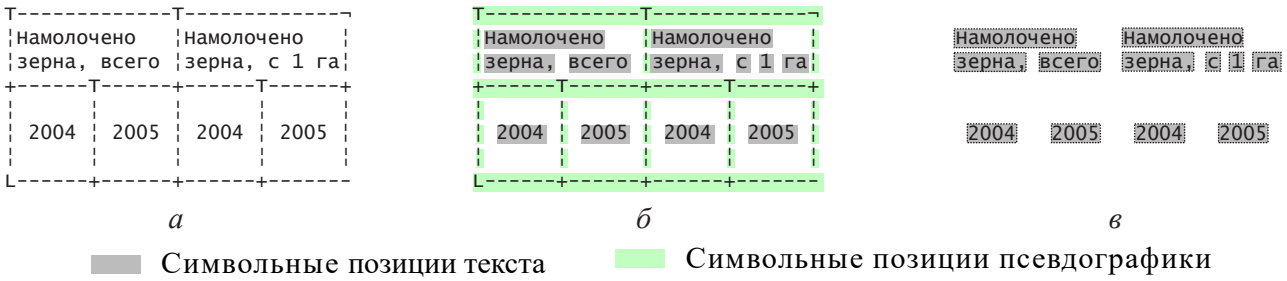


Рисунок 2.7 — Текст с псевдографикой (*а*); символьные позиции до (*б*) и после (*в*) исключения псевдографики

ляет линию), становится линейкой, соответствующей его наиболее длинной стороне. В противном случае, он заменяется четырьмя линейками, соответствующими его сторонам. Среди полученных линеек выбираются только те, которые являются ортогональными по отношению к сторонам страницы. Линейки, расположенные на одной прямой или двух параллельных попарно объединяются в одну в том случае, когда либо они пересекаются, либо расстояние между ними не превышает заданного порога (т. е. визуально они примыкают друг к другу). Следы перемещения пера формируются аналогичным образом в результате выполнения PDL-инструкций отрисовки невидимых линий и прямоугольников.

2.4.2. Группирование символьных позиций в текстовые компоненты

Рассматриваются только непробельные символьные позиции. Предполагается, что они не пересекаются между собой. Они группируются в текстовые компоненты, условно соответствующие отдельным словам напечатанного текста (рис. 2.8). Предполагается, что две соседние символьные позиции s_1 (левая) и s_2 (правая) принадлежат одному текстовому компоненту w в том

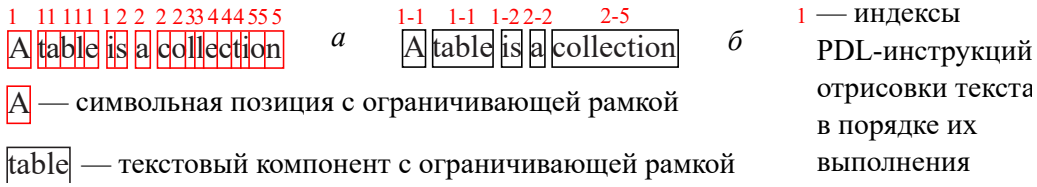


Рисунок 2.8 — Группирование символьных позиций ($а$) в текстовые компоненты ($б$); для каждого текстового компонента указаны начальный и конечный индексы PDL-инструкций, например: «1-2» для «is»

случае, когда выполняются следующие условия:

$$0 \leqslant x_l(s_2) - x_r(s_1) \leqslant \varepsilon_1, \tag{2.1}$$

$$0 \leqslant |y_t(s_1) - y_t(s_2)| \leqslant \varepsilon_2 \text{ и } 0 \leqslant |y_b(s_1) - y_b(s_2)| \leqslant \varepsilon_3, \tag{2.2}$$

$$\text{font}(s_1) = \text{font}(s_2), \tag{2.3}$$

$$\text{idx}(s_1) = \text{idx}(s_2) \text{ или } \text{idx}(s_1) = \text{idx}(s_2) - 1, \tag{2.4}$$

т. е. между боковых сторон их ограничивающих рамок нет пробельных промежутков — (2.1), пересечение их проекций на ось Y составляет существенную часть высоты текстового компонента — (2.2), шрифты (наименование, начертание и размер) полностью совпадают — (2.3), а индексы их отрисовки либо равны, либо индекс левой из них на единицу меньше правой — (2.4). Символьные позиции попарно сопоставляются, составляя текстовые компоненты обрабатываемой страницы.

2.4.3. Группирование текстовых компонентов в однострочные блоки

Обнаружение «строк» сводится к объединению текстовых компонентов в однострочные блоки. Пусть w — текстовый компонент, тогда определим две рамки, которые служат для захвата верхнего — (2.5) и нижнего — (2.6)

углов соседнего текстового компонента (рис. 2.9), следующим образом:

$$\mathrm{bbox}_t(w) = \big(x_r(w),\ y_t(w) - h \cdot k_h,\ x_r(w) + i \cdot k_i,\ y_t(w) + h \cdot k_h\big), \tag{2.5}$$

$$\mathrm{bbox}_b(w) = \big(x_r(w),\ y_b(w) - h \cdot k_h,\ x_r(w) + i \cdot k_i,\ y_b(w) + h \cdot k_h\big), \tag{2.6}$$

где h: $h = y_b(w) - y_t(w)$ — высота, а k_h: $k_h \in \mathbb{R}$, $0 \leqslant k_h \leqslant 1$ — коэффициент смещения; i — межсимвольный интервал пробела, соответствующий шрифту текстового компонента, а $k_i = 1$, если шрифт моноширинный, и $k_i = 2$, если шрифт пропорциональный.

Пусть w_1 и w_2 — два текстовых компонента, таких что $x_r(w_1) < x_l(w_2)$, тогда определим пробельный промежуток g между ними следующим образом:

$$g = \big(x_r(w_1),\ \min\{y_t(w_1), y_t(w_2)\},\ x_l(w_2),\ \max\{y_b(w_1), y_b(w_2)\}\big).$$

Кроме того, определим, что они принадлежат одному блоку, если выполняются следующие условия:

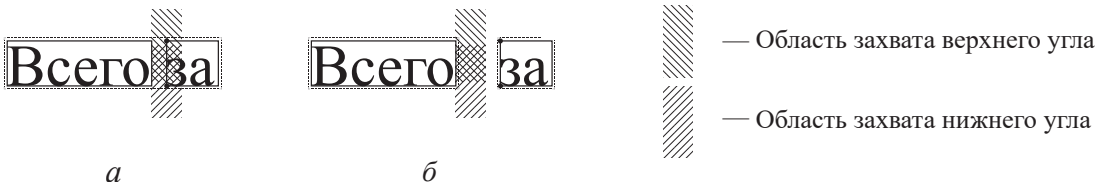
$$\mathrm{bbox}_t(w_1) \cap \mathrm{bbox}(w_2) \neq \varnothing \text{ и } \mathrm{bbox}_b(w_1) \cap \mathrm{bbox}(w_2) \neq \varnothing, \tag{2.7}$$

$$u_r \cap g = \varnothing,\ \forall\ u_r : u_r \in U_r, \tag{2.8}$$

$$u_p \cap g = \varnothing,\ \forall\ u_p : u_p \in U_p, \tag{2.9}$$

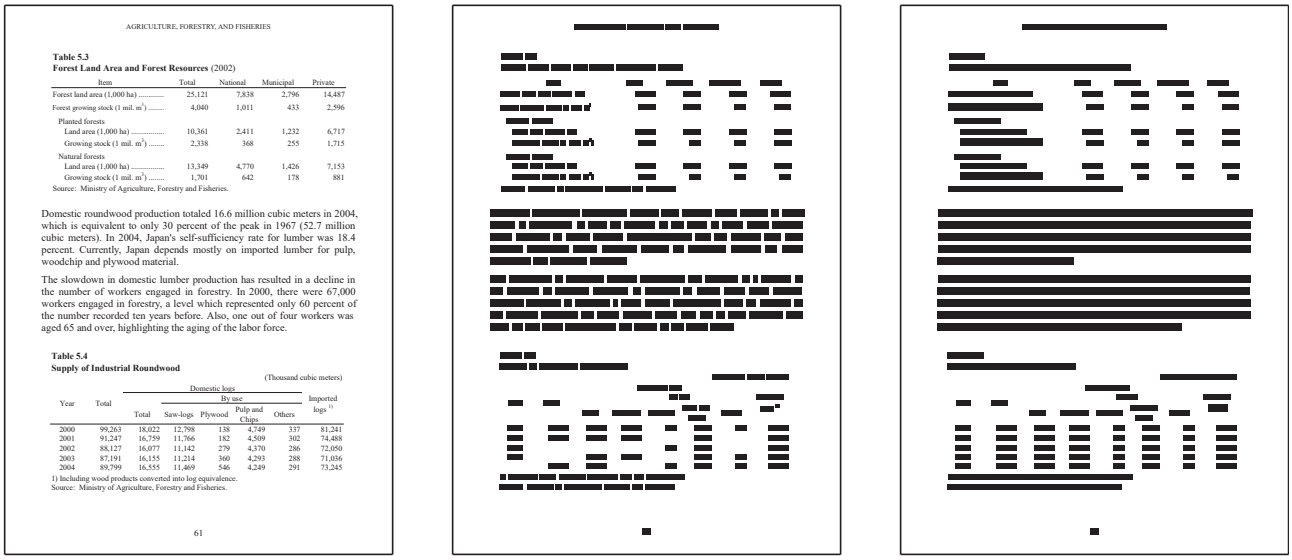
т. е. верхний угол текстового компонента w_2 лежит внутри ограничивающей рамки $\mathrm{bbox}_t(w_1)$, а нижний — внутри $\mathrm{bbox}_b(w_1)$ — (2.7); промежуток пробельного пространства g между текстовыми компонентами w_1 и w_2 не пересекается ни с линейками разграфки $u_r : u_r \in U_r$ — (2.8), ни со следами перемещения пера $u_p : u_p \in U_p$ — (2.9). Рисунок 2.9 иллюстрирует два случая расположения текстовых компонентов: в первом они принадлежат одному блоку, а во втором — нет.

Предполагается, что каждый полученный таким образом блок соответствует целой строке текста. При этом сами блоки могут быть как однокомпо-



а *б*

Рисунок 2.9 — Текстовые компоненты принадлежат одному блоку ($а$) или двум разным ($б$)



AGRICULTURE, FORESTRY, AND FISHERIES

Table 5.3
Forest Land Area and Forest Resources (2002)

Item	Total	National	Municipal	Private
Forest land area (1,000 ha)	25,121	7,838	2,796	14,487
Forest growing stock (1 mil. m³)	4,040	1,011	433	2,596
Planted forests				
Land area (1,000 ha)	10,361	2,411	1,232	6,717
Growing stock (1 mil. m³)	2,338	368	255	1,715
Natural forests				
Land area (1,000 ha)	13,349	4,770	1,426	7,153
Growing stock (1 mil. m³)	1,701	642	178	881

Source: Ministry of Agriculture, Forestry and Fisheries.

Domestic roundwood production totaled 16.6 million cubic meters in 2004, which is equivalent to only 30 percent of the peak in 1967 (52.7 million cubic meters). In 2004, Japan's self-sufficiency rate for lumber was 18.4 percent. Currently, Japan depends mostly on imported lumber for pulp, woodchip and plywood material.

The slowdown in domestic lumber production has resulted in a decline in the number of workers engaged in forestry. In 2000, there were 67,000 workers engaged in forestry, a level which represented only 60 percent of the number recorded ten years before. Also, one out of four workers was aged 65 and over, highlighting the aging of the labor force.

Table 5.4
Supply of Industrial Roundwood

(Thousand cubic meters)

Year	Total	Domestic logs					Imported logs 1)
		Total	By use				
			Saw-logs	Plywood	Pulp and Chips	Others	
2000	99,263	18,022	12,798	138	4,749	337	81,241
2001	91,247	16,759	11,766	182	4,509	302	74,488
2002	88,127	16,077	11,142	279	4,370	286	72,050
2003	87,191	16,155	11,214	360	4,293	288	71,036
2004	89,799	16,555	11,469	546	4,249	291	73,245

1) Including wood products converted into log equivalence.
Source: Ministry of Agriculture, Forestry and Fisheries.

61

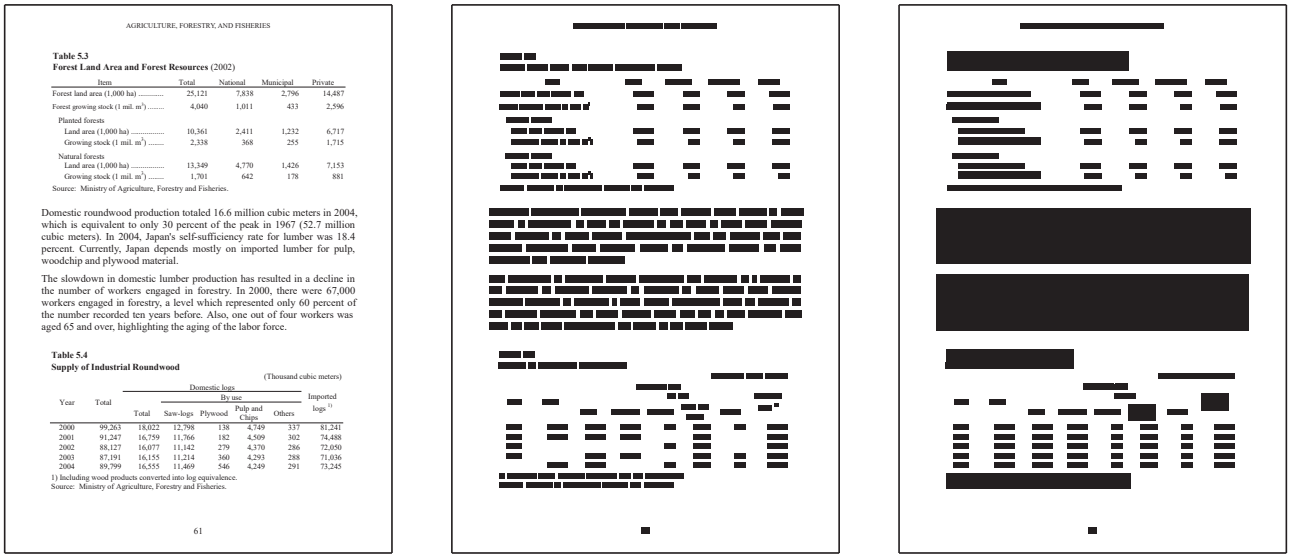
а *б* *в*

Рисунок 2.10 — Обнаружение однострочных блоков текста: исходная страница ($а$); ограничивающие рамки текстовых компонентов ($б$) и блоков ($в$)

нентными, т. е. включать по одному текстовому компоненту, так и многокомпонентными, т. е. включать по несколько текстовых компонентов. Поскольку текстовые блоки не пересекаются между собой, то и обнаруженные строки также не могут пересекаться. Рисунок 2.10 иллюстрирует пример формирования блоков, соответствующих строкам текста.

2.4.4. Группирование текстовых компонентов в многострочные блоки

Обнаружение «абзацев» сводится к объединению текстовых компонентов в многострочные блоки (рис. 2.11). Для этого предлагается адаптировать известный метод обнаружения блоков T-Recs [237, 238] к специфике НПОД.



а *б* *в*

Рисунок 2.11 — Обнаружение многострочных блоков текста: исходная страница ($а$); ограничивающие рамки текстовых компонентов ($б$) и блоков ($в$)

Метод T-Recs позволяет сгруппировать напечатанные слова в блоки внутри ASCII-текста. Его основное достоинство состоит в эффективной применимости к случаям выравнивания текста по ширине. Следует отметить, что именно такое выравнивание часто используется в печатно-ориентированных документах.

Оригинальный метод включает два этапа: начальное объединение напечатанных слов в блоки — (этап I) и коррекцию ложных случаев — (этап II). В настоящем исследовании алгоритм начального объединения блоков был адаптирован, чтобы он стал применимым к предлагаемой модели страницы документа (раздел 2.1), а также предложены собственные алгоритмы коррекции ложных случаев. Следует отметить, что, по сравнению с оригинальными алгоритмами T-Recs, адаптированные используют больше признаков, по которым принимаются решения о группировании напечатанных слов. Последнее позволяет улучшить качество результатов их применения.

Начальное формирование блоков текста

Оригинальная версия (T-Recs) алгоритма начального объединения напечатанных слов принимает в качестве входных данных их строчные и колоночные координаты в ASCII-тексте. В отличие от ASCII-текста, НПОД предоставляет ограничивающие рамки символьных позиций вместо строчных и колоночных координат. Поэтому оригинальный алгоритм невозможно применить к НПОД напрямую, но возможно адаптировать к терминам предлагаемой модели (раздел 2.1).

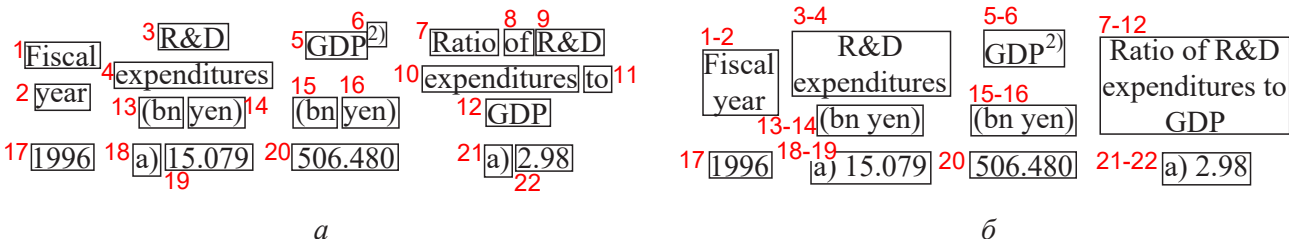
Вход адаптированной версии алгоритма составлен набором текстовых компонентов страницы документа $W = \{w_1, \ldots, w_n\}$, для которых известны их ограничивающие рамки, порядок отрисовки и шрифты символьных позиций. Выходом алгоритма является набор блоков $B = \{b_1, \ldots, b_m\}$, $m \leq n$, образованных из входных текстовых компонентов. Последовательность шагов данного алгоритма представлена ниже.

1. Выбрать случайный текстовый компонент $w : w \in W$.
2. Создать новый кандидатный блок b, добавив его в набор B.
3. Переместить текстовый компонент w в кандидатный блок b.
4. Выбрать всех соседей текстового компонента w — $\bar{W}$, изъяв их из W.
5. Повторить рекурсивно *шаги 3–5* для каждого соседнего текстового компонента $\bar{w} : \bar{w} \in \bar{W}$.
6. Если набор W не пустой, то перейти к *шагу 1*.

Для того чтобы найти соседей текстового компонента w (*шаг 4*), определим прямоугольную область поиска следующим образом:

$$\mathrm{bbox}_s(w) = (x_l(w), y_t(w) - \eta, x_r(w), y_b(w) + \eta)\,,$$

где η — значение высоты захвата (по умолчанию равно удвоенной высоте ограничивающей рамки текстового компонента w). Пусть ограничивающая



а *б*

Рисунок 2.12 — Использование порядка отрисовки текста на странице НПОД: исходные текстовые компоненты ($а$); целевые блоки текста ($б$); индексы в порядке отрисовки символьных позиций, принадлежащих текстовым компонентам ($а$) и блокам ($б$), выделены красным цветом

рамка текстового компонента $\bar{w}$ пересекается с областью поиска $\mathrm{bbox}_s(w)$, тогда $\bar{w}$ включается в кандидатный блок b.

По сравнению с оригинальным методом, настоящая адаптация включает дополнительную проверку кандидатных блоков. Предполагается, что подлинный блок удовлетворяет ряду условий:

1. Порядок отрисовки его текстовых компонентов не должен разрываться.
2. Внутри его ограничивающей рамки нет линеек разграфки.
3. Между его текстовых компонентов нет вертикальных следов перемещения пера.
4. Шрифты его текстовых компонентов согласованы друг с другом.

Первое ограничение базируется на том, что порядок чтения слов одного абзаца текста часто совпадает с порядком их отрисовки. Предполагается, что, когда два текстовых компонента одного блока следуют друг за другом в порядке чтения, тогда конечный индекс в порядке отрисовки символьных позиций первого из них либо меньше начального индекса в порядке отрисовки символьных позиций второго из них (рис. 2.12) ровно на единицу, либо равен ему. Второе ограничение устанавливает, что линейки разграфки не могут пересекать текст одного абзаца. Третье ограничение определяет, что вертикальные следы перемещения пера также не могут пересекать его. Чет-

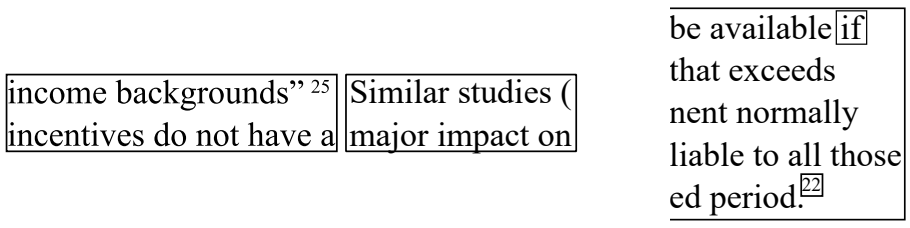
вертое ограничение сопоставляет шрифты текстовых компонентов, размещенных внутри собираемого блока. Предполагается, что шрифты принадлежат одному семейству, а разница их размеров не превышает заданного порога.

В том случае когда все перечисленные условия выполнены, принимается окончательное решение о подлинности блока. Текстовые компоненты, оставшиеся в результате не включенными в какой-либо подлинный блок, повторно используются для сборки блоков. Адаптированный алгоритм начального объединения напечатанных «слов» применяется до тех пор, пока каждому текстовому компоненту не будет сопоставлен некоторый блок.

Коррекция ошибок формирования блоков текста

Как в оригинальной, так и в адаптированной версии начальное объединение напечатанных слов в блоки может приводить к ошибкам: *ложноотрицательным* — «слова» одного абзаца остаются не объединенными в один блок; *ложноположительным* — «слова» двух разных абзацев оказываются объединенными в один блок. Метод T-Recs предлагает некоторые эвристики для коррекции ряда ложных случаев, неизбежно возникающих в результате начального объединения напечатанных слов к ASCII-тексту и поэтому называемых *врожденными* ошибками [237,238]. В отличие от оригинального метода T-Recs предлагаемая адаптация рассчитана на другое представление входных данных. Поэтому ее применение может приводить к другим ошибочным срабатываниям. Рассмотрим три вида ложных срабатываний, характерных для НПОД, и способы их коррекции.

Когда некоторый текстовый компонент w составляет в действительности один абзац совместно с другими напечатанными словами, расположенными вне его области поиска $\mathrm{bbox}_s(w)$. В результате начального объединения напечатанных слов w окажется ложно изолированным в однокомпонентном



а *б*

Рисунок 2.13 — Коррекция ошибок формирования блоков текста: ложные изолированные (*а*) и пересекающиеся (*б*) блоки текста

блоке. Например, этот случай можно наблюдать при наличии однострочных абзацев, а также выравнивании текста по левому/правому краю или центру (рис. 2.13, *а*).

С целью коррекции таких ошибок для каждого блока b определяется две области поиска изолированных соседей, левая и правая, следующим образом:

$$\mathrm{bbox}_l(b) = \big(x_l(b) - \upsilon, y_t(b), x_r(b), y_b(b)\big),$$
$$\mathrm{bbox}_r(b) = \big(x_l(b), y_t(b), x_r(b) + \upsilon, y_b(b)\big),$$

где υ — значение ширины захвата (по умолчанию равно $3/2$ от усредненной ширины пробела, оцениваемой по шрифтам символьных позиций внутри блока b). Выбирается каждый блок $\bar{b}$, захватываемый либо левой, либо правой областью поиска. Пара b и $\bar{b}$ объединяется в один блок, если для последнего выполняются условия подлинности, определенные в разделе 2.4.4. Приведенная процедура выполняется для всех пар соседних блоков.

Другой вид ошибок выражается в том, что ограничивающие рамки текстовых блоков могут пересекаться. Как правило, эта ситуация возникает при использовании различных шрифтов для форматирования разных частей текста одного абзаца. С другой стороны, такая ситуация не обязательно некорректна: часто она проявляется, когда размещение текста не соответствует так называемой *манхэттенской* компоновке, при которой все блоки обязательно вписаны в непересекающиеся прямоугольники. В данной работе принято

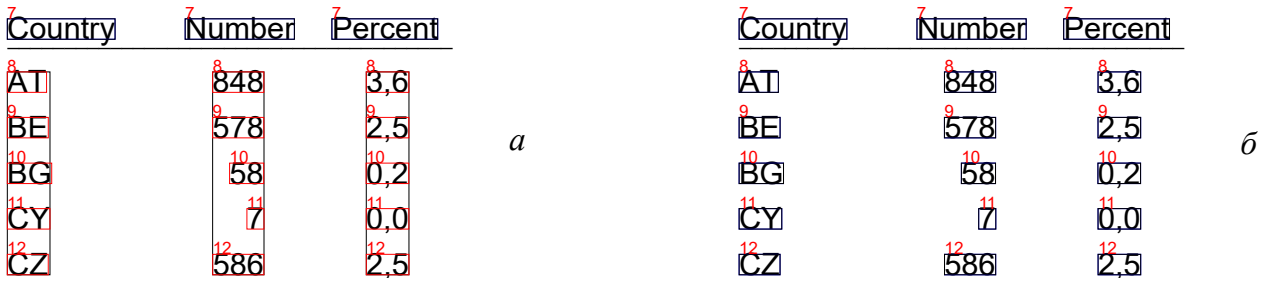


Рисунок 2.14 — Коррекция ошибок формирования блоков текста: ложноположительные многострочные блоки текста (*а*) разделяются на однострочные (*б*) при обнаружении дупликации порядка отрисовки текстовых компонентов; индексы в порядке отрисовки символьных позиций, принадлежащих текстовым компонентам в составе блоков, выделены красным цветом

решение пренебречь случаями неманхэттенской компоновки. В силу данного ограничения любые случаи пересечения ограничивающих рамок блоков рассматриваются как ошибки (рис. 2.13, *б*). Эти случаи исправляются за счет объединения пересекающихся блоков.

Третий случай характерен для текста внутри таблиц — соседние слова одного столбца ложно объединены в один абзац, хотя в действительности они принадлежат разным строкам таблицы (рис. 2.14, *а*). Такие ошибки возникают при построчной отрисовки символьных позиций таблицы, который иногда можно наблюдать в НПОД. При этом текстовые компоненты соседних блоков при построчном сопоставлении будут иметь одинаковый порядок отрисовки. Данная эвристика служит для того, чтобы обнаружить такие ложноположительные многострочные блоки и разделить их на отдельные однострочные блоки (рис. 2.14, *б*).

2.4.5. Сегментация пробельного пространства

Будем называть место внутри страницы документа или ее части, не занятое текстом, *пробельным пространством*. Его можно разделить на ограниченные некоторыми рамками *вертикальные* и *горизонтальные промежут-*

ки. Они расположенны между сторон ограничивающих прямоугольников блоков текста: *вертикальные* — между боковых сторон и *горизонтальные* — между нижних и верхних сторон.

Пусть имеется два блока текста b_l и b_r таких, что их проекции на ось X не пересекаются, при этом первый расположен слева по отношению ко второму, т. е. $x_r(b_l) < x_l(b_r)$. Определим некоторую область g между ними, заданную ограничивающей рамкой, следующим образом:

$$x_r(b_l) \leqslant x_l(g) \text{ и } x_r(g) \leqslant x_l(b_r);$$
$$y_t(g) \leqslant \min\{y_t(b_l), y_t(b_r)\} \text{ и } \max\{y_b(b_l), y_b(b_r)\} \leqslant y_b(g).$$

Будем говорить, что данную область занимает некоторый *вертикальный промежуток пробельного пространства* в том случае, когда нет ни одного блока текста, который пересекался бы с ней внутри ее границ.

Рассмотрим то, как могут быть обнаружены вертикальные промежутки пробельного пространства внутри некоторой части страницы, ограниченной рамкой bbox_s. Пусть внутри нее имеется всего n блоков текста, но по крайней мере два:

$$b_1, \ldots, b_n,\ n > 1.$$

Будем называть их ограничивающие рамки *препятствиями*. Определим множество соответствующих препятствий следующим образом:

$$O = \{o_1, \ldots, o_n\} \mid o_1 = \text{bbox}(b_1), \ldots, o_n = \text{bbox}(b_n).$$

Предполагается, что такие препятствия не пересекаются между собой.

Пусть по боковой стороне (x_l- или x_r-координате) препятствия o можно провести вертикальную линию l, характеризуемую тремя координатами: x, равной $x_l(o)$ или $x_r(o)$ соответственно; некоторыми y_t — верхней и y_b — нижней границами. По боковым сторонам (левой и правой) каждого препятствия $o_i : o_i \in O$ протягиваются две прямые линии вверх и вниз до тех пор, пока они

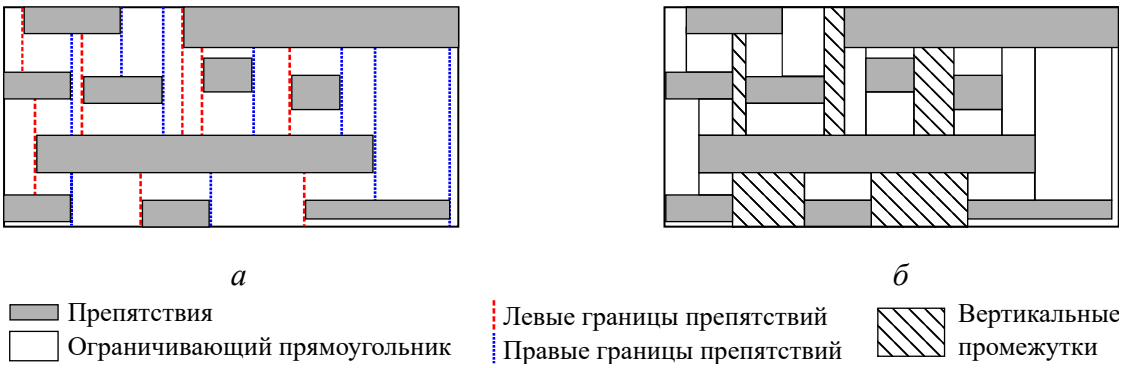


Рисунок 2.15 — Сегментация пробельного пространства страницы документа: нарезка по границам препятствий (*а*); выборка пробельных промежутков (*б*)

не достигнут либо границы другого препятствия, либо границы рамки bbox_s, как показано на рис. 2.15, *а*. Пусть линии, проходящие по правым сторонам препятствий, составляют набор L_r, а линии, проходящие по их левым сторонам, — набор L_l. На втором этапе выбираются пары линий (l_r, l_l): $l_r \in L_r$ и $l_l \in L_l$, удовлетворяющие следующим условиям:

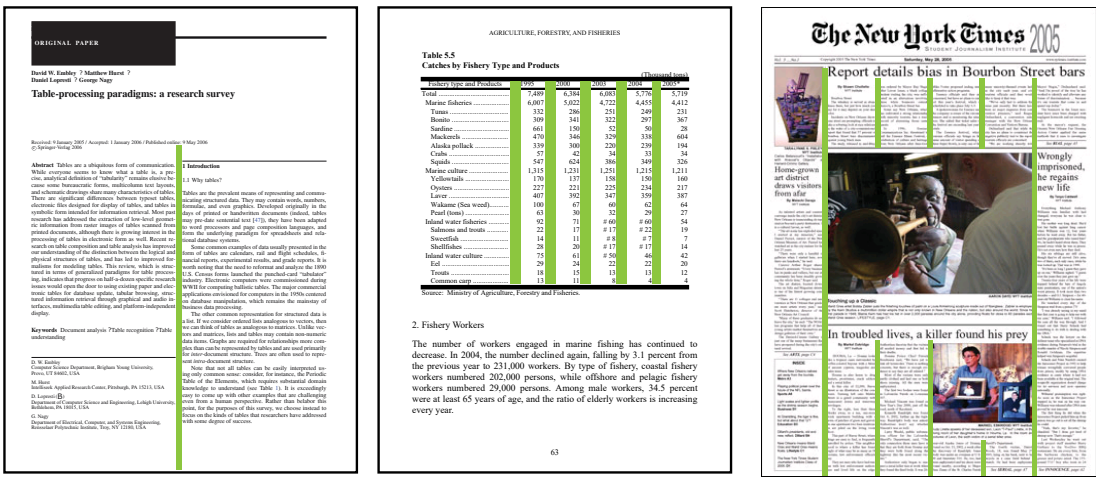
$$x(l_r) < x(l_l), \tag{2.10}$$

$$y_t(l_l) = y_t(l_r) \text{ и } y_b(l_l) = y_b(l_r), \tag{2.11}$$

т. е. линии соответствуют левой и правой границам разных препятствий, причем линия, проведенная по правой границе одного препятствия, расположена слева по отношению к линии, проведенной по левой границе другого препятствия — (2.10), а их верхние и нижние границы попарно совпадают — (2.11). Пусть область g между ними определена следующим образом:

$$g(l_r, l_l) = \big(x(l_r), y_t(l_l), x(l_l), y_b(l_l)\big).$$

Будем считать, что данная область является вертикальным промежутком пробельного пространства тогда и только тогда, когда нет ни одного другого препятствия $o_i : o_i \in O$, пересекающегося с ней внутри ее границ. Пример таких областей, образованных парами линий, проходящих по боковым границам блоков текста, показан на рис. 2.15, *б*.



ORIGINAL PAPER

David W. Embley ? Matthew Hurst ?
Daniel Lopresti ? George Nagy

Table-processing paradigms: a research survey

Received: 9 January 2005 / Accepted: 1 January 2006 / Published online: 9 May 2006
© Springer-Verlag 2006

Abstract Tables are a ubiquitous form of communication. While everyone seems to know what a table is, a precise, analytical definition of "tabularity" remains elusive because some bureaucratic forms, multicolumn text layouts, and schematic drawings share many characteristics of tables. There are significant differences between typeset tables, electronic files designed for display of tables, and tables in symbolic form intended for information retrieval. Most past research has addressed the extraction of low-level geometric information from raster images of tables scanned from printed documents, although there is growing interest in the processing of tables in electronic form as well. Recent research on table composition and table analysis has improved our understanding of the distinction between the logical and physical structures of tables, and has led to improved formalisms for modeling tables. This review, which is structured in terms of generalized paradigms for table processing, indicates that progress on half-a-dozen specific research issues would open the door to using existing paper and electronic tables for database update, tabular browsing, structured information retrieval through graphical and audio interfaces, multimedia table editing, and platform-independent display.

Keywords Document analysis ? Table recognition ? Table understanding

D. W. Embley
Computer Science Department, Brigham Young University, Provo, UT 84602, USA

M. Hurst
Intelliseek Applied Research Center, Pittsburgh, PA 15213, USA

D. Lopresti (B)
Department of Computer Science and Engineering, Lehigh University, Bethlehem, PA 18015, USA

G. Nagy
Department of Electrical, Computer, and System Engineering, Rensselaer Polytechnic Institute, Troy, NY 12180, USA

1 Introduction

1.1 Why tables?

Tables are the prevalent means of representing and communicating structured data. They may contain words, numbers, formulae, and even graphics. Developed originally in the days of printed or handwritten documents (indeed, tables may pre-date sentential text [47]), they have been adopted as word processors and page composition languages, and form the underlying paradigm for spreadsheets and relational database systems.

Some common examples of data usually presented in the form of tables are train schedules, rail and airline schedules, financial reports, experimental results, and grade reports. It is worth noting that the need to reformat and analyze the 1890 U.S. Census forms launched the punched-card "tabulator" industry. Electronic computers were commissioned during WWII for computing ballistic tables. The major commercial applications envisioned for computers in the 1950s centered on database manipulation, which remains the mainstay of business data processing.

The other common representation for structured data is a list. If we consider ordered lists analogous to vectors, then we can think of tables as analogous to matrices. Unlike vectors and matrices, lists and tables may contain non-numeric data items. Graphs are required for relationships more complex than can be represented by tables and are used primarily for inter-document structure. Trees are often used to represent intra-document structure.

Note that not all tables can be easily interpreted using only common sense: consider, for instance, the Periodic Table of the Elements, which requires substantial domain knowledge to understand (see Table 1). It is exceedingly easy to come up with other examples that are challenging even from a human perspective. Rather than belabor this point, for the purposes of this survey, we choose instead to focus on the kinds of tables that researchers have addressed with some degree of success.

AGRICULTURE, FORESTRY, AND FISHERIES

Table 5.5
Catches by Fishery Type and Products

(Thousand tons)

Fishery type and Products	1995	2000	2003	2004	2005*
Total	7,489	6,384	6,083	5,776	5,719
Marine fisheries	6,007	5,022	4,722	4,455	4,412
Tunas	332	286	251	249	231
Bonito	309	341	322	297	367
Sardine	661	150	52	50	28
Mackerels	470	346	329	338	604
Alaska pollack	339	300	220	239	194
Crabs	57	42	34	33	34
Squids	547	624	386	349	326
Marine culture	1,315	1,231	1,251	1,215	1,211
Yellowtails	170	137	158	150	160
Oysters	227	221	225	234	217
Laver	407	392	347	359	387
Wakame (Sea weed)	100	67	60	62	64
Pearl (tons)	63	30	32	29	27
Inland water fisheries	92	71	# 60	# 60	54
Salmons and trouts	22	17	# 17	# 22	19
Sweetfish	14	11	# 8	# 7	7
Shellfishes	28	20	# 17	# 17	14
Inland water culture	75	61	# 56	46	42
Eel	29	24	22	22	20
Trouts	18	15	13	13	12
Common carp	13	11	8	4	4

Source: Ministry of Agriculture, Forestry and Fisheries.

2. Fishery Workers

The number of workers engaged in marine fishing has continued to decrease. In 2004, the number declined again, falling by 3.1 percent from the previous year to 231,000 workers. By type of fishery, coastal fishery workers numbered 202,000 persons, while offshore and pelagic fishery workers numbered 29,000 persons. Among male workers, 34.5 percent were at least 65 years of age, and the ratio of elderly workers is increasing every year.

63

The New York Times 2005

Report details bias in Bourbon Street bars

Wrongly imprisoned, he regains new life

Home-grown art district draws visitors from afar

Touching up a Classic

In troubled lives, a killer found his prey

Рисунок 2.16 — Разделение колонок текста и таблицы промежутками пробельного пространства

Следует отметить, что представленная методика может использоваться как для обнаружения вертикальных, так и горизонтальных промежутков пробельного пространства . В последнем случае они могут быть построены после отображения ограничивающих рамок блоков текста, соответствующего повороту системы координат страницы на 90°. Полученные промежутки могут использоваться для анализа компоновки страниц и распознавания таблиц. В частности, вертикальные промежутки позволяют выделить страничные и табличные колонки (рис. 2.16).

2.5. Обнаружение таблиц на основе правил

Данная методика основана на правилах попарного сопоставления кандидатных табличных строк для выбора тех, которые составляют отдельные таблицы. Она строится как последовательность следующих этапов: сужение области поиска таблиц (раздел 2.5.1); формирование кандидатных табличных строк из блоков текста (раздел 2.5.2); поиск отдельных частей таблиц (начальных табличных регионов) среди полученных строк (раздел 2.5.3); ком-

бинирование самих таблиц (кандидатных табличных регионов) из их частей (раздел 2.5.4). Рассмотрим перечисленные этапы подробнее.

2.5.1. Разделение страницы на области поиска таблиц

Страница документа может содержать объекты, которые заведомо не должны рассматриваться как часть содержимого таблиц, в том числе, заголовки разделов и подразделов, надтабличные надписи и подрисуночные подписи, колонтитулы и «крупные» многострочные абзацы текста. Можно заметить, что такие объекты являются естественными границами *областей поиска таблиц* внутри страницы документа. Предполагается, что каждая область поиска o ограничена некоторой рамкой $\mathrm{bbox}(o)$ внутри страницы документа и охватывает несколько блоков текста $B = \{b_1, \ldots, b_n\}$, $n \in \mathbb{N}$. Если на одной странице документа имеется несколько таких областей, то их ограничивающие рамки не должны пересекаться, т. е. любой блок текста может принадлежать только одной из них.

Среди блоков текста, полученных в результате сегментации страницы документа, выбираются те, которые предположительно являются *разделителями*, образующими естественные границы областей поиска. Есть два критерия для их отбора. Во-первых, текстовый блок считается разделителем, если его содержание удовлетворяет одному из заданных регулярных выражений, например: `[tT]able\s+\d+(\.\d+)*.*` для выбора надтабличных надписей («Table 1.1 ...», «Table 1.2 ...» и т. д.); `[fF]igure\s+\d+(\.\d+)*.*` для выбора подрисуночных подписей («Figure 1.1 ...», «Figure 1.2 ...» и т. д.). Во-вторых, в случае сегментации текста страницы на абзацы (раздел. 2.4.4) текстовый блок считается разделителем, если его ширина совпадает с шириной области поиска и при этом он содержит по крайне мере n строк (по умолчанию 3).

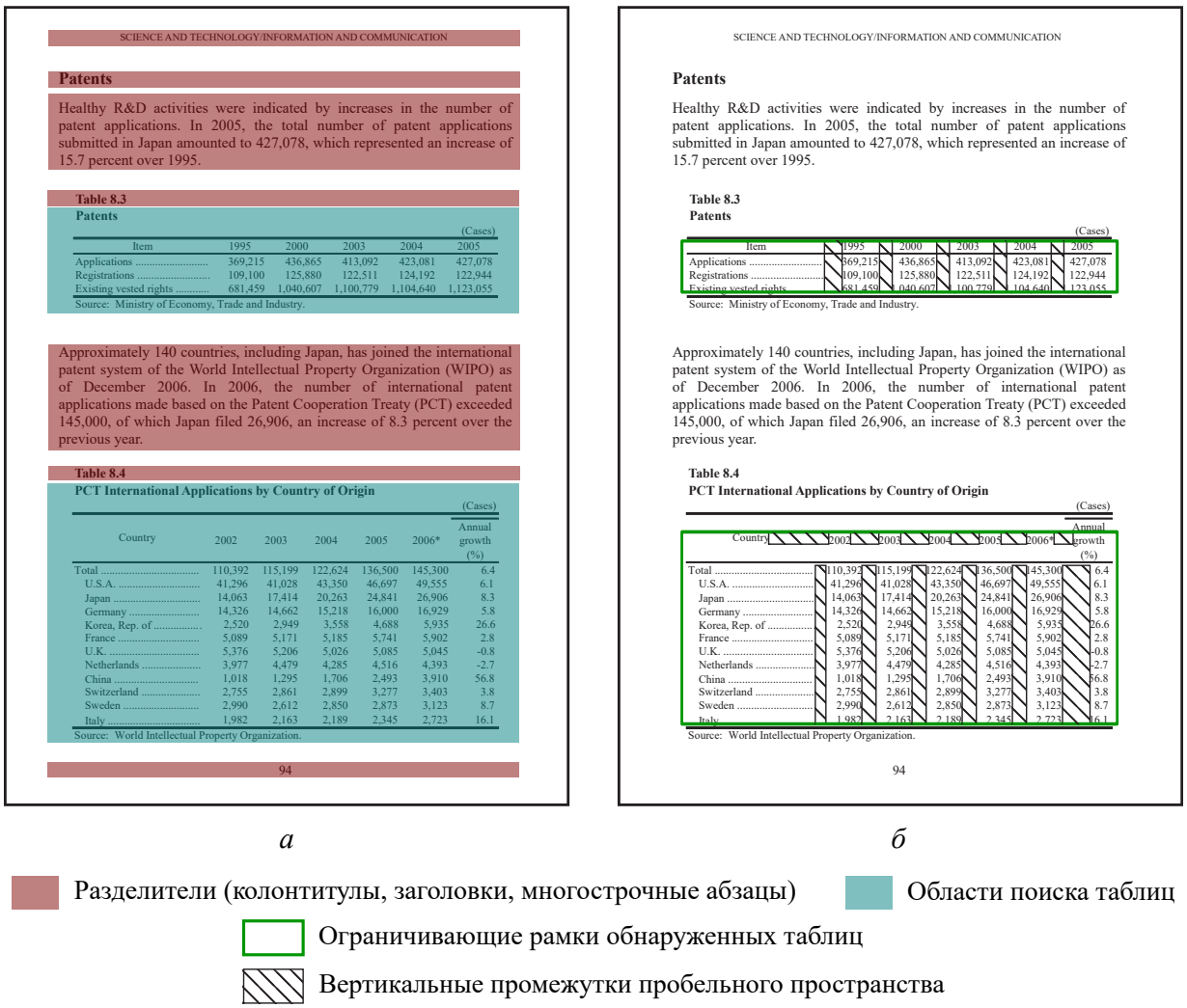


Рисунок 2.17 — Страница документа, разделенная на области поиска ($а$) и обнаруженные таблицы ($б$)

Пусть на странице p в качестве разделителей выбраны следующие блоки текста: $\bar{b}_1, \dots \bar{b}_m$. Предполагается, что они не пересекаются по своим проекциям на ось Y и каждый последующий из них расположен ниже предыдущего:

$$y_b(\bar{b}_1) < y_t(\bar{b}_2), y_b(\bar{b}_2) < y_t(\bar{b}_3), \dots, y_b(\bar{b}_{m-2}) < y_t(\bar{b}_{m-1}), y_b(\bar{b}_{m-1}) < y_t(\bar{b}_m).$$

Тогда распложенные между ними ограничивающие рамки составляют области поиска следующим образом:

$$o_1 = \big(x_l(p), y_t(p), x_r(p), y_t(\bar{b}_1)\big),\ o_2 = \big(x_l(p), y_b(\bar{b}_1), x_r(p), y_t(\bar{b}_2)\big), \dots,$$
$$o_m = \big(x_l(p), y_b(\bar{b}_{m-1}), x_r(p), y_t(\bar{b}_m)\big),\ o_{m+1} = \big(x_l(p), y_b(\bar{b}_m), x_r(p), y_b(p)\big).$$

В результате одна страница может быть разделена на несколько изолированных областей поиска (рис. 2.17), что потенциально упрощает последующее обнаружение таблиц. Следует отметить, что используемый словарь регулярных выражений выбора разделителей по умолчанию рассчитан на стандартные ключевые слова надтабличных надписей и подрисуночных подписей. Однако при необходимости он может расширяться для более специфических случаев распознавания заголовков разделов документов, колонтитулов и пр.

2.5.2. Формирование кандидатных табличных строк

Кандидатные табличные строки формируются из блоков текста заданной области поиска. В действительности они могут являться строками как текста, так и таблиц. Определим их следующим образом. Пусть σ — некоторая область поиска, ограничивающая набор блоков текста — B. Под *кандидатной табличной строкой* l внутри области поиска σ будем понимать пару следующего вида:

$$l = (\bar{B}, \mathrm{bbox}),$$

где $\bar{B} : \bar{B} \subset B$ — упорядоченная последовательность блоков, таких что их x_l-координаты возрастают слева направо:

$$x_l(b_i) < x_l(b_{i+1}),$$

а при равенстве последних их y_t-координаты возрастают сверху вниз:

$$x_l(b_i) = x_l(b_{i+1}), y_t(b_i) < y_t(b_{i+1});$$

ограничивающая рамка строки заключает все ее текстовые блоки:

$$\mathrm{bbox}(l) = (x_l, y_t, x_r, y_b) \mid$$

$$x_l = \min_{0 \leqslant i \leqslant n} x_l(b_i), y_t = \min_{0 \leqslant i \leqslant n} y_t(b_i), x_r = \max_{0 \leqslant i \leqslant n} x_r(b_i), y_b = \max_{0 \leqslant i \leqslant n} y_b(b_i).$$

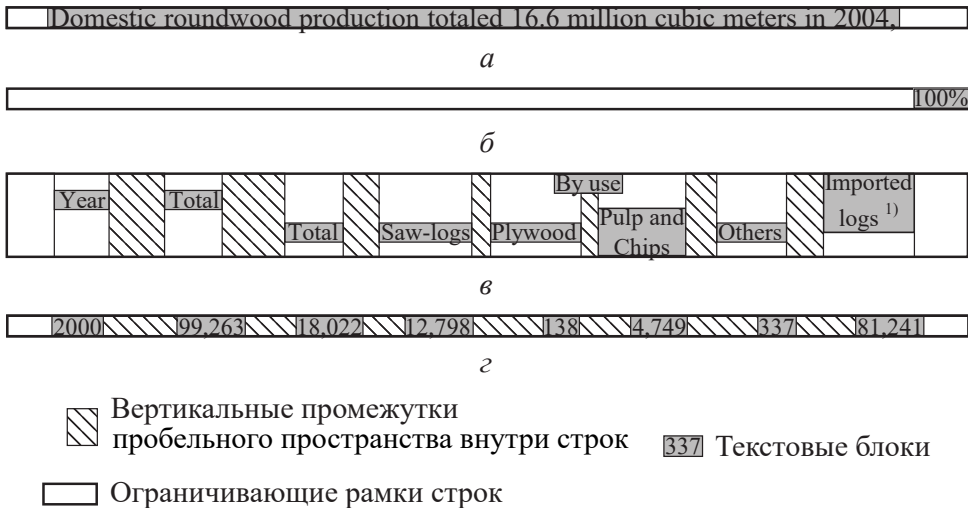


Рисунок 2.18 — Кандидатные табличные строки: однокомпонентные ($а, б$) и многокомпонентные ($в, г$)

Следует отметить, что строки не должны пересекаться между собой по своим ограничивающим рамкам; при этом любой блок текста внутри соответствующей области поиска должен принадлежать только одной из них.

Формирование кандидатных табличных строк внутри заданной области поиска таблиц σ начинается с разбиения множества ее блоков текста B на непересекающиеся подмножества:

$$B_1, \ldots, B_n, n \in \mathbb{N} : B_1 \cup \cdots \cup B_n = B \text{ и } B_1 \cap \cdots \cap B_n = \varnothing,$$

такие что в каждом из них все принадлежащие ему блоки находятся в транзитивном замыкании следующего отношения:

$$b, \bar{b} \in B_i : \; p_y(b) \cap p_y(\bar{b}) \neq \varnothing.$$

Будем считать, что каждое полученное подмножество такого разбиения составляет все блоки некоторой строки. В результате формируются строки, содержащие хотя бы по одному блоку.

Сформированные в результате кандидатные табличные строки могут быть *однокомпонентными* (рис. 2.18, *а–б*) или *многокомпонентными* (рис. 2.18, *в–г*). Определим их следующим образом. Пусть B — набор блоков тек-

ста некоторой области поиска σ, тогда будем говорить, что строка является *однокомпонентной* тогда и только тогда, когда она включает единственный блок $b_i : b_i \in B$, такой что он не пересекается ни с одним другим блоком $b_j : b_j \in B$ по проекциям на ось Y. Пусть B — множество блоков текста некоторой области поиска σ, тогда будем говорить, что строка является *многокомпонентной* тогда и только тогда, когда она включает по крайней мере два блока. Пусть многокомпонентная строка охватывает некоторое подмножество блоков $\bar{B} : \bar{B} \subset B$, тогда для любого из них $b_i : b_i \in \bar{B}$ найдется по крайней мере один другой блок $b_j : b_j \in \bar{B}$, $b_i \neq b_j$, такой что их проекции на ось Y пересекаются.

2.5.3. Формирование начальных табличных регионов

Под *табличным регионом* будем понимать некоторую последовательность подряд идущих кандидатных табличных строк внутри одной области поиска σ. Любая строка может принадлежать только одному табличному региону. Определим табличный регион r как пару следующего вида:

$$r = (L, \mathrm{bbox}),$$

где $L = \{l_1, \ldots, l_n\}$, $n \in \mathbb{N}$ — упорядоченная последовательность подряд идущих строк, таких что

$$y_b(l_i) < y_t(l_{i+1}),$$

ограниченная рамкой:

$$\mathrm{bbox}(r) = (x_l, y_t, x_r, y_b) \mid$$
$$x_l = x_l(\sigma), y_t = \min_{0 \leqslant i \leqslant n} y_t(l_i), x_r = x_r(\sigma), y_b = \max_{0 \leqslant i \leqslant n} y_b(l_i).$$

Табличный регион может являться либо целой таблицей, либо некоторой ее частью. Пример табличного региона, состоящего из трех многокомпонентных строк показан на рис. 2.19.

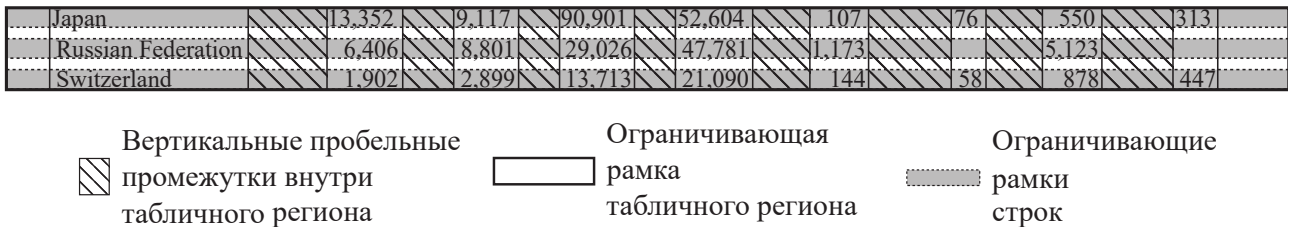


Рисунок 2.19 — Начальный табличный регион, состоящий из трех многокомпонентных строк

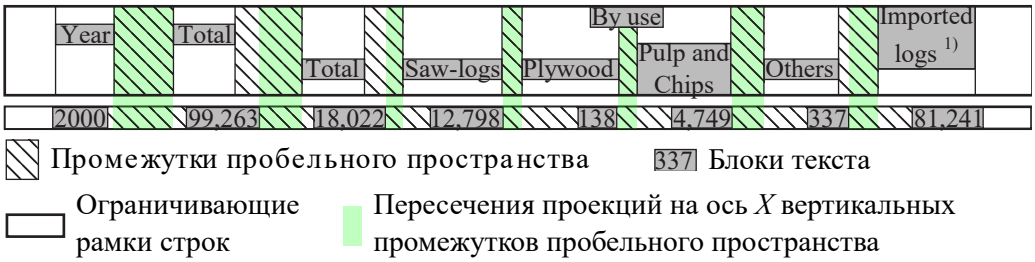


Рисунок 2.20 — Сопоставление соседних многокомпонентных табличных строк по проекциям на ось X вертикальных промежутков пробельного пространства; предполагается, что вертикальных промежутков пробельного пространства в строке снизу, прилегающих к ее верхней границе, будет не меньше, чем в строке сверху, прилегающих к нижней границе, а их проекции на ось X будут попарно пересекаться

Начальные табличные регионы формируются только из многокомпонентных строк внутри заданной области поиска. Предполагается, что каждый из них соответствует некоторой части таблицы. Обнаружение табличного региона выполняется как поиск последовательности подряд идущих многокомпонентных строк, имеющих общее выравнивание вертикальных промежутков пробельного пространства (рис. 2.20).

Для того чтобы выполнить такой поиск, внутри ограничивающей рамки каждой многокомпонентной строки с помощью методики сегментации пробельного пространства (см. раздел 2.4.5) выделяются вертикальные промежутки пробельного пространства, как показано на рис. 2.18, *в*, *г*. Затем подряд идущие многокомпонентные строки сопоставляются между собой, с тем чтобы выявить среди них те последовательности, которые составляют табличные регионы, по ряду условий, представленных ниже.

Пусть многокомпонентная строка l имеет множество вертикальных промежутков пробельного пространства

$$G = \{g_1, \ldots, g_n\}, n \in \mathbb{N}.$$

В этом случае она является строкой некоторого табличного региона тогда и только тогда, когда сумма ширин всех ее пробельных промежутков строки относительно ширины самой строки не превышает заданной пороговой величины:

$$\frac{1}{x_r(l) - x_l(l)} \sum_{i=1}^{n} \big(x_r(g_i) - x_l(g_i)\big) \geqslant \theta,$$

где θ: $\theta \in \mathbb{R}$ и $0 < \theta < 1$. (по умолчанию $\theta = 0,1$).

Определим функцию, которая принимает положительные значения, если проекции на ось X двух вертикальных промежутков g и $\bar{g}$ пересекаются, следующим образом:

$$\mathrm{wp}(g, \bar{g}) = \min\{x_r(g), x_r(\bar{g})\} - \max\{x_l(g), x_l(\bar{g})\}.$$

Пусть вертикальные промежутки пробельного пространства многокомпонентной строки l, у которых верхняя граница совпадает с верхней границей самой строки, составляют следующее подмножество:

$$G_t(l) = \big\{g \mid y_t(g) = y_t(l)\big\}.$$

Пусть вертикальные промежутки пробельного пространства многокомпонентной строки l, у которых нижняя граница совпадает с нижней границей самой строки, составляют следующее подмножество:

$$G_b(l) = \big\{g \mid y_b(g) = y_b(l)\big\}.$$

Тогда последовательность подряд идущих многокомпонентных строк

$$L = \{l_1, \ldots, l_n\},\ n \in \mathbb{N} \mid y_t(l_i) < y_t(l_{i+1}),$$

составляет один табличный регион, если для любой пары

$$l_i, l_j \mid l_i, l_j \in L,\ i < j,$$

выполняется следующее условие:

$$\forall\ g \in G_b(l_i)\ \exists\ \bar{g} \in G_t(l_j) : \mathrm{wp}(g, \bar{g}) \geqslant \theta,$$

где θ: $\theta \in \mathbb{R}$ — заранее заданная пороговая величина — минимально допустимое пересечение проекций на ось X вертикальных промежутков пробельного пространства (по умолчанию, среднее значение среди межсимвольных интервалов пробелов).

2.5.4. Формирование кандидатных табличных регионов

Кандидатный табличный регион может содержать как однокомпонентные, так и многокомпонентные кандидатные табличные строки. Предполагается, что он соответствует целой таблице, поэтому должен состоять по крайней мере из двух строк.

Рассматривая начальные табличные регионы, составляющие одну таблицу, можно заметить, что они имеют общее выравнивание вертикальных промежутков пробельного пространства, как показано на рис. 2.21. Кроме того, между такими табличными регионами, могут располагаться однокомпонентные строки. Данные эвристики используются для того, чтобы обнаружить целевые табличные регионы внутри некоторой области поиска.

Формирование целевых табличных регионов начинается с того, что внутри ограничивающей рамки каждого начального табличного региона выделяются вертикальные промежутки с помощью алгоритма сегментации пробельного пространства (см. раздел 2.4.5). Затем соседние начальные табличные регионы сопоставляются между собой, с тем чтобы выявить среди них те по-

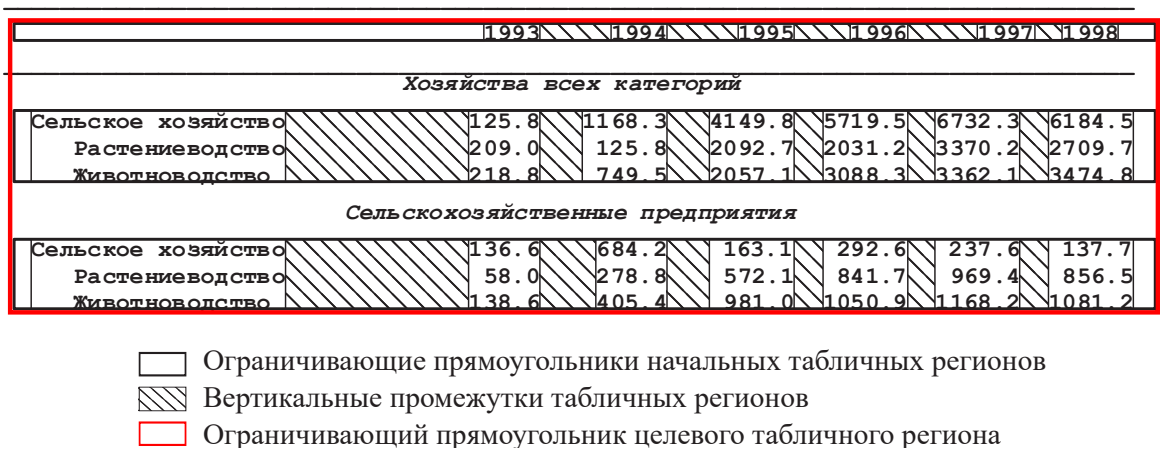


Рисунок 2.21 — Кандидатный табличный регион, составленный из трех начальных табличных регионов

следовательности, которые составляют целевые табличные регионы, по ряду условий, представленных ниже.

Пусть в некоторой области поиска есть два соседних табличных региона r_1 и r_2, таких что первый расположен над вторым, т. е. $y_b(r_1) < y_t(r_2)$. Кроме того, пусть вертикальные промежутки пробельного пространства внутри этих табличных регионов составляют два множества соответственно:

$$G_1 = \{g_{11}, \ldots, g_{1n}\},\ G_2 = \{g_{21}, \ldots, g_{2m}\},\ n, m \in \mathbb{N},\ n \leqslant m.$$

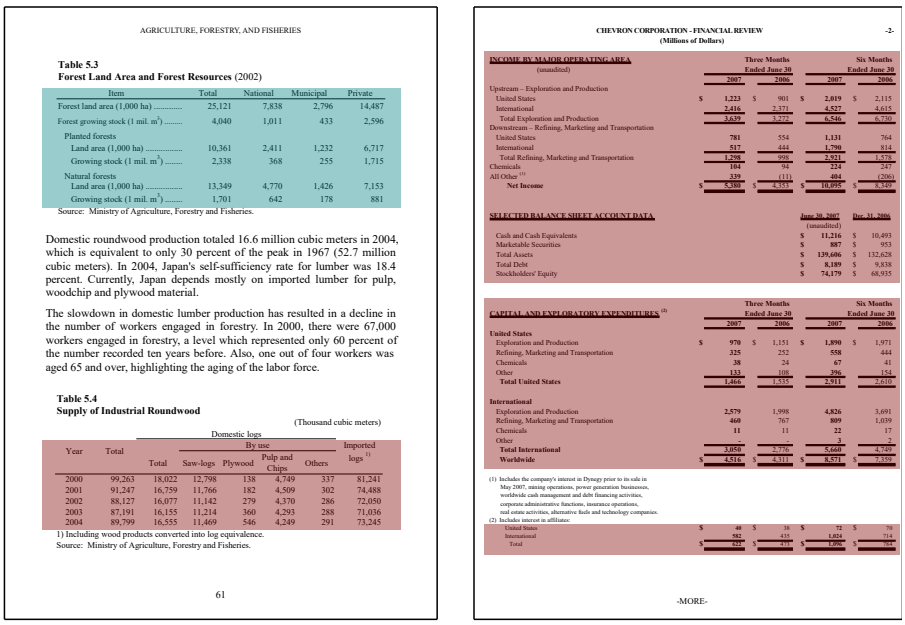
Тогда определим следующую функцию оценки выравнивания вертикальных промежутков пробельного пространства двух табличных регионов следующим образом:

$$\mathrm{rc}(r_1, r_2) = \sum_{i=1}^{n} \mathrm{gc}(g_{1i}, r_2),$$

где

$$\mathrm{gc}(g_{1i}, r_2) = \begin{cases} 1, \text{если существует } g_2 :\ g_2 \in G_2 \text{ и } \mathrm{wp}(g_{1i}, g_2) \geqslant \theta, \\ 0,\ \text{в противном случае,} \end{cases}$$

где $\theta : \theta \in \mathbb{N}$ — заданная пороговая величина (по умолчанию, среднее значение среди межсимвольных интервалов пробелов).



а *б*

Корректные случаи Некорректные случаи

Рисунок 2.22 — Случаи обнаружения таблиц: корректные (сверху) и некорректные (снизу) границы ($а$); одна кандидатная таблица вместо двух истинных (сверху) и две кандидатных таблицы вместо одной истинной (снизу) ($б$)

Будем считать, что два соседних табличных региона r_1 и r_2 являются частями одного целевого табличного региона тогда и только тогда, когда выполняется следующее условие:

$$\mathrm{rc}(r_1, r_2)/|G_1| \geqslant \theta,$$

где $\theta : \theta \in \mathbb{R}$ и $0 < \theta \leqslant 1$ — заданная пороговая величина (по умолчанию, $\theta = 0,8$). Более того, между r_1 и r_2 располагается не более n однокомпонентных строк и m пустых строк (по умолчанию, $n = 2$ и $m = 1$). Следует отметить, что количество пустых строк оценивается по высоте пробельного пространства между непустых строк.

Внутри некоторой области поиска могут быть обнаружены целевые табличные регионы. На первом этапе соседние табличные регионы сопоставляются попарно. Когда такая пара удовлетворяет вышеизложенным условиям, она включается в один кандидатный табличный регион. Предполагается, что

один начальный табличный регион может принадлежать двум целевым табличным регионам. На втором этапе рассматриваются те начальные табличные регионы, которые не вошли ни в один кандидатный табличный регион: каждый многострочный табличный регион становится целевым; однострочные табличные регионы игнорируются.

На рис. 2.22 показаны примеры результатов применения описанного обнаружения таблиц. Каждый кандидатный табличный регион определяет местоположение кандидатной таблицы. В некоторых случаях обнаруженные местоположения кандидатных таблиц некорректны: они могут оказаться ложными или иметь неверные границы.

2.6. Обнаружение таблиц на основе машинного обучения

Альтернативная методика основана на применении предобученных искусственных нейронных сетей (ИНС) обнаружения объектов. Она предусматривает два этапа: вначале используется ИНС, чтобы предсказать ограничивающие рамки кандидатных таблиц внутри страницы документа (раздел 2.6.1); затем может выполняться фильтрация кандидатных таблиц, чтобы исключить возможные ложноположительные предсказания ИНС (раздел 2.6.2). Рассмотрим их подробнее.

2.6.1. Искусственная нейронная сеть обнаружения объектов

Как упоминалось в разделе 1.3.1, сегодня существует большое разнообразие подобных ИНС. В общем случае процесс обнаружения таблиц в НПОД на основе такой ИНС включает следующие шаги: растеризация источника в набор изображений страниц; трансформация полученных изображений с по-

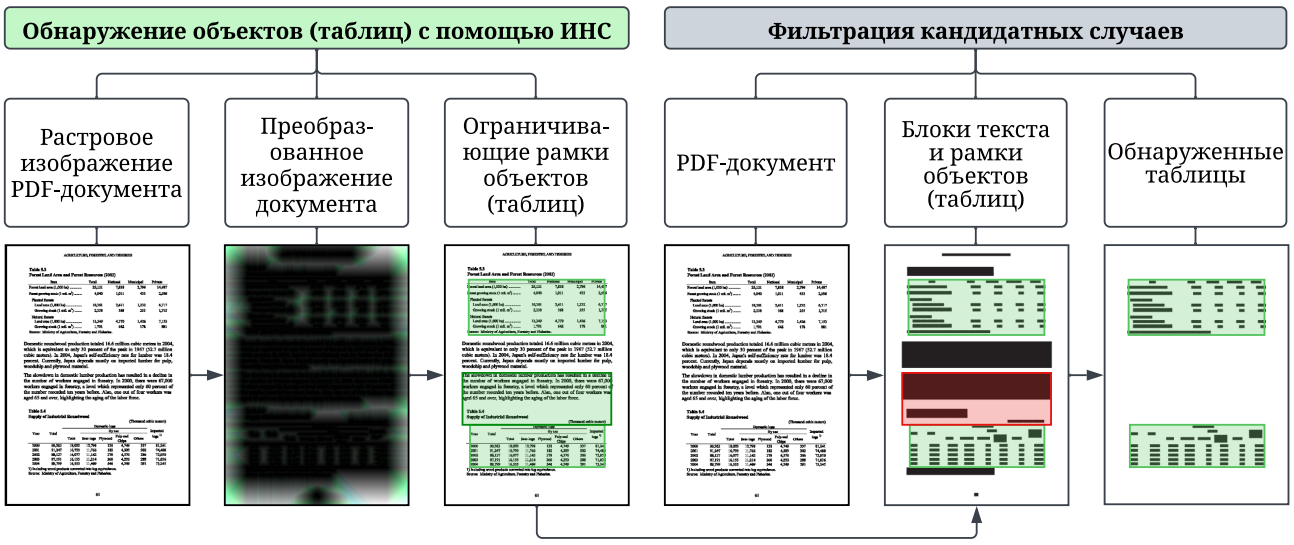


Рисунок 2.23 — Обнаружение таблиц в НПОД на основе ИНС обнаружения объектов и фильтрации кандидатных случаев

мощью процедуры A. Gilani и др. [187]; предсказание ограничивающих рамок кандидатных таблиц ИНС; считывание наборов блоков текста из исходного НПОД, каждый из которых ограничен рамкой отдельной кандидатной таблицы. Рассматриваемый процесс иллюстрируется на рис. 2.23.

Следует отметить, что качество предсказаний ИНС в общем случае зависит от данных, на которых она была обучена. Например, ИНС, обученная на научных статьях, необязательно обеспечит требуемое качество при работе с технической документацией или финансовыми отчетами. Часто полученные ограничивающие рамки кандидатных таблиц являются некорректными. Часть ложноположительных случаев может быть исключена за счет дополнительной фильтрации (раздел 2.6.2).

2.6.2. Фильтрация кандидатных случаев

Данная методика предлагается с целью улучшения точности обнаружения таблиц за счет исключения части ложноположительных результатов ИНС (рис. 2.23). Процесс фильтрации кандидатных случаев включает два шага: *шаг 1* — представить структуру размещения блоков текста внутри огра-

ничивающей рамки таблицы, предсказанной ИНС, в виде графа; *шаг 2* — выполнить бинарную классификацию графового представления, отнеся его либо к истинно-положительным, либо к ложноположительным случаям. Два примера графов кандидатных таблиц представлены на рис. 2.24 и 2.25.

Предполагается, что текстовые компоненты, расположенные внутри ограничивающей рамки некоторой кандидатной таблицы, объединены в многострочные блоки, соответствующие целым абзацам. Само их объединение выполняется на основе адаптированного метода обнаружения блоков T-Recs [237, 238], как описано в разделе 2.4.4. Такие блоки позволяют построить графовое представление кандидатного случая. Каждый блок принимается за вершину этого графа. Размещение блоков в вероятных столбцах таблицы выражается в связности этих вершин. Расстояние между вероятными табличными строками задает веса ребер. Например, когда два связанных блока размещены в двух вероятных строках таблицы, подряд идущих друг за другом, то вес соответствующего ребра равен единице.

Формирование графового представления кандидатного случая начинается с разбиения блоков текста внутри его ограничивающей рамки на вероятные строки. Предполагается, что если проекции на ось Y двух блоков пересекаются, то они принадлежат одной вероятной строке. Будем считать, что каждое транзитивное замыкание блоков, проекции которых на ось Y пресекаются, составляет одну такую строку. Пусть каждой из них сопоставлен ее индекс (номер строки) по порядку их размещения сверху вниз — rpos. Более того, каждому блоку b сопоставлен индекс той вероятной строки, которой он принадлежит, — $\mathrm{rpos}(b)$.

С учетом разбиения блоков текста на вероятные строки, связывание вершин графа выполняется следующим образом. Две вершины (два блока) — b_t и b_b, таких что они расположены в двух разных строках, b_t в верхней, а b_b в

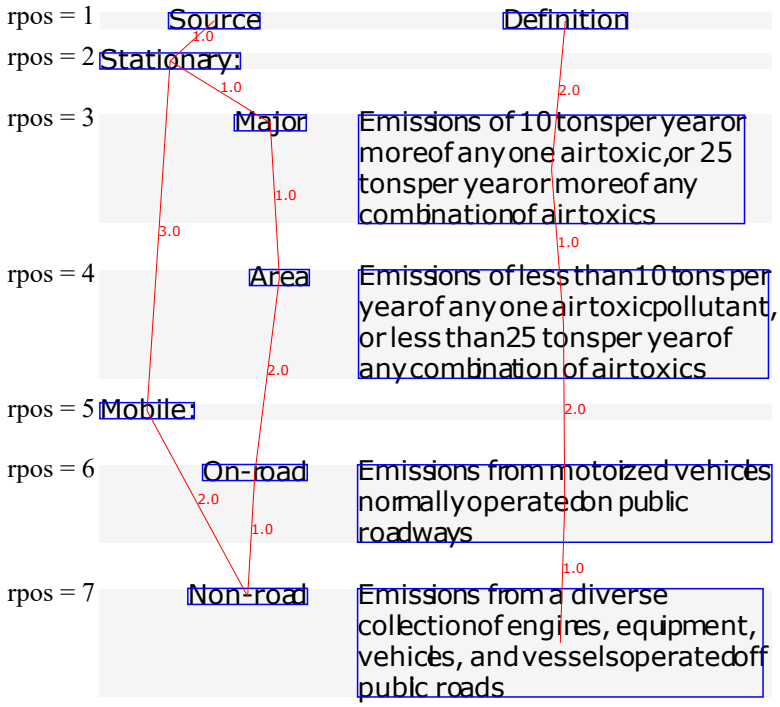


Рисунок 2.24 — Фильтрация кандидатных таблиц: истинно-положительного случай

нижней, следующим образом:

$$\text{rpos}(b_t) < \text{rpos}(b_b),$$

соединяются ребром, если выполняются два следующих условия. Во-первых, их проекции на ось X пересекаются:

$$p_x(b_t) \cap p_x(b_b) \neq \varnothing.$$

Во-вторых, между ними не расположено никаких других блоков:

$$\begin{aligned} \nexists\, b_c : y_b(b_t) < y_t(b_c), y_b(b_c) < y_t(b_b), \\ x_l(b_c) < \max\{x_r(b_t), x_r(b_b)\}, \\ x_r(b_c) > \min\{x_l(b_t), x_l(b_b)\}. \end{aligned}$$

Для каждой пары связанных вершин графа b_t и b_b подсчитывается расстояние между вероятными строками, которым они принадлежат. Полученная величина устанавливает вес ребра:

$$\text{weight}(b_t, b_b) = \text{row}(b_b) - \text{row}(b_t).$$

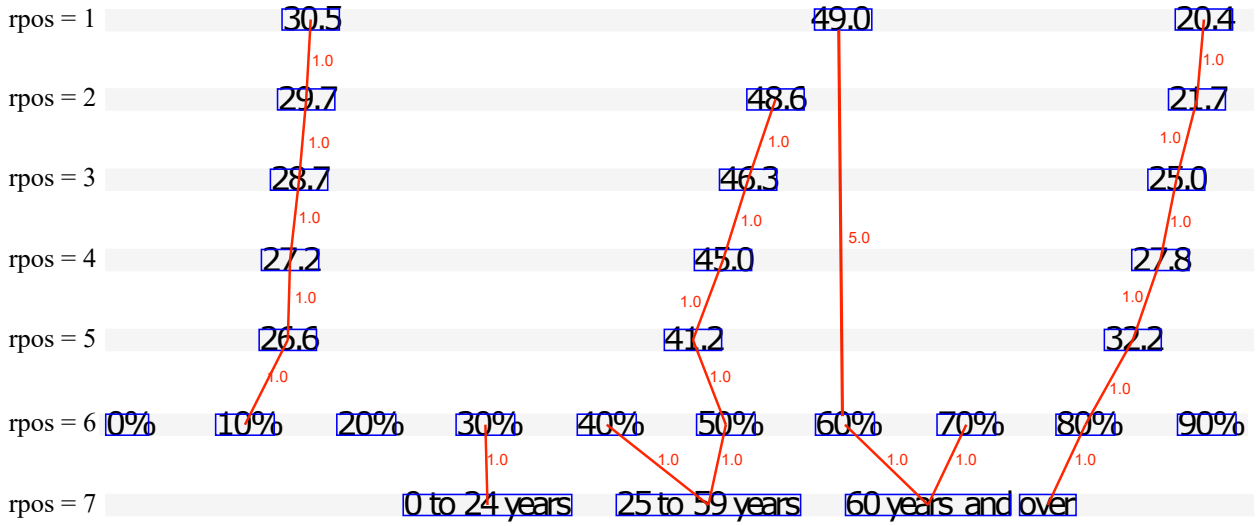


Рисунок 2.25 — Фильтрация кандидатных таблиц: ложноположительный случай

Изучение примеров графового представления кандидатных случаев позволило выделить 35 признаков, которые могут использоваться для предсказания их истинности или ложности. Первые четыре из них следующие: y-координата верхней границы ограничивающей рамки кандидатного случая (`bbyt`) и три количества соответственно вершин (`tbcount`), компонент связности (`cccount`) и ребер (`ecount`) графа. Остальные представлены в табл. 2.2. Они являются агрегациями семи типов (сумма, среднее, максимум, минимум, медиана, среднеквадратическое отклонение, несмещенная дисперсия) над значениями четырех типов: x- и y-координаты середин ограничивающих рамок блоков текста, степени вершин и веса ребер графа.

Предлагаемая фильтрация кандидатных таблиц сводится к бинарной классификации, которая получает вектор значений 35 перечисленных выше признаков графового представления и относит его к одному из двух классов с некоторой вероятностью, а именно либо к истинно-положительным, либо ложноположительным случаям. Обучение классификатора может выполнятся на основе известного алгоритма, например, ансамбля решающих деревьев. Обучающая и тестовая выборка данных могут быть подготовлены следующим образом: *шаг 1* — запуск ИНС обнаружения таблиц в некоторой коллекции

Таблица 2.2 — Признаки графового представления кандидатной таблицы

Агрегации	**x-координаты**	**y-координаты**	**степени**	**веса**
Сумма	`xsum`	`ysum`	`dsum`	`wsum`
Среднее	`xmean`	`ymean`	`dmean`	`wmean`
Максимум	`xmax`	`ymax`	`dmax`	`wmax`
Минимум	`xmin`	`ymin`	`dmin`	`wmin`
Медиана	`xmedian`	`ymedian`	`dmedian`	`wmedian`
Среднеквадратичное отклонение	`xsd`	`ysd`	`dsd`	`wsd`
Несмещенная дисперсия	`xvariance`	`yvariance`	`dvariance`	`wvariance`

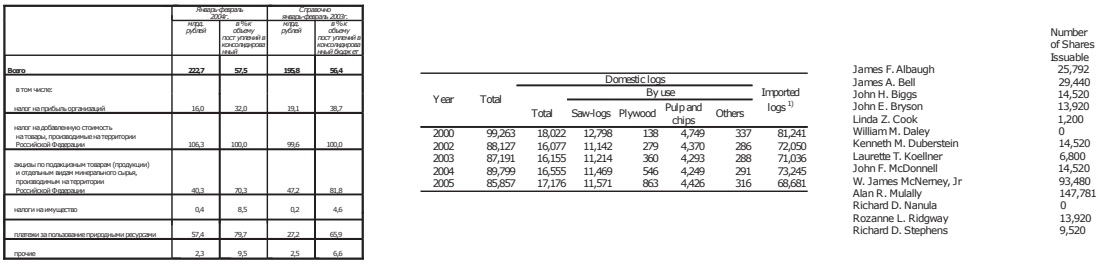
НПОД; *шаг 2* — генерация графового представления каждого кандидатного случая; *шаг 3* — подготовка эталонных данных (пометка кандидатных случаев целевыми классами); *шаг 4* — обучение классификатора на полученной коллекции кандидатных случаев.

2.7. Сегментация таблицы на основе правил

Предлагается две методики обнаружения ячеек, а именно на основе сегментации пробельного пространства и анализа компонент связности графового представления блоков текста внутри таблицы. Первая из них позволяет восстановить линейки разграфки из промежутков пробельного пространства (раздел 2.7.1), а вторая — распознать столбцы и строки как компоненты связности графа блоков текста (раздел 2.7.2). В обоих случаях восстановленная в результате физическая структура позволяет сгенерировать представление таблицы в редактируемом формате (раздел 2.7.3). Рассмотрим их подробнее.

2.7.1. Сегментация пробельного пространства таблицы

Исходная таблица может иметь полную разграфку: представленные линейки отделяют каждую ячейку от остальных (рис. 2.26, *а*). Можно утверждать, что в таких случаях структура таблицы задается ее разграфкой. Однако часто разграфка таблицы представлена только частью линеек (рис. 2.26, *б*) или отсутствует вовсе (рис. 2.26, *в*). Предлагаемая методика состоит в последовательном выполнении следующих шагов: *шаг 1* — восстановить разграфку таблицы по промежуткам пробельного пространства; *шаг 2* — сформировать ячейки таблицы по ее разграфке.



	Январь-февраль 2004г.		Справочно январь-февраль 2003г.	
	млрд. рублей	в % к объему пост. уплачей в консолидированный бюдж.ет	млрд. рублей	в % к объему пост. уплачей в консолидированный бюджет
Всего	222.7	57.5	195.8	56.4
в том числе:				
налог на прибыль организаций	16,0	32,0	19,1	38,7
налог на добавленную стоимость на товары, производимые на территории Российской Федерации	106,3	100,0	99,6	100,0
акцизы по подакцизным товарам (продукции) и отдельным видам минерального сырья, производимым на территории Российской Федерации	40,3	70,3	47,2	81,8
налоги на имущество	0,4	8,5	0,2	4,6
платежи за пользование природными ресурсами	57,4	79,7	27,2	65,9
прочие	2,3	9,5	2,5	6,6

а

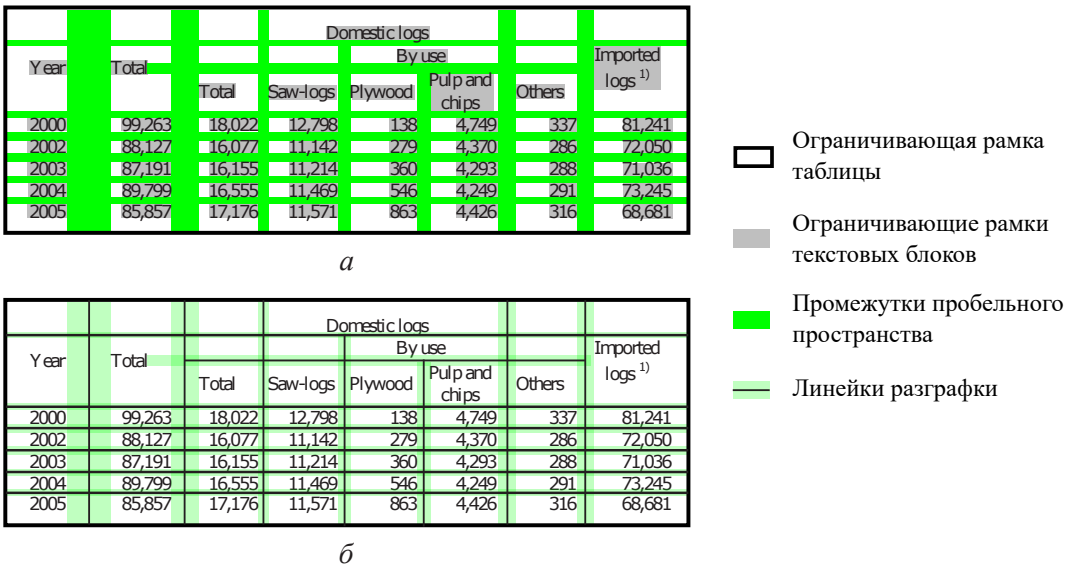
Year	Total	Domestic logs					Imported logs [1]
		Total	By use				
			Saw-logs	Plywood	Pulp and chips	Others	
2000	99,263	18,022	12,798	138	4,749	337	81,241
2002	88,127	16,077	11,142	279	4,370	286	72,050
2003	87,191	16,155	11,214	360	4,293	288	71,036
2004	89,799	16,555	11,469	546	4,249	291	73,245
2005	85,857	17,176	11,571	863	4,426	316	68,681

б

	Number of Shares Issuable
James F. Albaugh	25,792
James A. Bell	29,440
John H. Biggs	14,520
John E. Bryson	13,920
Linda Z. Cook	1,200
William M. Daley	0
Kenneth M. Duberstein	14,520
Laurette T. Koellner	6,800
John F. McDonnell	14,520
W. James McNerney, Jr	93,480
Alan R. Mulally	147,781
Richard D. Nanula	0
Rozanne L. Ridgway	13,920
Richard D. Stephens	9,520

в

Рисунок 2.26 — Варианты разграфки таблиц: полная (*а*) и частичная (*б*) разграфка, без разграфки (*в*)



Year	Total	Domestic logs					Imported logs [1]
		Total	By use				
			Saw-logs	Plywood	Pulp and chips	Others	
2000	99,263	18,022	12,798	138	4,749	337	81,241
2002	88,127	16,077	11,142	279	4,370	286	72,050
2003	87,191	16,155	11,214	360	4,293	288	71,036
2004	89,799	16,555	11,469	546	4,249	291	73,245
2005	85,857	17,176	11,571	863	4,426	316	68,681

а

б

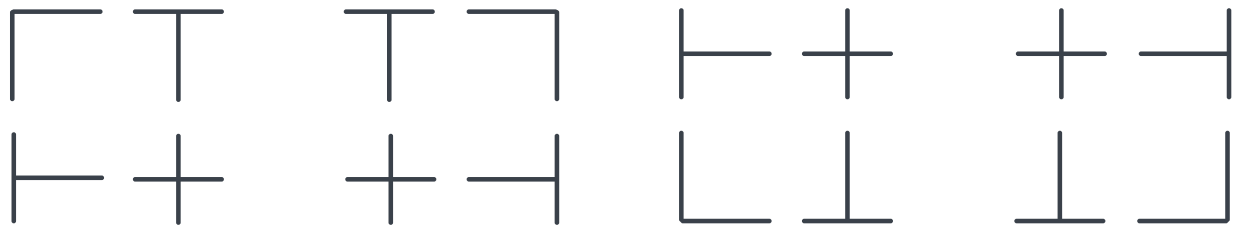
Ограничивающая рамка таблицы

Ограничивающие рамки текстовых блоков

Промежутки пробельного пространства

Линейки разграфки

Рисунок 2.27 — Восстановление разграфки по пробельным промежуткам: сегментация пробельного пространства (*а*) и сформированная разграфка таблицы (*б*)



а *б* *в* *г*

Рисунок 2.28 — Типы пересечений линеек разграфки таблицы соответствуют углам ячеек: левому верхнему (*а*), правому верхнему (*б*), левому нижнему (*в*) и правому нижнему (*г*)

Нетрудно видеть, что линейки разграфки проходят по промежуткам пробельного пространства. Восстановление отсутствующих линеек разграфки сводится к обнаружению вертикальных и горизонтальных промежутков. Последнее можно выполнить с помощью методики, представленной ранее в разделе 2.4.5. Посередине каждого вертикального промежутка пробельного пространства проводится вертикальная линейка. Аналогично, по центру каждого горизонтального промежутка пробельного пространства проводится горизонтальная линейка. Стороны ограничивающей рамки таблицы также рассматриваются как линейки разграфки. В тех случаях когда начальная (конечная) y-координата вертикальной линейки не лежит на какой-либо из горизонтальных линеек, она смещается к ближайшей горизонтальной линейке, с которой она пересекается. Аналогично, в тех случаях когда начальная (конечная) x-координата горизонтальной линейки не лежит на какой-либо из вертикальных линеек, она смещается к ближайшей вертикальной линейке, с которой она пересекается. Таким образом, из полученных линеек строится двумерная решетка, принимаемая за полную разграфку таблицы (рис. 2.27, *б*).

Формирование ячеек выполняется следующим образом. Предполагается, что каждая ячейка образована четверкой пересечений линеек разграфки, соответствующих ее четырем углам: левому верхнему, правому верхнему, левому нижнему и правому нижнему (рис. 2.28). Все пересечения линеек разграф-

ки разбиваются на такие четверки: сперва выбирается базовое пересечение, тип которого может образовывать левый верхний угол (рис. 2.28, *а*); затем выбирается соседнее пересечение справа от базового, тип которого может образовывать правый верхний угол (рис. 2.28, *б*); затем выбирается соседнее пересечение снизу от базового, тип которого может образовывать левый нижний угол (рис. 2.28, *в*); наконец выбирается пересечение снизу и справа от базового, тип которого может образовывать правый нижний угол (рис. 2.28, *г*).

2.7.2. Анализ компонент связности графового представления блоков текста таблицы

Альтернативный способ распознать структуру таблицы состоит в определении связности блоков текста. Введем следующие определения. Два блока считаются связанными, если их проекции либо на ось X, либо на ось Y пересекаются. Пусть B — множество блоков внутри ограничивающей рамки таблицы. Пусть $b : b \in B$ — некоторый блок и $B_x : B_x \subset B$ — подмножество блоков, таких что они пересекаются с блоком b по своим проекциям на ось X. Если существуют два и более таких блоков из подмножества B_x, которые пересекаются между собой по своим проекциям на ось Y, тогда будем говорить, что блок b — *многосвязный* по x-координатам, в противном случае, что он — *односвязный* по x-координатам (рис. 2.29, *а*). Кроме того, пусть $B_y : B_y \subset B$ — подмножество блоков, таких что они пересекаются с блоком b по своим проекциям на ось Y. Если существуют два и более таких блоков из подмножества B_y, которые пересекаются между собой по своим проекциям на ось X, тогда будем говорить, что блок b — *многосвязный* по y-координатам, в противном случае, что он — *односвязный* по y-координатам (рис. 2.30, *а*).

Распознавание структуры таблицы состоит из следующих шагов: *шаг 1* — разделить таблицу на столбцы на основе анализа связности блоков, явля-

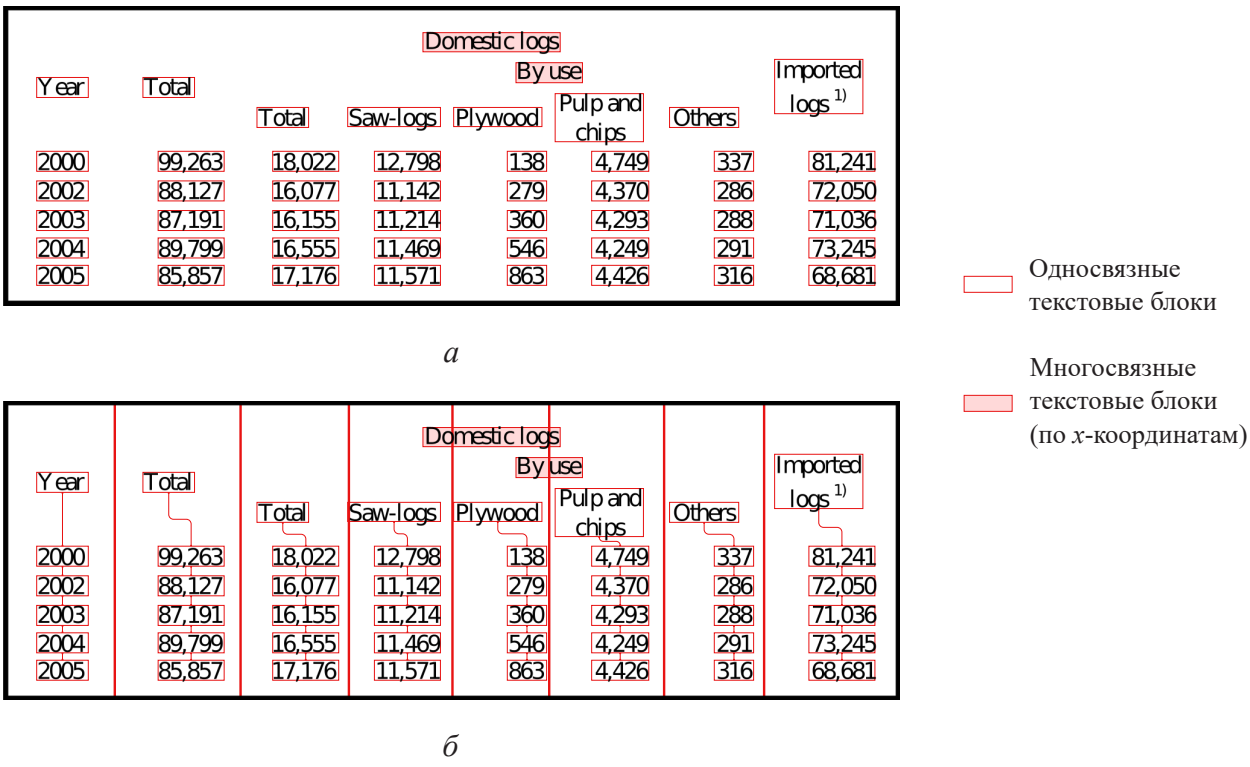
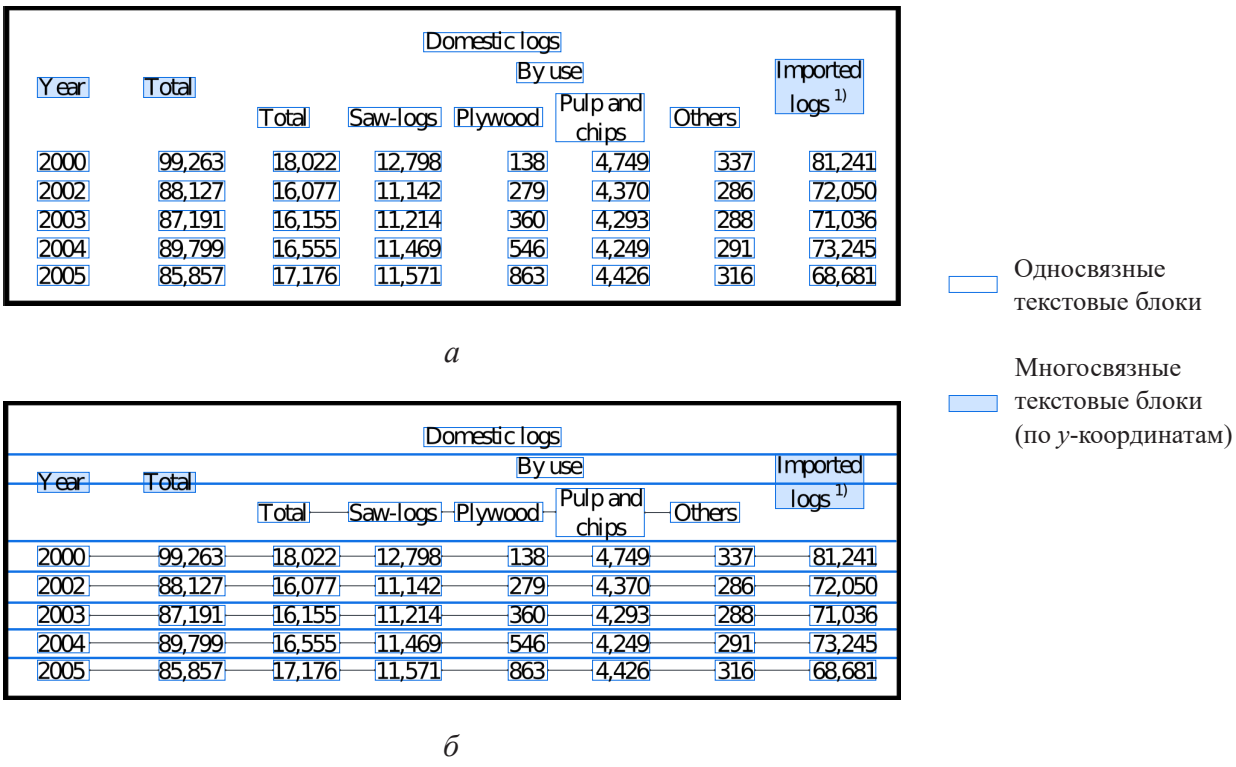


Рисунок 2.29 — Распознавание столбцов таблицы (шаг 1): блоки текста разделяются на односвязные и многосвязные по x-координатам ($а$); каждой компоненте связности сопоставляется отдельный столбец ($б$)

ющихся односвязными по x-координатам; *шаг 2* — разделить таблицу на строки на основе анализа связности блоков, являющихся односвязными по y-координатам; *шаг 3* — построить решетку ячеек.

На первом шаге (рис. 2.29) строится несвязный граф, вершинами которого являются исключительно односвязные по x-координатам блоки, а ребрами — отношения пересечения проекций соседних блоков на ось X. Предполагается, что каждая компонента связности такого графа составляет отдельный столбец — col. Пусть $b_1, \ldots, b_n$, $n \in \mathbb{N}$, — блоки некоторого компонента связности, тогда левая граница соответствующего столбца является минимумом среди левых сторон этих блоков, а его правая граница — максимумом среди правых сторон этих блоков:

$$x_l(\text{col}) = \min_{1 \leqslant i \leqslant n} \{x_l(b_i)\} \text{ и } x_r(\text{col}) = \max_{1 \leqslant i \leqslant n} \{x_r(b_i)\}.$$



а

б

Рисунок 2.30 — Распознавание строк таблицы (шаг 2): блоки текста разделяются на односвязные и многосвязные по y-координатам ($а$); каждой компоненте связности сопоставляется отдельная строка ($б$)

На втором шаге (рис. 2.30) строится другой несвязный граф, вершинами которого являются исключительно односвязные по y-координатам блоки, а ребрами — отношения пересечения проекций соседних блоков на ось Y. Предполагается, что каждая компонента связности такого графа составляет отдельную строку — row. Пусть $b_1, \ldots, b_n$, $n \in \mathbb{N}$ — блоки некоторого компонента связности, тогда верхняя граница соответствующей строки является минимумом среди верхних сторон этих блоков, а его нижняя граница — максимумом среди нижних сторон этих блоков:

$$y_t(\text{row}) = \min_{1 \leqslant i \leqslant n} \{y_t(b_i)\} \text{ и } y_b(\text{row}) = \max_{1 \leqslant i \leqslant n} \{y_b(b_i)\}.$$

На третьем шаге (рис. 2.31) начальное формирование решетки ячеек выполняется из полученных столбцов и строк. Каждое пересечение столбца и строки адресует одну ячейку. При этом столбец определяет левую и пра-

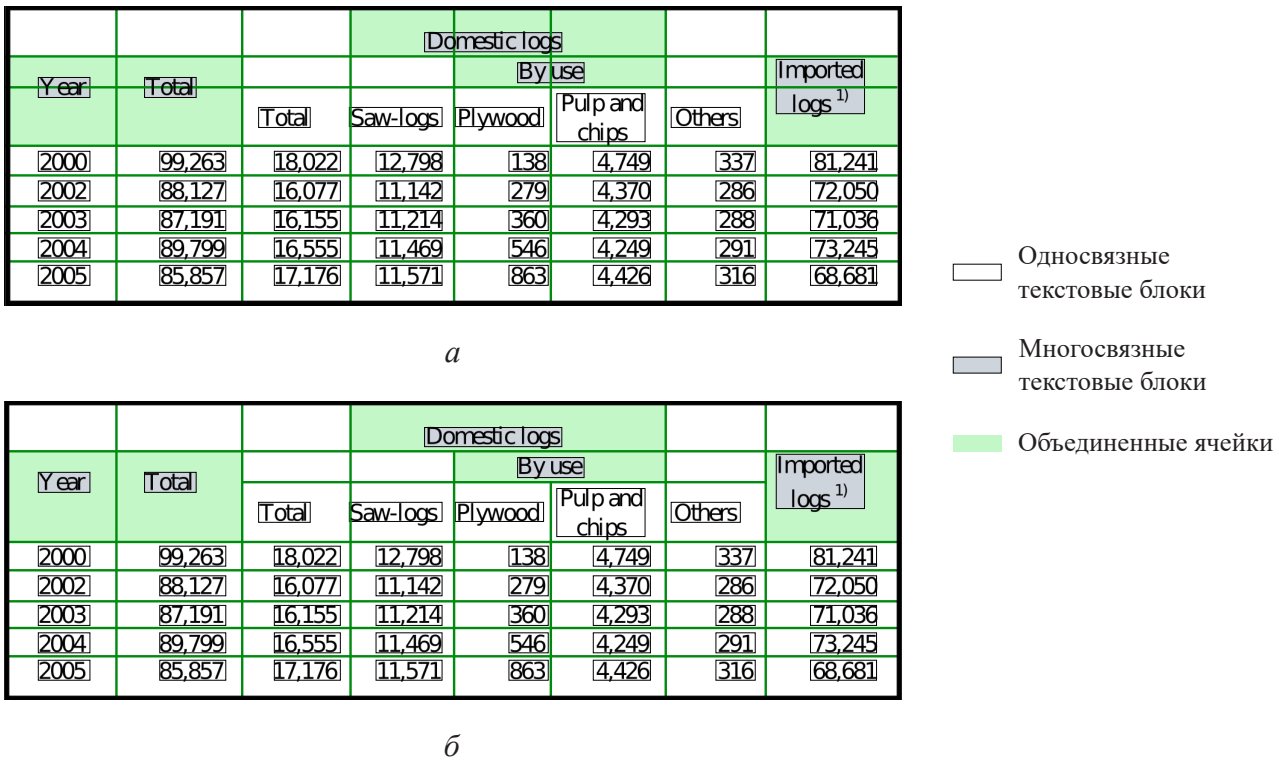


Рисунок 2.31 — Распознавание ячеек таблицы (шаг 3): формируется решетка ячеек ($а$); если n ячеек содержат общий многосвязный блок текста, то они объединяются ($б$)

вую границы ячейки, а строка — ее верхнюю и нижнюю границы. Каждый блок, являющийся односвязным как по x-, так и по y-координатам, сопоставляется с единственной ячейкой. Предполагается, что ограничивающая рамка такого блока полностью лежит внутри границ соответствующих ему столбца и строки. Последующая корректировка решетки ячеек предусматривает объединение тех ячеек, которые охватывают многосвязные блоки. Если ограничивающая рамка некоторого блока, являющегося многосвязным по x- или y-координатам, пересекает n ячеек, то они объединяются в одну.

2.7.3. Формирование редактируемого представления таблицы

Исходные ячейки, полученные в результате применения как первого (раздел 2.7.1), так и второго (раздел 2.7.2) способа сегментации таблицы, характеризуются вещественными координатами своих границ внутри ограничивающей рамки кандидатной таблицы. Полученные границы используют-

```
<table>
<tr>
<td></td>
<td></td>
<td></td>
<td>Domestic logs</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Year</td>
<td>Total</td>
<td></td>
<td></td>
<td>By use</td>
<td></td>
<td></td>
<td>Imported logs 1)</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Total</td>
<td>Saw-logs</td>
<td>Plywood</td>
<td>Pulp and chips</td>
<td>Others</td>
<td></td>
</tr>
<tr>
<td>2000</td>
<td>99,263</td>
<td>18,022</td>
<td>12,798</td>
<td>138</td>
<td>4,749</td>
<td>337</td>
<td>81,241</td>
</tr>
<tr>
<td>2002</td>
<td>88,127</td>
<td>16,077</td>
<td>11,142</td>
<td>279</td>
<td>4,370</td>
<td>286</td>
<td>72,050</td>
</tr>
<tr>
<td>2003</td>
<td>87,191</td>
<td>16,155</td>
<td>11,214</td>
<td>360</td>
<td>4,293</td>
<td>288</td>
<td>71,036</td>
</tr>
<tr>
<td>2004</td>
<td>89,799</td>
<td>16,555</td>
<td>11,469</td>
<td>546</td>
<td>4,249</td>
<td>291</td>
<td>73,245</td>
</tr>
<tr>
<td>2005</td>
<td>85,857</td>
<td>17,176</td>
<td>11,571</td>
<td>863</td>
<td>4,426</td>
<td>316</td>
<td>68,681</td>
</tr>
</table>
```

Рисунок 2.32 — Представление распознанной таблицы в редактируемом формате HTML

ся для формирования ее физической структуры следующим образом. Каждой полученной ячейке c сопоставляется четыре координаты в пространстве строк и столбцов кандидатной таблицы: c_l — левый столбец, r_t — верхняя строка, c_r — правый столбец и r_b — нижняя строка. Если левые или правые границы ячеек совпадают, то они принадлежат общему столбцу. Аналогично, если верхние или нижние границы ячеек совпадают, то они принадлежат общей строке. Любой блок текста, ограничивающая рамка которого лежит внутри границ некоторой ячейки, принимается за часть ее текстового содержимого. Сформированные в результате ячейки могут быть представлены в редактируемом формате данных: HTML, Excel или др. Пример HTML-кода, соответствующего физической структуре распознанной таблицы (рис. 2.31), показан на рис. 2.32.

2.8. Выводы

Имеющиеся методы распознавания таблиц НПОД преимущественно прибегают либо к растеризации источников, либо к их конвертированию в редак-

тируемый формат. Следование любому из этих двух подходов неизбежно приводит к потери информации: в первом случае из-за возможных ошибок OCR-обработки, а во втором — из-за некорректного анализа компоновки документов. По сравнению ними прямое обращение к НПОД предоставляет дополнительную информацию: порядок отрисовки текста в графическом контексте, следы перемещения пера и пр., позволяющую улучшить качество распознавания таблиц в некоторых случаях. Насколько известно из литературы, до сих пор никто не использовал упомянутую информацию при распознавании таблиц. В диссертационном исследовании впервые предложен метод, в котором для решения поставленной задачи применяется именно такая информация.

Известные методы распознавания таблиц в НПОД либо опираются исключительно на ИНС, либо не задействуют их вовсе. В отличие от них метод, предлагаемый в диссертации, позволяет применить оба подхода. Его преимущество заключается в отсутствии необходимости предварительной настройки параметров и обучения с учителем. Однако при наличии подходящих ИНС они могут заменить соответствующие методики на основе правил. При этом невысокая точность ИНС может компенсироваться фильтрацией кандидатных случаев. Полученные результаты представляются в редактируемом формате HTML/Excel. В таком виде они могут подвергаться дальнейшей обработке (глава 3). Программная реализация предлагаемого метода представлена в главе 4, оценка производительности реализованных решений — в главе 5, а их прикладное применение — в главе 6.

Результат, представленный в данной главе

Метод автоматизации распознавания таблиц НПОД (приводимых к формату PDF), т. е. конвертирования их в редактируемый формат (PK). Впервые данный процесс базируется на использовании PDL-специфичной информации, такой как порядок отрисовки текста в графическом контексте, следы пере-

мещения пера и пр. Показано, что такая информация может использоваться методиками как на основе правил, так и машинного обучения.

Публикации

Представленный результат опубликован в ряде рецензируемых изданий, в том числе журналах [2, 4, 12, 55, 59, 60, 63, 64], а также материалах всероссийских и международных мероприятий [41, 57, 58, 62, 65–70].

Глава 3

Автоматизация анализа и интерпретации таблиц

Глава посвящена вопросам автоматизации анализа и интерпретации таблиц РК. Вводятся определения, составляющие модель документной таблицы (раздел 3.1). Рассматривается постановка задачи (раздел 3.2). Предлагается метод ее решения на основе пользовательского программирования правил (раздел 3.3). Определяются состав допустимых условий и действий пользовательских правил анализа и интерпретации таблиц (раздел 3.4), предметно-ориентированный язык таких правил (раздел 3.5) и каноническая форма таблицы для представления результатов их исполнения (раздел 3.6). В конце главы делаются выводы (раздел 3.7).

3.1. Модель документной таблицы

Предлагается модель документной таблицы, обеспечивающая представление физического и логического уровней ее структуры в процессе анализа и интерпретации (рис. 3.1). Предполагается, что оба уровня составлены наборами фактов определенных типов: физический — ячейками, логический — взаимосвязанными функциональными элементами. Для того чтобы обеспечить доступ к таким фактам в процессе исполнения правил, между уровнями определены следующие связи: ячейки ссылаются на функциональные элементы и наоборот.

Физическая структура документной таблицы

Представляет ячейки, включая их координаты в пространстве строк и столб-

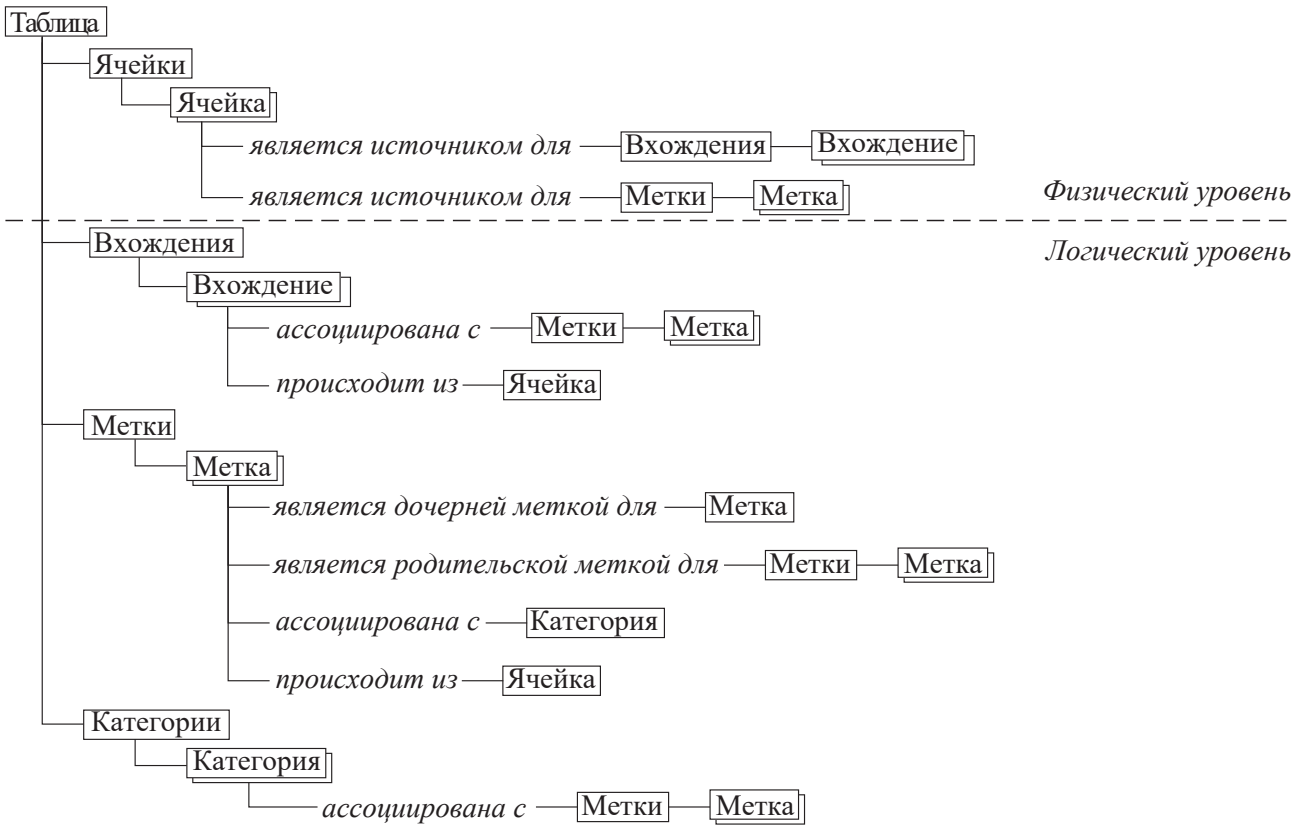


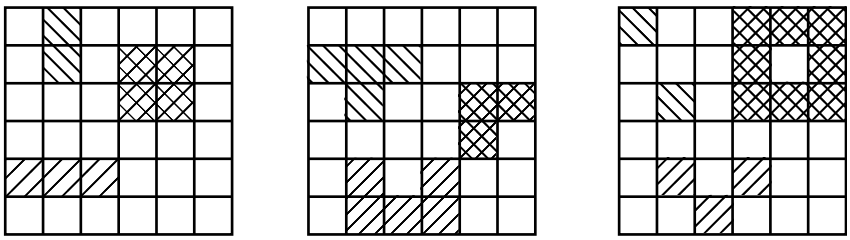
Рисунок 3.1 — Модель документной таблицы

цов, визуальное форматирование и текстовое содержимое. Пусть некоторая таблица состоит из n строк и m столбцов, тогда любая ее ячейка характеризуется четырьмя координатами:

$$c_l, r_t, c_r, r_b \mid c_l \geqslant 1,\ r_t \geqslant 1,\ c_r \geqslant c_l,\ r_b \geqslant r_t,\ c_r \leqslant m,\ r_b \leqslant n,$$

где c_l — левый столбец, r_t — верхняя строка, c_r — правый столбец и r_b — нижняя строка.

Любая ячейка, располагаясь на пересечении одной или нескольких подряд идущих строк и одного или нескольких подряд идущих столбцов, всегда имеет прямоугольную форму, как показано на рис. 3.2, *а*. При этом любые две ячейки одной таблицы никогда не пересекаются. Каждая ячейка характеризуется визуальным форматированием, включая следующее: шрифт (семейство, размер, начертание и цвет), выравнивание текста, визуальные границы (наличие и стили), заливку, индентацию и физические размеры (ширину и высоту).



а *б* *в*

Рисунок 3.2 — Объединенные ячейки: физически объединенная ячейка всегда занимает несколько плиток решетки строк и столбцов, образуя прямоугольник (*а*); ячейка «непрямоугольной» формы может быть представлена визуально с помощью разграфки, однако подобные случаи в работе не рассматриваются как редко используемые (*б*–*в*)

Непустая ячейка имеет единственную содержательную характеристику — размещенный в ней текст. Дополнительно ячейке может быть сопоставлен некоторый пользовательский дескриптор (тег), который обычно используется, чтобы указать ее функциональное назначение (например, заголовок или данные). Ячейке также сопоставлены все функциональные элементы, порожденные из ее текстового содержимого в процессе анализа и интерпретации таблицы.

Упрощенно ячейку можно определить следующим образом:

$$c = \langle c_l, r_t, c_r, r_b, F, v, u, I \rangle ,$$

где c_l, r_t, c_r, r_b — координаты в пространстве строк и столбцов, F — множество характеристик визуального форматирования, v — текстовое содержимое, u — пользовательский дескриптор и I — множество функциональных элементов.

Логическая структура документной таблицы

Представляет функциональные элементы и связи между ними (рис. 3.3). Рассматривается три вида функциональных элементов: *вхождения*, *метки* и *категории*, а также два типа связей: «*вхождение–метка*», «*метка–метка*»

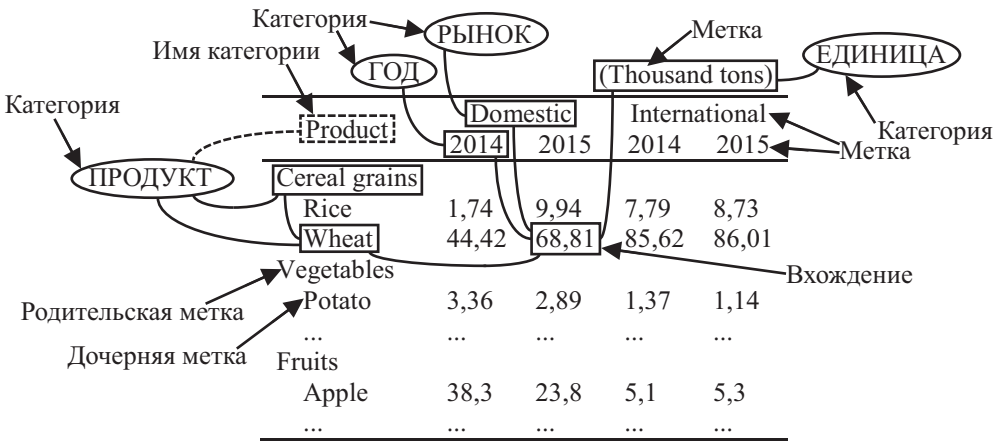


Рисунок 3.3 — Концепции модели документной таблицы

и «*метка–категория*». (Используемая терминология частично заимствована из работы [397].)

Под вхождением понимается значение данных, а под меткой — значение некоторого измерения, описывающего данные. Все метки, соответствующие значениям одного измерения, образуют категорию. Предполагается, что каждое вхождение связано по крайней мере с одной меткой, образуя связь вида «вхождение–метка». Метки организуют иерархию в виде ориентированного дерева при наличии связи вида «метка–метка». Каждая метка принадлежит единственной категории, образуя связь вида «метка–категория». При этом каждое вхождение может быть ассоциировано только с одной меткой в каждой категории, а метки, связанные в дерево, принадлежат одной категории.

Вхождение e характеризуется некоторым значением — v, ячейкой его происхождения — c и множеством ассоциированных с ним меток L:

$$e = \langle v, c, L \rangle .$$

Метка l характеризуется некоторым значением — v, ячейкой ее происхождения — c, некоторой категорией — d, родительской меткой — p при наличии и множеством дочерних меток — G при наличии:

$$l = \langle v, c, d, p, G \rangle .$$

Категория d характеризуется некоторым именем v и множеством принадлежащих ей меток L:

$$d = \langle v, L \rangle .$$

Документная таблица

Документная таблица t является четверкой следующего вида:

$$t = \langle C, E, L, D \rangle ,$$

составленной множествами соответственно: C — ячеек, E — вхождений, L — меток и D — категорий.

3.2. Постановка задачи анализа и интерпретации таблиц

Примем, что любая исходная таблица может быть представлена с помощью введенной предварительно модели. Тогда рассматриваемая задача может быть сформулирована следующим образом:

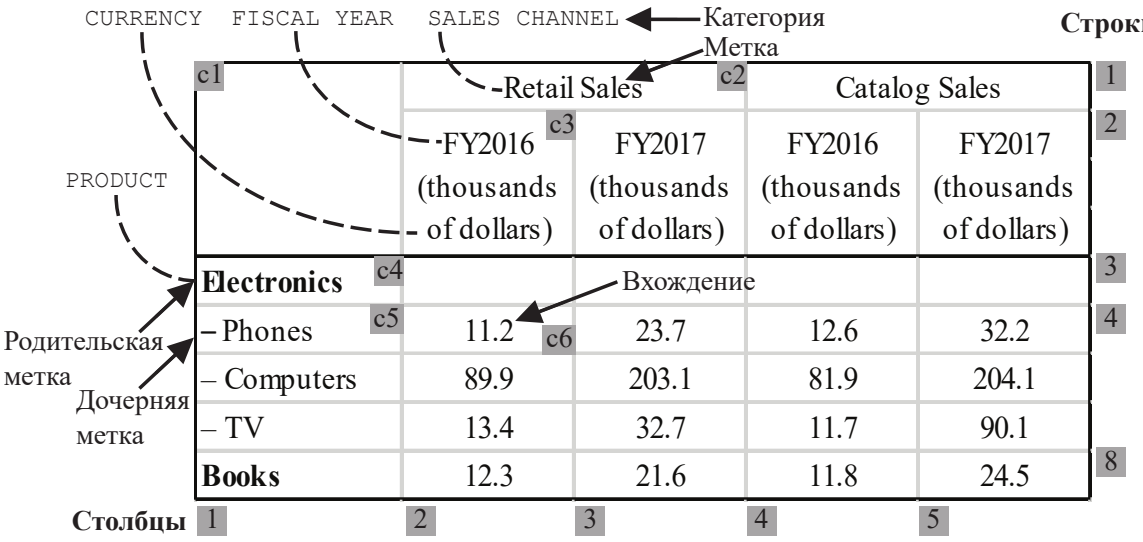
- Имеется таблица t с доступной физической структурой $\phi(t)$, которая визуально представляет набор записей R с общей схемой S.
- Необходимо извлечь набор записей R в канонической форме из t.

Исходные таблицы могут быть представлены в различных редактируемых форматах (Excel, HTML и др.). Каждый из них предоставляет некоторую форматно-зависимую информацию, например: скрытые строки/столбцы, формулы и типы данных — Excel; функциональную разметку и вложенные таблицы — HTML. Тем не менее, хотя и с потерей такой специфики, но они могут быть приведены к предлагаемой форматно-независимой модели. При этом исходная таблица должна удовлетворять следующим ограничениям: физическая структура совпадает с визуальной (человеко-читаемой); любая ячейка может быть наполнена только текстом. Каноническая форма целевого

набора записей предполагает, что одно из его полей перечисляет значения вхождений, а остальные соответствуют категориям: каждое из них перечисляет значения меток отдельной категории. Каждая его запись представляет связь между вхождением и метками всех категорий. Пример исходной таблицы и соответствующего ей набора записей в канонической форме показан на рис. 3.4.

Настоящая задача может быть сведена к задаче отображения физической структуры таблицы в ее логическую структуру. Поскольку последняя обеспечивает автоматический вывод целевого набора записей в канонической форме. Предполагается, что из текстового содержимого каждой ячейки можно породить один или несколько функциональных элементов (вхождений и/или меток), а также сгенерировать имя категории. Из порожденных меток необходимо породить категории (анонимные или именованные) так, чтобы в результате все метки, соответствующие значениям одного измерения, принадлежали той же самой категории. Метки, составляющие единое значение некоторого измерения, следует ассоциировать между собой через родительско-дочерние связи. Наконец, каждое из порожденных вхождений требуется сопоставить с единственной меткой в каждой категории. Пример исходных фактов о физической структуре и целевых фактов о логической структуре показан на рис. 3.4.

Следует отметить, что документная таблица может иметь некоторый контекст, представленный внутри документа. Например, время, место и единицы измерения, описывающие данные некоторой документной таблицы, могут быть представлены именно в контексте. Как отмечалось в разделе 1.3.4, анализ и интерпретация таблицы в таких случаях могут требовать извлечения фактов из контекста и сопоставления их с фактами, содержащимися в самой таблице. Однако настоящая постановка задачи ограничивается анализом и интерпретацией таблиц без вовлечения контекста.



а

```
c1=(cl=1, rt=1, cr=1, rb=2, text=null)
c2=(cl=2, rt=1, cr=3, rb=1, text="Retail Sales")
c3=(cl=2, rt=2, cr=2, rb=2, text="FY2016 (thousands of dollars)")
c4=(cl=1, rt=3, cr=1, rb=3, style.font.bold=true, text="Electronics")
c5=(cl=1, rt=4, cr=1, rb=4, text="- Phones")
c6=(cl=2, rt=4, cr=2, rb=4, text="11.2")
```

б

```
e1=(value="11200", labels={l1,l2,l3,l5}, cell=c6)
l1=(value="retail", category=d1, cell=c2)
l2=(value="2016", category=d2, cell=c3)
l3=(value="u.s. dollars", category=d3, cell=c3)
l4=(value="electronics", children={l5,...}, category=d4, cell=c4)
l5=(value="phones", parent=l4, category=d4, cell=c5)
d1=(name="SALE CHANNEL", labels={l1,...})
d2=(name="FISCAL YEAR", labels={l2,...})
d3=(name="CURRENCY", labels={l3,...})
d4=(name="PRODUCT", labels={l4,l5,...})
```

в

DATA	SALES CHANNEL	FISCAL YEAR	CURRENCY	PRODUCT
11200	retail	2016	u.s. dollars	electronics/phones
23700	retail	2017	u.s. dollars	electronics/phones
12600	catalog	2016	u.s. dollars	electronics/phones
32200	catalog	2017	u.s. dollars	electronics/phones

г

Рисунок 3.4 — Данные процесса анализа и интерпретации таблицы: фрагмент исходной таблицы (*а*); некоторые из фактов о ее структуре: доступные ячейки — `c1`,..., `c6` (*б*) и восстановленные вхождение — `e1`, метки — `l1`,..., `l5` и категории — `d1`,..., `d4` (*в*); фрагмент целевого набора записей в канонической форме (*г*)

3.3. Метод автоматизации анализа и интерпретации таблиц

Наблюдая документные таблицы, опубликованные в одном источнике, можно заметить, что они скорее всего будут следовать общим требованиям к оформлению и содержимому, а именно компоновке ячеек, индентации и выравниванию текста, шрифтам, границам, раскраске, ключевым словам и пр. Более того, такие требования часто разделяются многими источниками, имеющими общее происхождение. Они могут быть стандартизированы на организационном, корпоративном, национальном и международном уровнях. Поэтому можно собрать целые коллекции документов с однообразным оформлением и содержимым таблиц. Приведенное свойство источников может использоваться компьютерными программами, чтобы предсказать реляционную структуру, скрытую в документных таблицах.

Предлагаемый метод базируется на следующей гипотезе: если имеется класс таблиц с общими свойствами компоновки, форматирования и наполнения, то можно разработать некоторый набор правил их анализа и интерпретации (рис. 3.5). Предполагается, что структуру любой таблицы можно представить в виде фактов. Сопоставление их с предоставленными правилами должно отражать физическую часть структуры в логическую. Сам набор

Рисунок 3.5 — Использование общих свойств компоновки, форматирования и наполнения таблиц

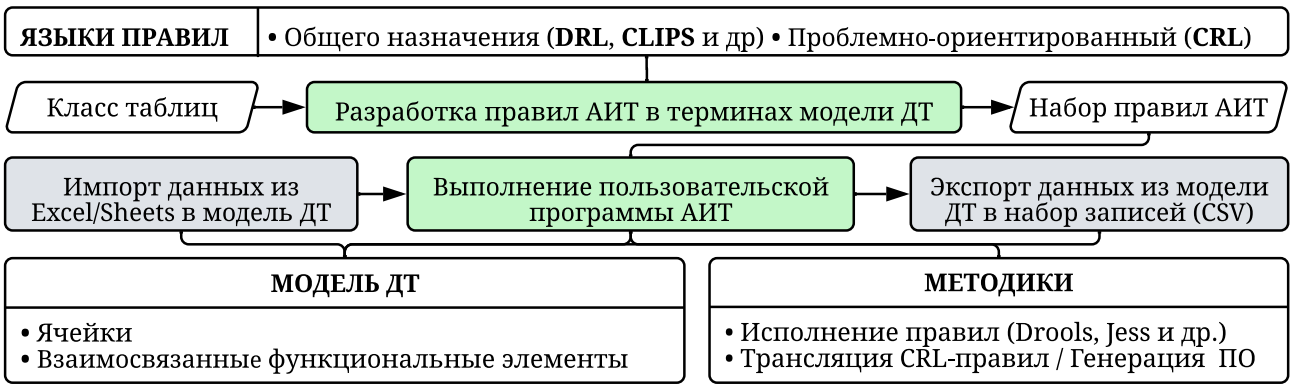


Рисунок 3.6 — Состав и этапы метода автоматизации анализа и интерпретации таблиц РК

правил составляет решение некоторого частного случая приведения таких таблиц к канонической форме.

Предлагаемый метод может быть сформулирован следующим образом:

- Пусть T — класс таблиц с общими свойствами компоновки, форматирования и наполнения такими, что каждому из них можно сопоставить правило κ, отображающее элементы физической структуры в элементы логической структуры.
- Создать набор правил $K = \{\kappa_1, \ldots, \kappa_n\}$, который позволяет отобразить физическую структуру $\phi(t)$ любой таблицы $t : t \in T$ в ее логическую структуру $\lambda(t)$.
- Применить набор правил K к таблице $t : t \in T$, в результате чего должна быть восстановлена логическая структура $\lambda(t)$, обеспечивающая автоматический вывод набора записей R в канонической форме.

Состав и этапы предлагаемого метода схематично показаны на рис. 3.6. Конечное решение извлечения реляционных данных из документных таблиц может быть реализовано на основе пользовательских правил. Их исполнение должно отображать исходные факты о известной физической структуре (компоновке, форматировании и содержимом ячеек) в целевые факты о логической структуре (взаимосвязанных функциональных элементов). Подоб-

ные правила можно выражать на одном из имеющихся формальных языков (DRL[1], CLIPS[2] или др.), а их исполнение производить с помощью совместимой машины вывода (Rete, Leaps или др.). Однако для того, чтобы упростить разработку правил конечными пользователями, предлагается собственный проблемно-ориентированный язык CRL. Правила, записанные на нем, можно транслировать на языки правил общего назначения или процедурного программирования. Для представления структуры таблицы в виде базы фактов предлагается собственная модель (раздел 3.1). Она определяет допустимые условия и действия, которые могут использоваться в правилах. Из восстановленной в результате логической структуры выводится каноническая форма таблицы.

3.4. Правила анализа и интерпретации таблиц

Правила анализа и интерпретации таблиц являются продукционными. Левая часть правила состоит из *условий*, которые позволяют выбрать доступные факты, удовлетворяющие заданным ограничениям (раздел 3.4.1). Правая часть правила содержит *действия*, выполняющие изменение выбранных фактов или создание новых (раздел 3.4.2). В настоящем исследовании определены допускаемые условия и возможные действия. Рассмотрим их подробнее.

3.4.1. Условия

Левая часть правила состоит из одного или нескольких взаимосвязанных условий. Каждое из них проверяет наличие или, напротив, отсутствие фактов одного из типов: ячеек, вхождений, меток и категорий, удовлетворяющих заданным ограничениям. Если выполняются все условия левой части

[1] `https://www.drools.org`

[2] `https://www.clipsrules.net`

некоторого правила, то производятся действия, включенные в правую часть данного правила. При этом выбранные факты в левой части правила доступны для использования в его правой части.

Каждое условие указывает следующие детали: вид — проверка наличия или отсутствия фактов; тип запрашиваемых фактов — ячейки, вхождения, метки или категории; ссылки на запрашиваемые факты при их наличии; список ограничений при необходимости. Ссылки могут использоваться в других условиях и действиях, определенных соответственно в левой и правой частях того же самого правила. Ограничение является логическим выражением. Список ограничений рассматривается как их конъюнкция. Пустой список ограничений означает запрос всех доступных фактов указанного типа.

3.4.2. Действия

Правая часть правила содержит действия, выполняющие изменение фактов, выбранных с помощью условий левой части правила, или создание новых. Возможные действия охватывают задачи очистки, анализа (ролевого и структурного) и интерпретации таблицы. Рассмотрим их подробнее.

Разделение ячеек

Данное действие позволяет разделить объединенную ячейку c, состоящую из n плиток, на n ячеек, каждая из которых полностью дублирует ее содержимое и стилевые характеристики. Координаты i-й целевой ячейки вычисляются по i-й плитке исходной ячейки. Например, если объединенная ячейка имеет координаты $(1, 1, 2, 2)$, то она разделяется на четыре ячейки с координатами $(1, 1, 1, 1)$, $(2, 1, 2, 1)$, $(1, 2, 1, 2)$ и $(2, 2, 2, 2)$. Все ячейки, полученные в результате такого разделения, вносятся в базу фактов, в то время как исходная объединенная ячейка удаляется из нее.

Следует отметить, что данное действие приходится выполнять, когда ячейка, состоящая из n плиток, содержит некоторое вхождение. В таком случае дублирование вхождения n раз обеспечивает его сопоставление с несколькими метками одной категории.

Объединение ячеек

Противоположное предыдущему действие состоит в объединении двух соседних ячеек c_1 и c_2 с общей стороной. Объединенная ячейка c_m принимает координаты, охватывающие обе ячейки. Ее содержимое составляется либо из текста обеих ячеек, когда эти тексты разные, либо из текста только одной из них, в противном случае. При этом сами объединяемые ячейки удаляются из базы фактов.

Данное действие может производиться для объединения ячеек с общим содержимым, например, когда две соседние ячейки содержат две разные части одной метки. Кроме того, серия пустых ячеек часто может неявно повторять содержимое некоторой непустой ячейки. В таких случаях можно восстановить объединенные ячейки с целью расширения класса таблиц, описываемого одним набором правил.

Пользовательская разметка ячеек

Действие привязывает к ячейке c некоторый пользовательский дескриптор. В других правилах такие размеченные ячейки могут запрашиваться без необходимости дупликации ограничений. Обычная практика состоит в сопоставлении общих дескрипторов тем ячейкам, которые выполняют одинаковую функцию. Например, если таблица состоит из таких регионов, как шапка, боковик и тело, то ячейкам в зависимости от их месторасположения могут быть сопоставлены следующие дескрипторы: *head*, *stub* и *body* соответствен-

но. Данное действие позволяет разделить ячейки на подмножества, которые затем анализируются и интерпретируются различными цепочками правил.

Порождение вхождений и меток

Действие обеспечивает создание функциональных элементов, привязывая их к некоторой ячейке c. Порождаемое вхождение e или метка l принимает в качестве своего значения либо текстовое содержимое ячейки c как есть, либо результат некоторой его обработки. В результате созданные вхождения и метки добавляются в базу фактов. При этом они и ячейка их происхождения взаимно ссылаются друг на друга. Поэтому, зная ячейки, можно запрашивать в базе фактов порожденные из них функциональные элементы и, наоборот, зная последние, — запрашивать ячейки.

Категоризация меток

Предлагается два способа категоризации меток. В первом из них метка l ассоциируется с выбранной категорией d, доступной в базе фактов. Во втором выполняется поиск категории по заданному имени. Если такая категория доступна в базе фактов, то она ассоциируется с меткой l. В противном случае в базе фактов создается новая категория с заданным именем, которая затем ассоциируется с меткой l.

Ассоциирование меток

Две метки могут быть ассоциированы друг с другом как родительская l_i и дочерняя l_{i+1}. При этом формируются составные значения меток следующим образом. Если в дереве меток существует некоторый путь $l_1,\ldots,l_n$, где l_1 — его корень, то составное значение метки l_n включает значения всех узлов в этом пути в порядке их вложенности, начиная с корня, следующим обра-

зом: $l_1 \rightarrow \ldots \rightarrow l_n$. Полученные составные значения могут использоваться при генерации выходной канонической формы.

Группирование меток

Рассматриваемое действие добавляет две метки l_1 и l_2 в одну группу — некоторый набор меток. Когда одна из этих меток уже принадлежат некоторой группе, то вторая также добавляется в нее. Если обе ячейки принадлежат двум разным группам, то обе группы объединяются в одну. Следует отметить, что если сгруппированная метка ассоциирована с некоторой категорией, то все остальные метки из той же самой группы также должны принадлежать данной категории.

Применение данного действия связано с тем, что часто можно установить принадлежность нескольких меток одной категории, не зная имя последней. Например, в сводных таблицах часто каждая строка внутри шапки, каждый столбец внутри боковика формируются метками отдельной категории. Группы, метки которых не сопоставлены категориям, рассматриваются как анонимные категории. В результате метки каждой такой группы формируют отдельную категорию с автоматически сгенерированным именем.

Ассоциирование вхождений

Данное действие связывает вхождение e с меткой l, выбранной из базы фактов. Предполагается, что каждая метка, сопоставленная вхождению, должна быть категоризирована. Поэтому если метка l не связана с категорией d, то ассоциирование вхождения e откладывается. Вместо этого данная метка становится кандидатом, попытка ассоциировать вхождение e с которым выполняется только после ее категоризации.

В том случае когда нет возможности выбрать необходимую метку l из базы фактов, данное действие может выполняться как поиск метки с заданным значением v в категории d, выбранной из базы фактов. При наличии искомой метки она ассоциируется с вхождением e, в противном случае сперва она создается в категории d, только после чего выполняется их связывание. Более того, когда нет также возможности выбрать необходимую категорию d из базы фактов, данное действие может выполняться как поиск категории с заданным именем n в базе фактов. Если категория доступна, то она используется далее, как описано выше, для поиска метки с заданным значением v, в противном случае она создается и далее используется для создания метки со значением v.

Следует отметить, что две последние формы данного действия позволяют создавать метки независимо от ячеек, определяя их значения непосредственно в правилах. Последнее используется в тех случаях, когда метки не могут быть сгенерированы из содержимого ячеек. Например, время, место и единицы измерения, не представленные в самой таблице, но известные из ее контекста.

3.5. Предметно-ориентированный язык правил CRL

Правила анализа и интерпретации таблиц могут быть выражены на языках правил общего назначения, включая Drools[3] и Jess[4]. Однако поскольку такие правила ограничены предметной областью — анализом и интерпретаций таблиц, в диссертационном исследовании предложен предметно-ориентированный язык CRL[5]. По сравнению с языками правил общего назначения, предоставляется упрощенный синтаксис, ориентированный на конечных поль-

[3] `https://www.drools.org`

[4] `https://www.jessrules.com`

[5] `https://github.com/tabbydoc/tabbyxl/wiki/crl-language`

зователей (приложение А). CRL скрывает несущественные для данного предмета детали и позволяет сосредоточиться на логике анализа и интерпретации таблиц. Ниже рассматривается CRL-синтаксис.

3.5.1. CRL-правила

Набор правил состоит из *импортов* и *правил* (рис. 3.7, *а*). *Импорт* соответствует инструкции статического импорта Java (рис. 3.7, *б*). Импортированные статические функции и константы становятся доступными в правилах. *Правило* состоит из двух частей: левой и правой (рис. 3.7, *в*)). Левая часть (`when`) включает одно или несколько *условий*; правая (`then`) — одно или несколько *действий*. Когда все условия, размещенные в левой части правила, истинны, выполняется правая часть правила. Целочисленный идентификатор правила `id`, следующий за ключевым словом `rule`, используется, чтобы установить порядок исполнения правил. Ниже представлена форма правила:

```
rule #id
    when
        condition_1
        ...
        condition_n
    then
        action_1
        ...
        action_m
end
```

Следует отметить, что поскольку предлагаемая модель таблицы обеспечивает связи между физической и логической частями ее структуры, то через ячейки можно получить доступ к связанным с ними функциональным элементам («вхождениям» и «меткам»), и наоборот, через «вхождения» и

«метки» можно получить доступ к ячейкам, из которых они происходят. В CRL-правилах такие обращения записываются с помощью точечной нотации.

3.5.2. Условия CRL-правил

Синтаксически *условие* состоит из двух частей (рис. 3.7, *г*). Первая часть определяет *тип запрашиваемых фактов* и связывает с ними переменную при необходимости; вторая опционально перечисляет *ограничения*, накладываемые на факты, — логические выражения Java, разделенные запятой в качестве логического «и», а также может включать *присваивания* — дополнительные переменные и их значения.

Условие позволяет запрашивать доступные факты одного из заданных типов: ячейки, вхождения, метки и категории (табл. 3.1). Предлагается два вида условий: *наличия* и *отсутствия* фактов. Условие первого рода требует, чтобы существовал хотя бы один факт указанного типа (`cell`, `entry`, `label` и `category`), который удовлетворяет набору ограничений. Напротив, условие второго рода требует обратного: чтобы не существовало ни одного факта указанного типа (`no cells`, `no entries`, `no labels` и `no categories`), удовлетворяющего заданным ограничениям.

Таблица 3.1 — Типы условий в CRL-правилах

Запрашиваемые факты	**Вид условия**	
	наличие фактов	**отсутствие фактов**
Ячейки	`cell var: constraints`	`no cells: constraints`
Вхождения	`entry var: constraints`	`no entries: constraints`
Метки	`label var: constraints`	`no labels: constraints`
Категории	`category var: constraints`	`no categories: constraints`

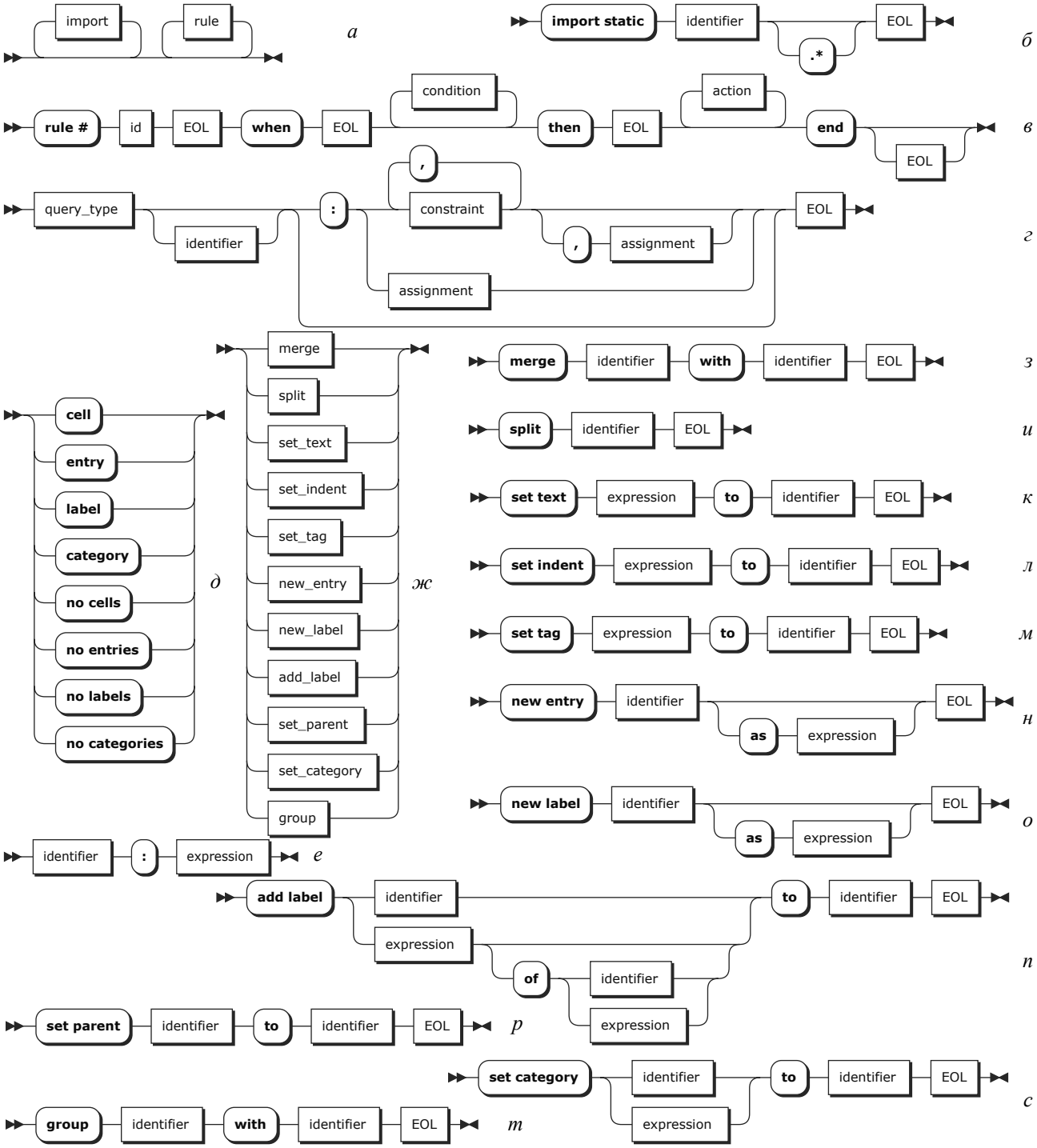


Рисунок 3.7 — Синтаксическая диаграмма CRL: набор правил ($а$); импорт статистических функций и констант ($б$); правило ($в$); условие ($г$) с типом запрашиваемых фактов ($д$) и возможностью присвоить значения переменной ($е$); действие ($ж$), определяемое как объединение ячеек ($з$), разделение ячейки ($и$), изменение текстового содержимого ($к$) или отступа ячейки ($л$), присваивание пользовательского дескриптора ячейке ($м$), создание вхождения ($н$) или метки ($о$), связывание вхождения с меткой ($п$), связывание дочерней и родительской меток ($р$), категоризация метки ($с$) или группирование меток ($т$)

Тип запрашиваемых фактов и вид условия (проверка наличия или отсутствия) определяются ключевым словом (рис. 3.7, *д*). В одном условии все запрашиваемые факты относятся к одному из следующих типов: ячейки, вхождения, метки или категории (раздел 3.1). *Присваивание* устанавливает значение некоторой переменой (рис. 3.7, *е*).

Действия CRL-правил

Действие изменяет существующие или создает новые факты (рис. 3.7, *ж*). Функционально их можно разделить на четыре группы: очистка (`merge`, `split`, `set text`, `set indent`), ролевой анализ (`set tag`, `new entry`, `new label`), структурный анализ (`add label`, `set parent`) и интерпретация (`set category`, `group`). Рассмотрим их подробнее.

Очистка

Данная группа действий позволяет изменить структуру и содержимое ячеек. Действие `merge` (рис. 3.7, *з*) объединяет две ячейки `c1` и `c2`, а `split` (рис. 3.7, *и*) позволяет разделить объединенную ячейку `c`, охватывающую n плиток, на n ячеек:

```
merge c1 with c2
split c
```

Два других редактируют содержимое ячеек: `set text` (рис. 3.7, *к*) задает ячейке `c` новый текст `string_value` (при этом может использоваться обработка строк с помощью импортированных Java-методов):

```
set text string_value to cell
```

`set indent` (рис. 3.7, *л*) устанавливает ячейке `c` новый отступ (количество пробелов) `integer_value`:

```
set indent integer_value to c
```

Ролевой анализ

Действия данной группы порождают вхождения и метки, а также выполняют пользовательскую разметку ячеек. Действие `set tag` (рис. 3.7, *м*) присваивает ячейке `c` пользовательский тег `user_tag`:

```
set tag user_tag to c
```

Два действия `new entry` (рис. 3.7, *н*) и `new label` (рис. 3.7, *о*) генерируют вхождения и метки, соответственно. По умолчанию в качестве значения создаваемого вхождения или метки используется текстовое содержимое ячейки `c` без изменений:

```
new entry c
new label c
```

Опционально в качестве такого значения можно указать строковое выражение `value`, обычно получаемое в результате обработки текстового содержимого ячейки `c`:

```
new entry c as value
new label c as value
```

Структурный анализ

Действия данной группы устанавливают связи между функциональными элементами: `add label` ассоциирует вхождение с меткой (рис. 3.7, *п*); `set parent` сопоставляет дочернюю метку с родительской (рис. 3.7, *р*).

Предлагается три способа создания пары «вхождения–метка». Основной записывается следующим образом:

```
add label l to e
```

Здесь `l` — метка, а `e` — вхождение. Два других связывают вхождение с меткой, искомой по значению `label_value`. В первом случае метка ищется в категории, заданной именем `category_name`:

```
add label label_value of category_name to e
```

Во втором случае она ищется в имеющейся категории `c`:

```
add label label_value of c to e
```

Действие ассоциирования родительской метки `parent_label` с дочерней `child_label` записывается следующим образом:

```
set parent parent_label to child_label
```

Интерпретация

Данная группа направлена на восстановление пар вида «метка–категория». Действие `set category` (рис. 3.7, *c*) ассоциирует метку `l` с категорией `c`:

```
set category c to l
```

Категория может искаться по имени `category_name`:

```
set category category_name to l
```

Действие `group` (рис. 3.7, *m*) группирует две метки `l1` и `l2` в одну категорию:

```
group label1 with label2
```

3.6. Каноникализация данных

Предлагаемая методика каноникализации данных, восстановленных в результате исполнения правил, включает следующие шаги: *шаг 1* — очистка значений меток; *шаг 2* — упрощение деревьев меток; *шаг 3* — формирование канонической формы. Рассмотрим перечисленные шаги подробнее.

3.6.1. Очистка значений меток

Значения меток внутри одной или нескольких таблиц могут различаться по написанию, но означать одну и ту же сущность. Например, следующие метки: `FY2022`, `Year 2022`, `2022 г.` и `Previous Year`, в некотором контексте могут быть синонимами, означающими 2022 год. В качестве их эталона может использоваться значение `2022`. Такие метки заменяются соответствующими эталонами путем сопоставления с внешним словарем, который содержит набор пар вида (S, R), где S — регулярное выражение для идентификации синонимов, а R — соответствующий эталон, заданный также в виде регулярного выражения. Например, если в словаре задать пару вида (`FY202[0-3]`, `202[0-3]`), то значения меток `FY2020`,. . . , `FY2023` будут заменены на соответствующие эталоны `2020`,. . . , `2023`.

3.6.2. Упрощение деревьев меток

Упоминаемое в разделе 3.4.2 разделение меток на группы в процессе исполнения правил не является обязательным. Вместо этого родительско-дочерние метки можно представить в виде деревьев, как описано в разделе 3.4.2. При этом полученные деревья могут упрощаться уже в процессе каноникализации данных (рис. 3.8). Деревья меток могут сопоставляться с описаниями измерений, представленными внешними словарями в форме пар вида (S, d_i), где S — регулярное выражение для идентификации значений измерения, а d_i — соответствующее измерение.

В процессе такого сопоставления из деревьев исключаются метки, которые являются значениями измерений d_i. Исключаемая метка становится частью соответствующей категории d_i, а ее место в дереве переходит поддереву вложенных в нее меток. В том случае когда каждая метка дерева соотнесена с некоторым измерением, само дерево вырождается.

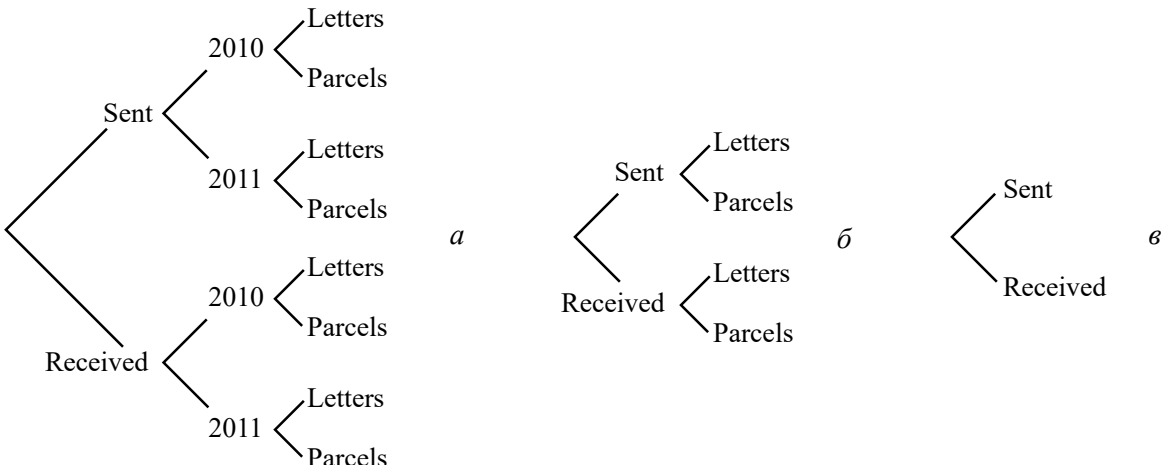


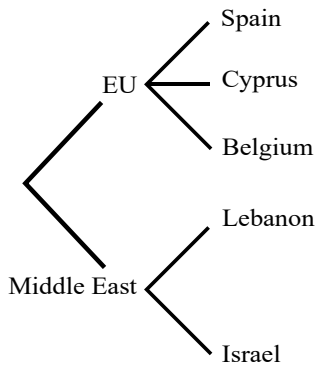
Рисунок 3.8 — Упрощение дерева меток в результате сопоставления с внешними словарями: исходные деревья (*а*); после сопоставления меток `2010` и `2011` с измерением времени (*б*); после сопоставления меток `Letters` и `Parcels` с измерением тип отправления (*в*)

3.6.3. Формирование набора записей в канонической форме

Восстановленные функциональные элементы и связи обеспечивают генерацию набора записей в канонической форме, где столбцы являются полями, первая строка содержит имена полей, а каждая последующая строка является записью. Предлагаемая каноническая форма обязательно включает поле данных, содержащее все восстановленные вхождения. Остальные поля соответствуют восстановленным категориям. Каждое из них содержит метки, принадлежащие соответствующей категории. Каждая запись представляет восстановленные связи между вхождением и метками.

В том случае когда метки некоторой категории составляют дерево, например, как показано на рис. 3.9, *б*, возможны два способа вывода их значений в каноническую форму. Первый выводит пути в дереве меток в виде составных значений в единственное поле (рис. 3.9, *в*), второй выводит значения каждого уровня вложенности в отдельное поле (рис. 3.9, *г*). Сформированная в результате каноническая форма может экспортироваться в реляционную базу данных с помощью стандартных средств.

а



б

Данные	*Операция*	*Год*	*Тип отправления*	*Регион / Страна*
462.9	Sent	2010	Letters	EU / Spain
82.9	Sent	2010	Letters	EU / Cyprus
...	...	...	...	...
12.3	Sent	2010	Parcels	Middle East / Lebanon
469.4	Sent	2011	Letters	EU / Spain
89.7	Sent	2011	Letters	EU / Cyprus
341.1	Sent	2011	Letters	EU / Belgium
21.5	Sent	2011	Letters	Middle East / Lebanon
483.0	Sent	2011	Letters	Middle East / Israel
109.3	Sent	2011	Parcels	EU / Spain
13.1	Sent	2011	Parcels	Middle East / Lebanon
556.3	Received	2010	Letters	EU / Spain
...	...	...	...	...
11.3	Received	2011	Parcels	Middle East / Lebano

в

Данные	*Операция*	*Год*	*Тип отправления*	*Регион*	*Страна*
462.9	Sent	2010	Letters	EU	Spain
82.9	Sent	2010	Letters	EU	Cyprus
...	...	...	...	...	...
12.3	Sent	2010	Parcels	Middle East	Lebanon
469.4	Sent	2011	Letters	EU	Spain
89.7	Sent	2011	Letters	EU	Cyprus
341.1	Sent	2011	Letters	EU	Belgium
21.5	Sent	2011	Letters	Middle East	Lebanon
483.0	Sent	2011	Letters	Middle East	Israel
109.3	Sent	2011	Parcels	EU	Spain
13.1	Sent	2011	Parcels	Middle East	Lebanon
556.3	Received	2010	Letters	EU	Spain
...	...	...	...	...	...
11.3	Received	2011	Parcels	Middle East	Lebanon

г

Рисунок 3.9 — Два варианта формирования набора записей в канонической форме: исходная таблица (*а*); дерево меток категории `Регион/Страна` (*б*); пути дерева меток в единственном поле `Регион/Страна` (*в*); каждому уровню вложенности дерева меток соответствует отдельное поле `Регион` и `Страна` (*г*)

Следует отметить, что в настоящей работе в основном рассматриваются примеры таблиц, представляющих многомерные данные. Однако реляционные сущностные таблицы также могут быть в ней представлены. В случае реляционных таблиц, за ячейки с данными (т. е. «вхождения» канонической формы) следует принимать только те ячейки, которые содержат упоминание сущностей одного класса. Остальные ячейки записей принимаются за «метки» канонической формы, а ячейки с именами атрибутов — за «категории» канонической формы. При этом если строка такой таблицы представляет един-

ственную запись, в результате каноникализации она не будет разбиваться на части.

3.7. Выводы

Предлагаемый метод подходит для реализации решений извлечения реляционных данных из документных таблиц. Один набор CRL-правил может обеспечить анализ и интерпретацию целого класса таблиц с общими свойствами структуры. Сложность реализации CRL-правил зависит от разнообразия и противоречивости используемых таких свойств. Эффективность его применения зависит от качества исходных данных. Некорректные случаи, вероятно, потребуют предобработки с целью улучшения качества самих данных.

Конкурентные решения извлечения данных из таблиц, в том числе [120, 166,295,339,349,350], ограничены заданными свойствами структуры, встроенными непосредственно в их алгоритмы. Все они предполагают, что таблица может быть разделена на несколько функциональных регионов атомарных ячеек, относительное размещение которых внутри таблицы строго задано (раздел 1.6.2). В отличие от аналогов, предлагаемый метод обеспечивает создание решений в виде пользовательских правил анализа и интерпретации таблиц. Специфические ограничения выносятся в правила, обеспечивающие порождение целевых алгоритмов. Программная реализация предлагаемого метода представлена в главе 4, оценка производительности реализованных решений — в главе 5, а их прикладное применение — в главе 6.

Результаты, представленные в данной главе

- Модель таблицы, обеспечивающая представление табличной информации в процессах АПТ. В отличие от известных аналогов, она не огра-

ничивает структуру таблицы предопределенными типами компоновки, атомарностью ячеек и плоскими заголовками.

- Метод автоматизации анализа и интерпретации таблиц редактируемого формата (РК). Впервые данный процесс реализуется посредством пользовательских правил; обеспечена поддержка произвольности структуры таблицы: свободной компоновки, структурированности ячеек и иерархичности заголовков.
- Проблемно-ориентированный язык правил анализа и интерпретации документных таблиц (CRL). В отличие от формальных языков общего назначения, он позволяет сфокусироваться исключительно на реализации логики соответствующих этапов АПТ.

Публикации

Представленные результаты опубликованы в ряде рецензируемых изданий, в том числе журналах [1, 7–11, 50], а также материалах всероссийских и международных мероприятий [24, 29, 31, 39, 42, 44–49, 51, 52, 54, 56].

Глава 4

Реализация программного обеспечения

Глава представляет комплекс инструментальных средств (КИС), являющихся программной реализацией методов, предлагаемых в главах 2 и 3. Описывается архитектура и функциональность разработанного ПО РТ (раздел 4.1). Рассматривается автоматизация процесса трансферного обучения ИНС обнаружения таблиц (раздел 4.2). Излагаются архитектура и функциональность разработанного ПО АИТ (раздел 4.3). Рассматривается применение CRL-правил в различных случаях, которые можно часто встретить на практике (раздел 4.4). Приводятся данные исследования поведения пользователей при применении языка CRL (раздел 4.5). В конце главы делаются выводы (раздел 4.6).

4.1. Инструментальные средства распознавания таблиц

Методики обнаружения и сегментации таблиц, а также предобработки (сегментации страниц) и постобработки (фильтрации кандидатных случаев), предложенные в главе 2, реализованы в виде кодовой базы Java-программ[1,2]. В качестве обоснования возможности разработки конкретных решений РТ в НПОД на основе реализованных методик представлено модельное веб-ориентированное приложение TabbyPDF[3,4] для интерактивного конвертирования таблиц нередактируемого формата PDF в редактируемые форматы (HTML/Excel). Рассмотрим подробнее его архитектуру (раздел 4.1.1), функциональность (раздел 4.1.2) и настройку параметров (раздел 4.1.3).

[1] `https://github.com/tabbydoc/tabbypdf`

[2] `https://github.com/tabbydoc/tabbypdf2`

[3] `https://github.com/tabbydoc/tabbypdf-web`

[4] `https://github.com/tabbydoc/tabbypdf-front`

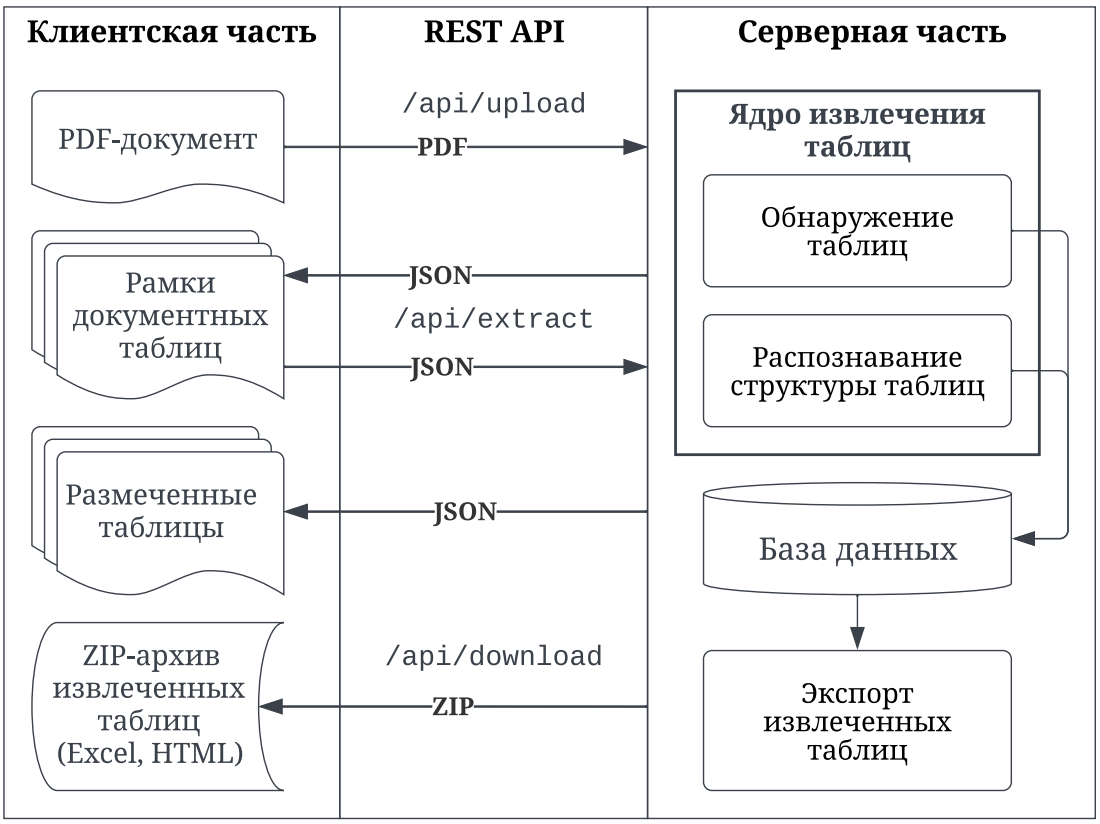


Рисунок 4.1 — Архитектура ПО TabbyPDF

4.1.1. Архитектура программного обеспечения

Модельное веб-ориентированное приложение TabbyPDF имеет клиент-серверную архитектуру, показанную на рис. 4.1. Клиентская часть предоставляет веб-ориентированный пользовательский интерфейс, который реализован с использованием инструментальных средств Semantic UI[5]. Для визуализации PDF-документов в клиентской части используется инструмент Mozilla PDF.js[6]. Серверная часть представляет собой приложение на базе семейства фреймворков Spring[7]. Серверная часть исполняется в составе контейнера Java-сервлетов Apache Tomcat[8]. Кэширование выполняется с помощью резидентной базы данных Apache Derby[9].

[5] `https://semantic-ui.com`

[6] `https://mozilla.github.io/pdf.js`

[7] `https://spring.io`

[8] `http://tomcat.apache.org`

[9] `https://db.apache.org/derby`

Взаимодействие двух частей реализовано посредством веб-ориентированного интерфейса прикладного программирования, реализующего архитектурный стиль взаимодействия компонентов распределенного приложения в сети «REST API». Предоставляется три конечных точки:

- `/api/upload`: клиент отправляет на сервер исходный PDF-документ (POST-запрос); ядро выполняет обнаружение таблиц в данном документе; сервер возвращает клиенту ограничивающие рамки таблиц (JSON);
- `/api/extract`: клиент отправляет на сервер подтвержденные пользователем координаты ограничивающих рамок таблиц; ядро выполняет распознавание структуры данных таблиц; сервер возвращает клиенту размеченные таблицы (JSON);
- `/api/download/:id`: клиент отправляет на сервер идентификатор обработанного документа; сервер возвращает клиенту zip-архив таблиц, извлеченных из данного документа (в формате Excel).

4.1.2. Функциональность программного обеспечения

Сценарий использования предполагает следующий порядок действий. Пользователь выбирает PDF-документ и запускает его обработку. Приложение обнаруживает рамки таблиц. Они рисуются поверх страниц PDF-документа (рис. 4.2). При необходимости пользователь может корректировать, удалять и добавлять рамки таблиц. Затем он запускает их обработку. Приложение распознает структуру данных таблиц. Пользователь изучает полученные результаты — размеченные таблицы (рис. 4.3). Он может скачать архив извлеченных таблиц в формате Excel. Само функциональное ядро PT также может использоваться в пакетном режиме без участия пользователя через командную строку.

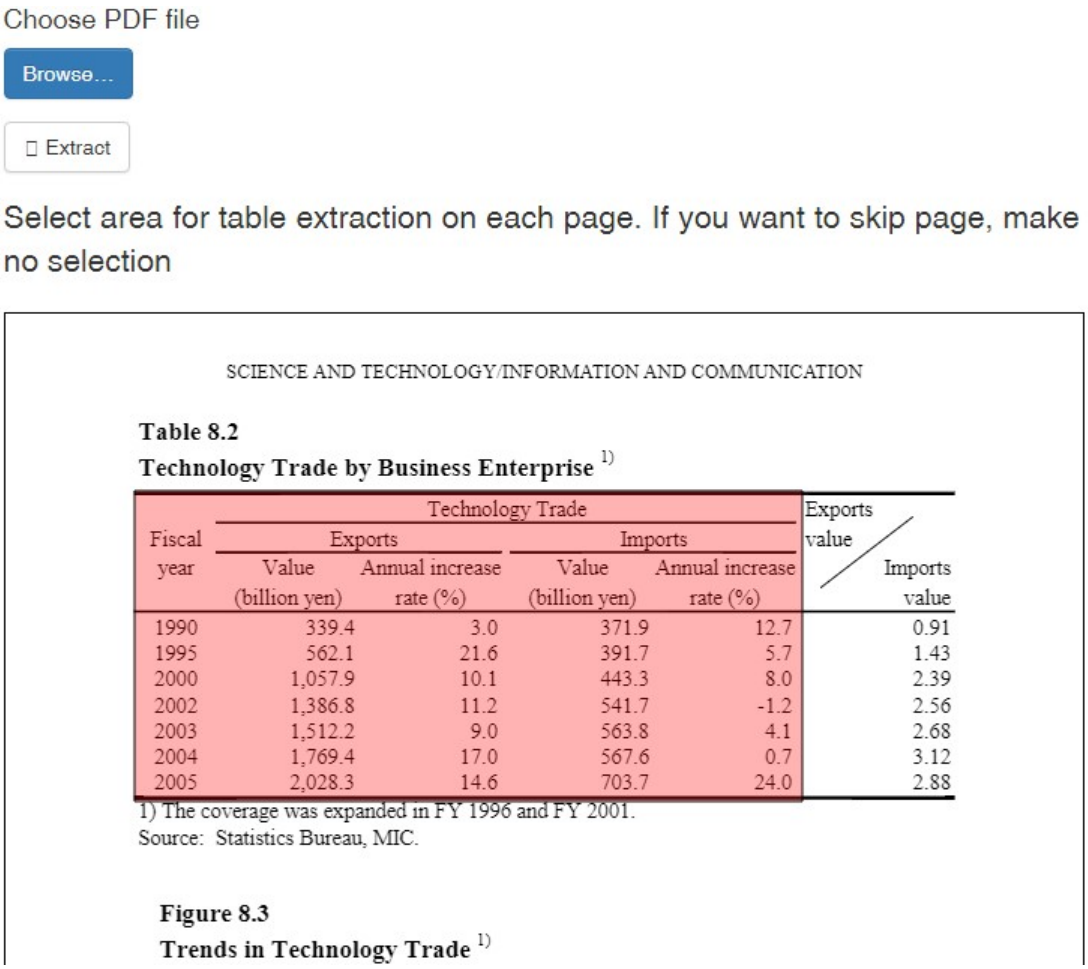


Рисунок 4.2 — Экран пользовательского интерфейса TabbyPDF: автоматически обнаруженная таблица

4.1.3. Настройка параметров

Обеспечивается возможность настройки некоторых параметров, управляющих реализованными алгоритмами распознавания таблиц. В частности, качество результатов можно улучшить, установив подходящие пороговые значения и коэффициенты. Последние могут быть найдены в результате поиска максимального значения оценки производительности (например, F-меры — гармонического среднего между точностью и полнотой) на некоторой выборке предметных данных. Рисунок 4.4 демонстрирует поиск максимального значения F-меры как функции двух переменных: коэффициентов межсловного пробела — k_w и межстрочного интервала — k_h, используемых в процессе

	Technology Trade			
Fiscal	Exports		Imports	
year	Value	Annual increase	Value	Annual increase
	(billion yen)	rate (%)	(billion yen)	rate (%)
1990	339.4	3.0	371.9	12.7
1995	562.1	21.6	391.7	5.7
2000	1,057.9	10.1	443.3	8.0
2002	1,386.8	11.2	541.7	-1.2
2003	1,512.2	9.0	563.8	4.1
2004	1,769.4	17.0	567.6	0.7
2005	2,028.3	14.6	703.7	24.0

Рисунок 4.3 — Экран пользовательского интерфейса TabbyPDF: автоматически распознанная таблица

объединения текстовых компонентов в блоки. Всего на обучающей выборке набора данных ICDAR-2013 оценено 2500 тестов, в которых значения k_w и k_h увеличились от 0 до 5 с шагом 0,1; максимальное значение F-меры было достигнуто при k_w равном 0,9, а k_h — 1,0.

4.2. Настройка нейросетевых моделей обнаружения таблиц

Рассмотрим реализованное в диссертации инструментальное средство DL4TD[10] для управления процессом трансферного обучения ИНС обнаружения таблиц на платформе TensorFlow. Исследовалась применимость ИНС обнаружения объектов одной из наиболее популярных архитектур, а именно «Faster R-CNN». Данная архитектура состоит из следующих компонентов:

[10] `https://github.com/tabbydoc/dl4td`

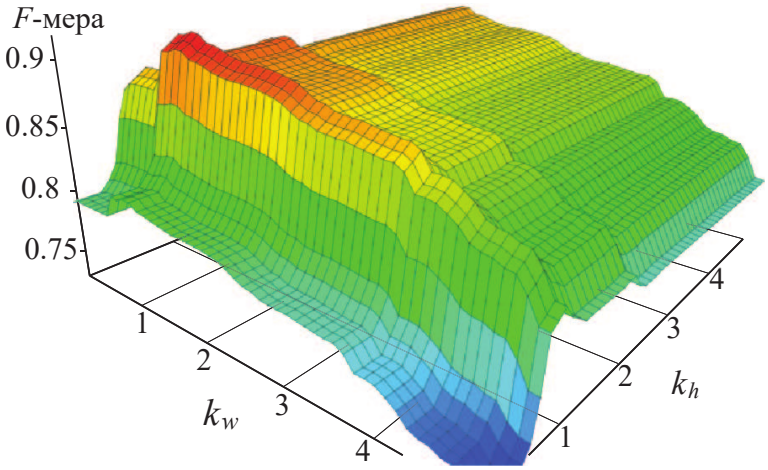


Рисунок 4.4 — Настройка параметров распознавания таблиц: поиск значений коэффициентов межсловного пробела k_w и межстрочного интервала k_h, дающих максимальное значение F-меры на обучающей выборке набора данных ICDAR-2013

CNN (базовой сети) — сверточной ИНС извлечения признаков, предобученной на большом массиве изображений; RPN — ИНС прогнозирования регионов (ограничивающих рамок объектов любых классов); RoI-pooling — слоя сопоставления координат регионов с картой признаков; R-CNN — полносвязных слоев классификации и уточнения координат гипотез. Каждый из перечисленных компонентов данной архитектуры соответствует отдельному шагу процесса обнаружения объектов, как показано на рис. 4.5.

Данная архитектура позволяет заранее выполнить так называемую *тонкую настройку* весов полносвязных слоев, отвечающих за прогнозирование регионов (RPN), классификацию обнаруженных объектов и уточнение координат их ограничивающих рамок (R-CNN). В рассматриваемом случае такая настройка должна позволить ИНС относить объекты, обнаруживаемые в изображениях документов, к классу таблиц. Исследование [187, 354] показало, что сама настройка может выполняться на небольших наборах предметных данных: UW3/UNLV [359], Marmot [173], ICDAR-2013 [189], ICDAR-2017 POD [181], ICDAR-2019 [182] и др., включающих от нескольких сотен до ты-

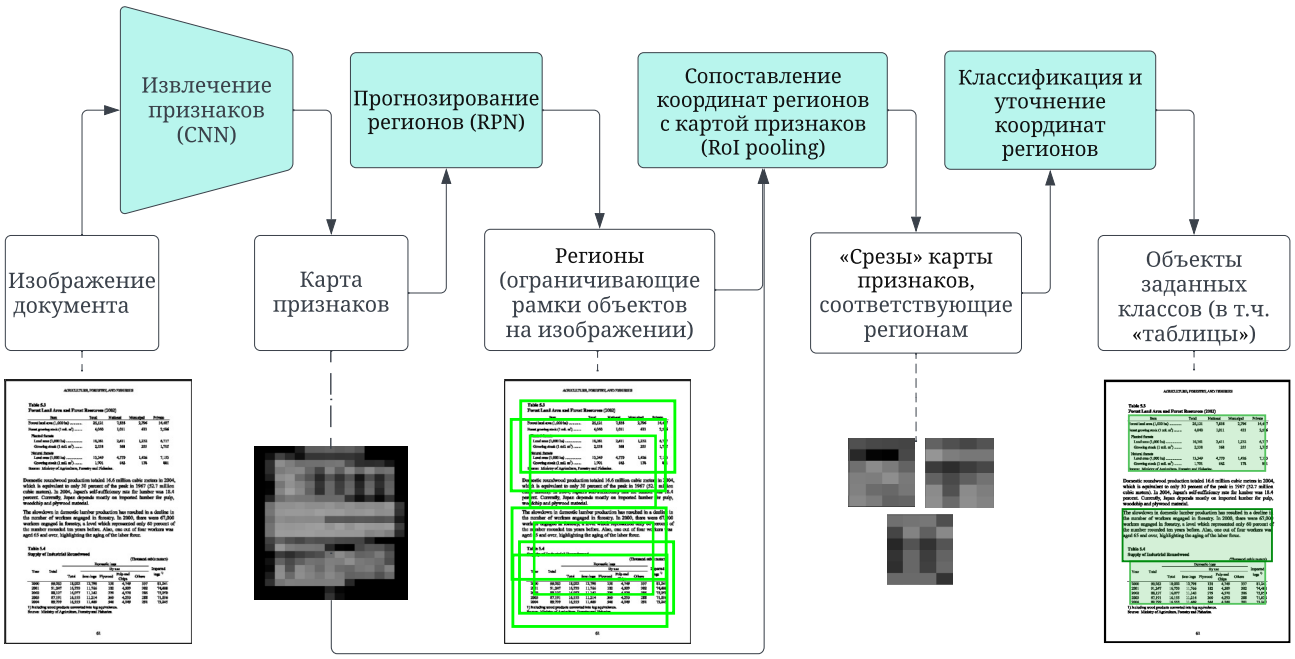


Рисунок 4.5 — Процесс обнаружения таблиц в изображении документа с помощью ИНС архитектуры «Faster R-CNN»

сяч примеров изображений страниц документов с сопроводительной разметкой представленных в них таблиц.

Предлагается рабочий процесс тонкой настройки ИНС данной архитектуры (рис. 4.6), включающий следующие шаги: *шаг 1* — доступные наборы данных в нативных форматах приводятся к единому формату — Pascal VOC; *шаг 2* — изображения страниц документов размываются с помощью преобразования расстояний; *шаг 3* — набор данных дополняется синтетическими примерами, полученными путем аффинных преобразований исходных изображений; *шаг 4* — генерация обучающей и тестовой выборок в формате TensorFlow Records; *шаг 5* — обучение модели ИНС на платформе TensorFlow[11].

Очевидно, что представленный рабочий процесс вовлекает большое количество действий оператора от подготовки данных до запуска обучения и тестирования целевой ИНС. Для того чтобы сократить усилия оператора, в

[11] `https://www.tensorflow.org`

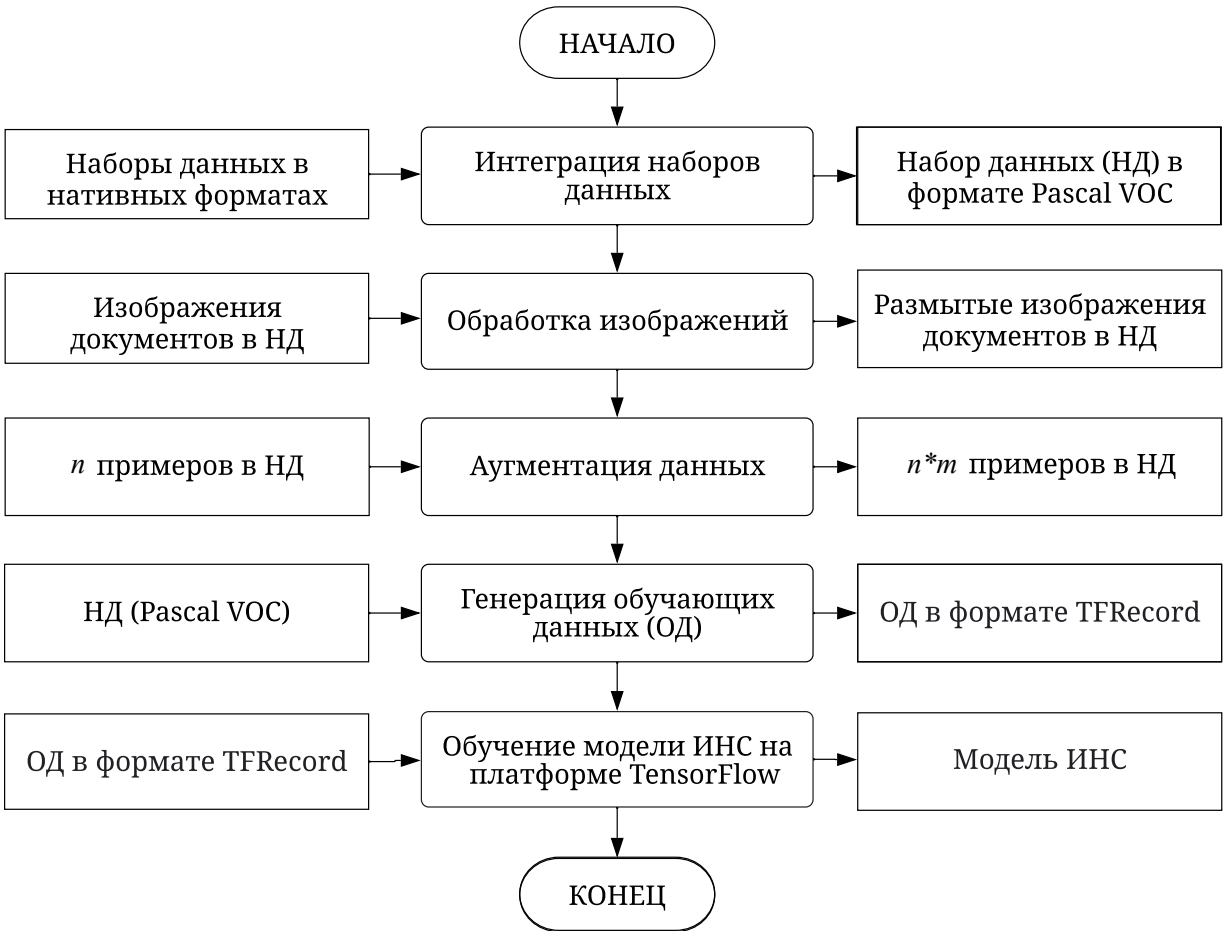


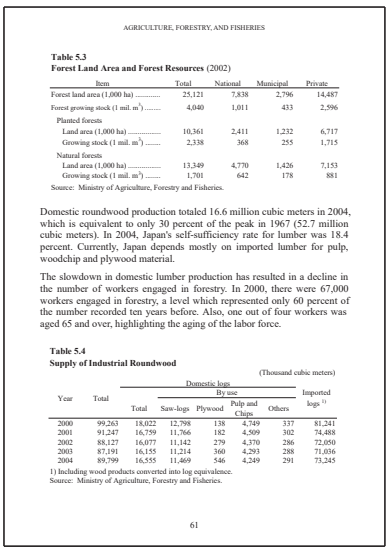
Рисунок 4.6 — Рабочий процесс обучения ИНС моделей обнаружения таблиц; НД — наборы данных

рамках диссертационной работы был разработан вычислительный конвейер, автоматизирующий представленный рабочий процесс. Все его шаги реализованы с помощью скриптов на языке программирования Python[12] и библиотеки алгоритмов обработки изображений OpenCV[13].

В качестве базовой сети извлечения признаков (CNN) используется ИНС ResNet-101 [204], обученная на большом массиве размеченных фотоизображений ImageNet [147]. Два полносвязных слоя R-CNN — softmax-классификации объектов и регрессии координат регионов обучаются на наборе данных, скомбинированном из доступных коллекций изображений документов с раз-

[12] `https://www.python.org`

[13] `https://opencv.org`



AGRICULTURE, FORESTRY, AND FISHERIES

Table 5.3
Forest Land Area and Forest Resources (2002)

Item	Total	National	Municipal	Private
Forest land area (1,000 ha)	25,121	7,838	2,796	14,487
Forest growing stock (1 mil. m^3)	4,040	1,011	433	2,596
Planted forests				
Land area (1,000 ha)	10,361	2,411	1,232	6,717
Growing stock (1 mil. m^3)	2,338	368	255	1,715
Natural forests				
Land area (1,000 ha)	13,349	4,770	1,426	7,153
Growing stock (1 mil. m^3)	1,701	642	178	881

Source: Ministry of Agriculture, Forestry and Fisheries.

Domestic roundwood production totaled 16.6 million cubic meters in 2004, which is equivalent to only 30 percent of the peak in 1967 (52.7 million cubic meters). In 2004, Japan's self-sufficiency rate for lumber was 18.4 percent. Currently, Japan depends mostly on imported lumber for pulp, woodchip and plywood material.

The slowdown in domestic lumber production has resulted in a decline in the number of workers engaged in forestry. In 2000, there were 67,000 workers engaged in forestry, a level which represented only 60 percent of the number recorded ten years before. Also, one out of four workers was aged 65 and over, highlighting the aging of the labor force.

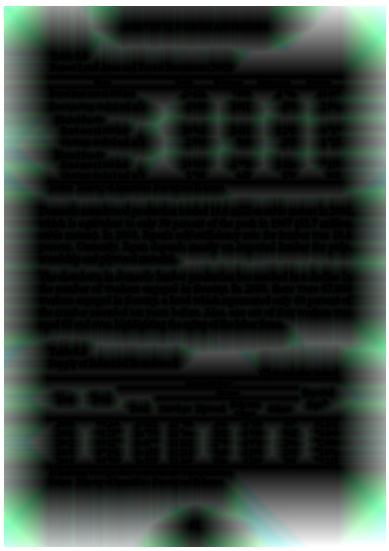
Table 5.4
Supply of Industrial Roundwood

(Thousand cubic meters)

Year	Total	Domestic logs					Imported logs 1)
		Total	By use				
			Saw-logs	Plywood	Pulp and Chips	Others	
2000	99,263	18,022	12,798	138	4,749	337	81,241
2001	91,247	16,759	11,766	182	4,509	302	74,488
2002	88,127	16,077	11,142	279	4,370	286	72,050
2003	87,191	16,155	11,214	360	4,293	288	71,036
2004	89,799	16,555	11,469	546	4,249	291	73,245

1) Including wood products converted into log equivalence.
Source: Ministry of Agriculture, Forestry and Fisheries.

61



а *б*

Рисунок 4.7 — Трансформация изображения документа алгоритмами преобразования расстояния: исходное ($а$) и целевое ($б$) изображение

меченными регионами таблиц. Для увеличения обучающей выборки используется аугментация данных на основе аффинных преобразований.

Настройка целевой ИНС реализована на комбинации из трех исходных наборов данных: UNLV [359], Marmot [173] и ICDAR-2017 POD [181]. Разметка обеспечивает единственный класс распознаваемых объектов — таблиц. Всего задействовано 2800 примеров (страниц документов). Аугментация данных позволяет увеличить размер обучающей выборки до 16000 примеров.

Следует отметить, что доступные базовые сети извлечения признаков (CNN), включая сети семейств AlexNet [258], VGG [366] и ResNet [204], обучены на основе больших массивов размеченных фотоизображений, таких как ImageNet [147], Pascal VOC [170], MS COCO [274]. Изображения документов имеют иную природу: объекты их компоновки состоят из множества напечатанных символов текста с четко различимой на пробельном фоне границей. Как указывает исследование [187], для того чтобы сделать изображения документов более пригодными для извлечения признаков доступными базовыми сетями, можно использовать алгоритмы *преобразования расстояния* (*distance*

transform) [101,171], позволяющие выполнить размытие растра, как показано на рис. 4.7. В диссертации предлагается выполнять трансформацию исходного изображения I, предлагаемую в работе [187], при которой каналы целевого изображения — P (красный r, зеленый g и синий b) формируются разными алгоритмами преобразования расстояния (максимального, линейного и евклидова).

4.3. Инструментальные средства анализа и интерпретации таблиц

В качестве обоснования реализуемости теоретических основ АИТ, представленных в главе 3, создана инструментальная платформа TabbyXL[14] для разработки прикладного ПО извлечения данных из таблиц РК на основе пользовательского программирования правил. Рассмотрим подробнее ее архитектуру (раздел 4.3.1) и функциональность, а именно исполнение правил (раздел 4.3.2) и генерацию исполняемых Java-приложений (раздел 4.3.3), а также дополнительные инструментальные средства: утилиту автоматической коррекции физической структуры ячеек заголовка, обеспечивающую предобработку исходных таблиц в TabbyXL (раздел 4.3.4) и веб-ориентированное приложение интерактивной демонстрации провенанса данных, извлекаемых из таблиц РК посредством TabbyXL (раздел 4.3.5).

4.3.1. Архитектура программного обеспечения

Разработанная инструментальная платформа TabbyXL имеет монолитную архитектуру, как показано на рис. 4.8. Определено взаимодействие следующих основных компонентов:

[14] `https://github.com/tabbydoc/tabbyxl`

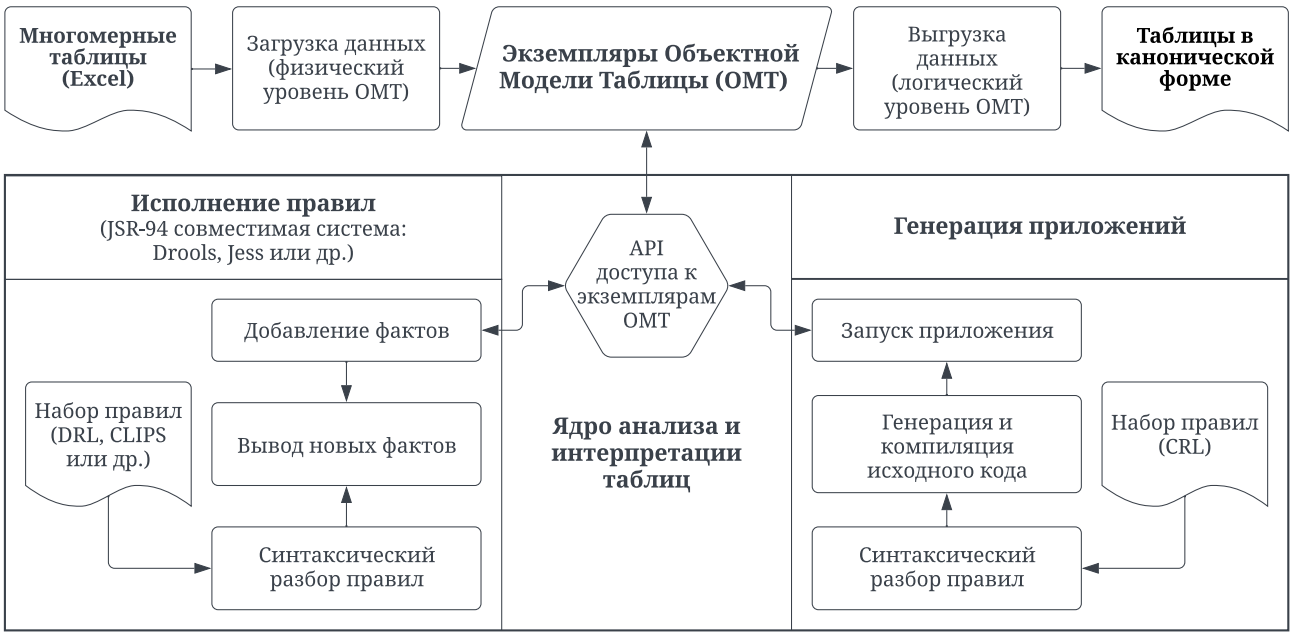


Рисунок 4.8 — Архитектура ПО TabbyXL

- *Объектная модель таблицы* (далее OMT) реализует двухуровненное представление структуры документных таблиц (раздел 3.1). Загрузчик данных отображает ячейки конкретной таблицы в физический уровень экземпляра OMT. Предоставляется интерфейс прикладного программирования доступа к загруженному экземпляру OMT.
- *Ядро анализа и интерпретации таблиц* (АИТ) восстанавливает логический уровень экземпляра OMT, используя одну из двух опций: *исполнение правил* (раздел 4.3.2) или *генерация приложений* (раздел 4.3.3). В результате обращения к любой из двух опций дополненный экземпляр OMT отображается в каноническую форму.

Следует отметить, конечное исполняемое приложение создается на основе одного набора пользовательских правил анализа и интерпретации таблиц. Выполнение последних трансформирует документные таблицы РК форматов Excel/Sheets в наборы записей канонической формы.

4.3.2. Исполнение правил

Исполнение правил базируется на системах вывода, совместимых со стандартом JSR-94[15]. Правила должны быть представлены на соответствующем формальном языке системы вывода. Исходные факты из экземпляра OMT помещаются в рабочую память системы исполнения, где они сопоставляются с предоставленными правилами. Выводимые факты соответствуют логическому уровню экземпляра OMT.

По умолчанию поддерживается две системы вывода: Drools[16] и Jess[17]. Ожидается, что исполняемые правила выражены в нативных форматах этих систем: DRL и CLIPS, соответственно. Может также использоваться диалект языка CRL, поддерживающий DRL-атрибуты в объявлениях правил. Такие CRL-правила автоматически транслируются в формат DRL посредством спецификации отображений (приложение Г). Выбор системы вывода настраивается в конфигурационном файле.

Следует отметить, что использование любой JSR-94 совместимой системы исполнения правил предполагает, что факты, размещаемые в рабочей памяти, являются экземплярами Java-классов, соответствующими JavaBeans[18] спецификации. OMT реализована именно такими классами: ячейки, вхождения, метки и категории являются их экземплярами.

4.3.3. Генерация исполняемых приложений

Генерация исполняемых приложений состоит в трансляции CRL-правил в исходный код на языке программирования Java, готовый к компиляции в исполняемое приложение. Рабочий процесс трансляции CRL-правил

[15] `https://www.jcp.org/ja/jsr`

[16] `https://www.drools.org`

[17] `https://www.jessrules.com`

[18] `https://www.oracle.com/java/technologies/javase/javabeans-spec.html`

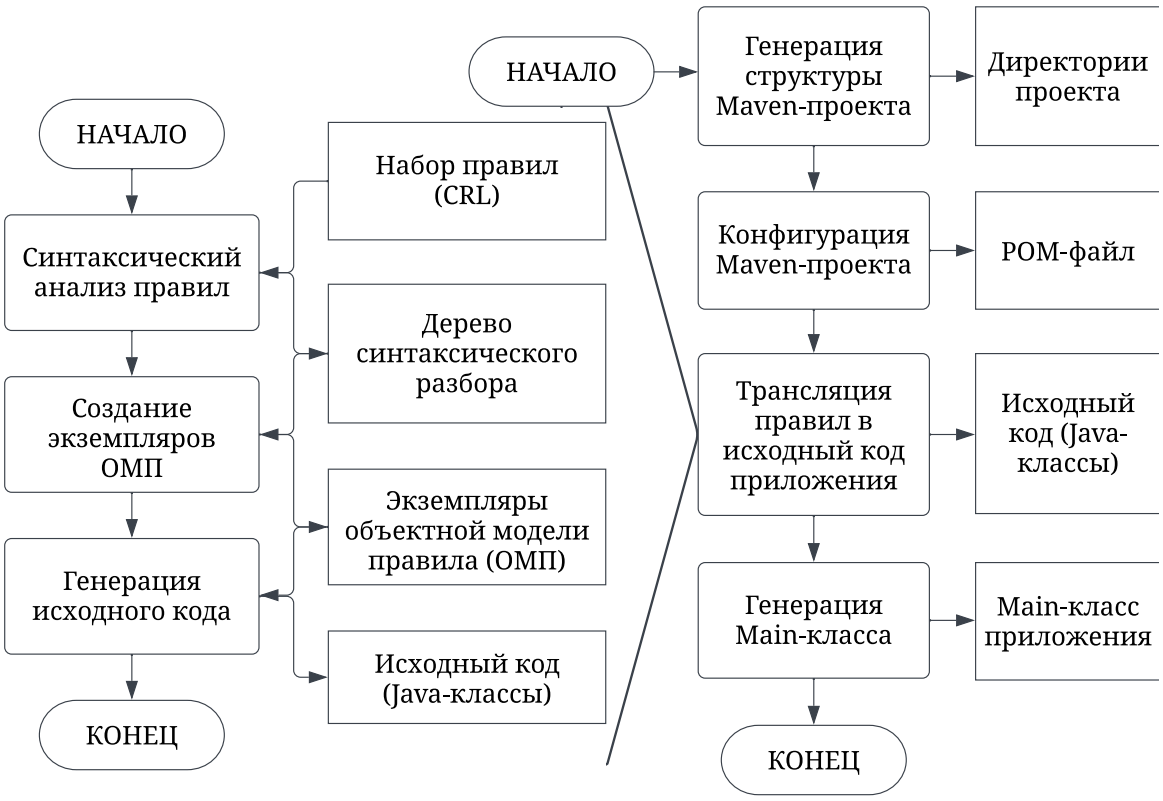


Рисунок 4.9 — Рабочий процесс трансляции CRL-правил в исходный код целевого приложения извлечения данных из таблиц РК ($а$); рабочий процесс генерации Maven-проекта целевого приложения извлечения данных из таблиц РК ($б$)

(рис. 4.9, $а$) состоит из следующих шагов: синтаксический анализ правил; создание экземпляра объектной модели каждого правила; генерация исходного кода целевого приложения. Перечисленные шаги реализованы следующим образом. CRL-парсер синтезирован по грамматике CRL (приложение Г) с помощью инструментальных средств ANTLR[19]. Объектная модель правила (рис. 4.10) представлена в виде Java-классов. Генерация исходного кода базируется на инструментальном средстве JavaPoet[20]. Каждое правило транслируется в отдельный Java-класс, реализующий общий функциональный интер-

[19] `https://www.antlr.org`

[20] `https://github.com/square/javapoet`

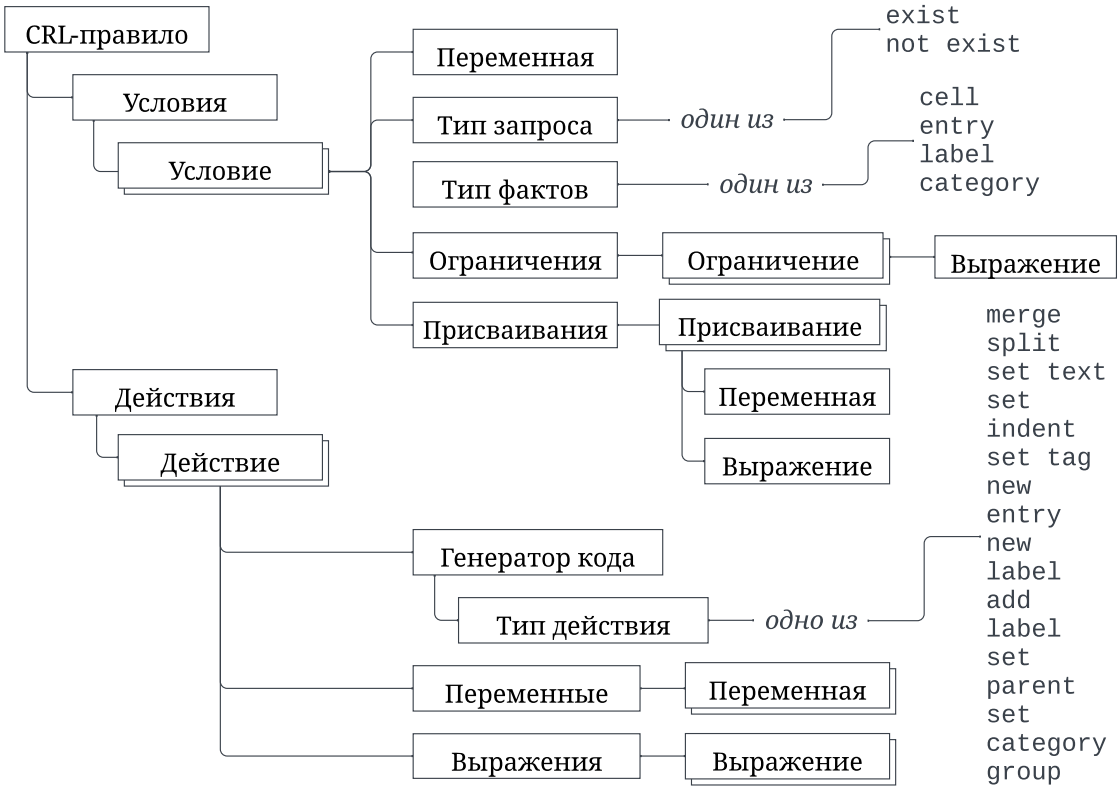


Рисунок 4.10 — Объектная модель CRL-правила

фейс — `Consumer<T>`[21] (процедура с единственным аргументом). В результате каждый класс реализует процедуру (`accept(T t)`), которая выполняет операцию, соответствующую исходному правилу над данной таблицей.

Рассмотрим пример трансляции правила, показанного ниже:

```
when
  entry e
  label l: l.cell.cr == e.cell.cr
then
  add label l to e
```

Его левая часть включает два условия, а правая — одно-единственное действие. Фрагмент исходного кода, сгенерированного из этого правила, приводится ниже:

[21] `https://docs.oracle.com/en/java/javase/14/docs/api/java.base/java/util/function/Consumer.html`

```
public void accept(CTable table) {
  Iterator<CEntry> eIterator = table.getEntries();
  while (eIterator.hasNext()) {
    CEntry e = eIterator.next();
    if (true) {
      Iterator<CLabel> lIterator = table.getLabels();
      while (lIterator.hasNext()) {
        CLabel l = lIterator.next();
        if (l.getCell().getCr() == e.getCell().getCr())
          e.addLabel(l);
      }
    }
  }
}
```

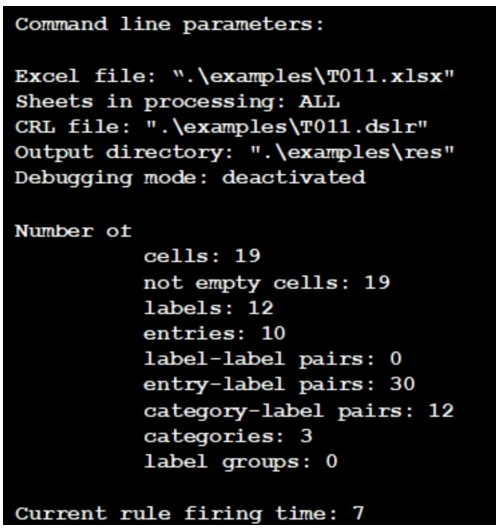
Для того чтобы собрать исполняемое Java-приложение извлечения данных из таблиц РК, генерируется проект в формате Maven[22]. Рабочий процесс его генерации (рис. 4.9, *б*) включает следующие шаги: создание иерархии директорий проекта в соответствии с соглашениями Maven; генерацию конфигурационного файла проекта в формате POM[23]; сериализацию файловых объектов Java (исходного кода); генерацию Main-класса исполняемого приложения. Структура проекта, сгенерированного из n правил, соответствует следующей форме:

```
+-- src/
¦   L-- main/
¦       L-- java/
¦           L-- generated/
¦           ¦   +-- TableConsumer1.java <generated from rule_1>
¦           ¦   +-- ...
¦           ¦   L-- TableConsumerN.java <generated from rule_n>
¦           L-- SpreadsheetTableCanonicalizer.java <Main-class>
L-- pom.xml <POM file>
```

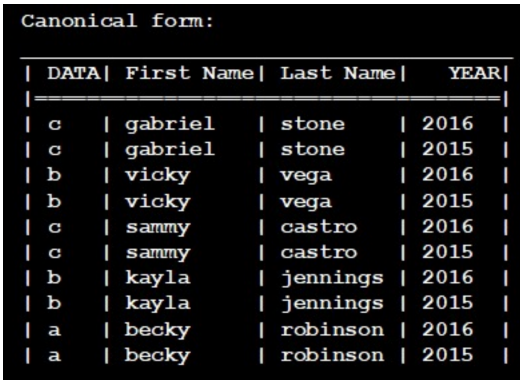
Полученный проект полностью готов для автоматической сборки. Запуск полученного приложения позволяет привести таблицы, удовлетворяющие ис-

[22] https://maven.apache.org

[23] https://maven.apache.org/pom.html



а



б

Рисунок 4.11 — Экраны пользовательского интерфейса целевого приложения, сгенерированного с помощью TabbyXL: начальные параметры (*а*) и результаты обработки данных — набор записей в канонической форме (*б*)

ходному набору правил, к канонической форме (рис. 4.11). Более того, имея сгенерированный Maven-проект и исходный код Java, конечные пользователи получают возможность его доработки в соответствии со своими требованиями к извлечению данных из таблиц РК.

4.3.4. Утилита автоматической коррекции физической структуры ячеек заголовка

В составе инструментальной платформы TabbyXL разработана утилита (HeadRecog), предназначенная для автоматического сопоставления физической структуры ячеек заголовка с его визуальной структурой и внесение необходимых корректировок в тех случая, когда они не совпадают. Применение данной утилиты в процессе предобработки исходных таблиц позволяет улучшить качество результатов последующего АИТ. Данная утилита разработана специально для экспериментальной оценки качества результатов АИТ на тестовом наборе SAUS200. Подробно ее применение рассматривается в разделе 5.4.3.

4.3.5. Веб-ориентированное приложение интерактивной демонстрации провенанса извлеченных наборов записей

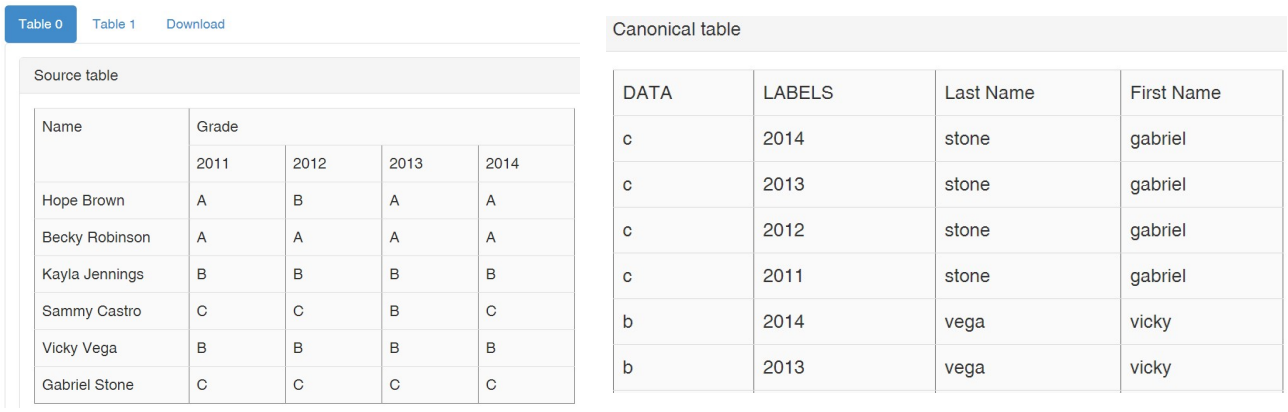
Разработано веб-ориентированное приложение (WebSSDC) для интерактивной демонстрации провенанса наборов записей, извлеченных из произвольных таблиц РК посредством инструментальной платформы TabbyXL (рис. 4.12). Обеспечивается следующая функциональность: каноникализация данных таблиц РК с помощью CRL-правил; генерация извлеченных наборов записей в формате JSON; интерактивная демонстрация провенанса в виде связей между исходными таблицами и извлеченными наборами записей. Оно имеет клиент-серверную архитектуру.

Клиентская часть служит для передачи на сервер исходных РК в формате XLSX, наборов CRL/DRL-правил и спецификаций категорий в формате YAML, а также для отображения результатов извлеченных наборов записей. Клиент предоставляет веб-ориентированный пользовательский интерфейс для загрузки исходных данных и демонстрации результатов с поддержкой интерактивного выделения связанных друг с другом исходных ячеек и целевых значений записей для отслеживания происхождения последних.

Серверная часть выполняет процесс извлечения наборов записей из таблиц средствами инструментальной платформы TabbyXL, используя загруженные РК, наборы правил АИТ и спецификаций категорий. Сервер возвращает клиенту результаты, а именно наборы записей и их провенанс, в формате JSON. Опционально набор записей экспортируются в форматы XLSX/CSV.

4.4. Примеры разработки CRL-правил

В данном разделе рассматривается применение CRL-правил в различных случаях, которые можно часто встретить на практике.



а *б*

Рисунок 4.12 — Экраны пользовательского интерфейса веб-ориентированного приложения интерактивного доступа к провенансу извлеченных наборов записей: исходная таблица (*а*); целевой набор записей в канонической форме (*б*)

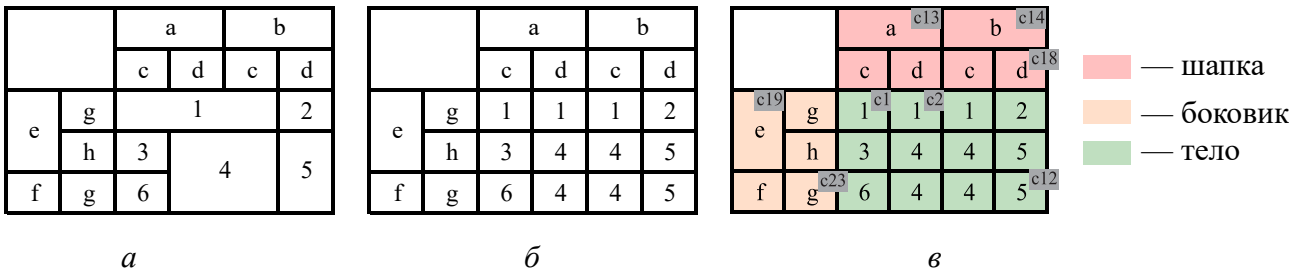
4.4.1. Критические ячейки

Рассмотрим пример функциональной сегментации таблиц посредством, так называемых, *критических ячеек* [300], определяющих границы функциональных регионов ячеек. Пусть имеется таблица, в которой критическая ячейка, расположенная в верхнем левом углу, разделяет между собой шапку, боковик и тело. Пусть также тело содержит объединенные ячейки (рис. 4.13, *а*). Тогда следующее правило разделит их на плитки (рис. 4.13, *б*):

```
when
cell corner: cl == 1, rt == 1
cell c: cl > corner.cr, rt > corner.rb
then
set tag "body" to c
split c
```

Кроме того, каждой ячейке, выбранной через условие левой части, будет присвоен пользовательский тег `"body"` для обеспечения возможности обращения к ней в последующих правилах.

Для того чтобы создать вхождение в каждой ячейке тела, можно воспользоваться следующим правилом:



а *б* *в*

Рисунок 4.13 — Сценарии применения CRL-правил: исходная таблица (*а*); она же после разделения объединенных ячеек в теле (*б*); она же после функциональной сегментации (*в*)

```
when
cell c: tag == "body"
then
new entry c
```

При обработке подобных таблиц (рис. 4.13, *б*) метки можно разделить на строчные и столбцевые в зависимости от того, как они адресуют вхождения: по строкам или столбцам, соответственно. Одно правило выполняет порождение столбцевых меток с присваиванием ячейкам пользовательского тега `head`:

```
when
cell corner: cl == 1, rt == 1
cell c: cl > corner.cr, rb <= corner.rb
then
set tag "head" to c
new label c
```

Другое правило аналогично выполняет порождение строчных меток с присваиванием ячейкам пользовательского тега `stub`:

```
when
cell corner: cl == 1, rt == 1, blank
cell c: cr <= corner.cr, rt > corner.rb
then
set tag "stub" to c
```

В результате исполнения представленных правил будет выполнена функциональная сегментация таблицы, как показано на рис. 4.13, *в*. Некоторые из помеченных ячеек представлены ниже:

```
c1=(cl=3, rt=3, cr=3, rb=3, value="1", tag="body")
c12=(cl=6, rt=5, cr=6, rb=5, value="5", tag="body")
c13=(cl=3, rt=1, cr=4, rb=1, value="a", tag="head")
c18=(cl=6, rt=2, cr=6, rb=2, value="d", tag="head")
c19=(cl=1, rt=3, cr=1, rb=4, value="e", tag="stub")
c23=(cl=2, rt=5, cr=2, rb=5, value="g", tag="stub")
```

При этом в ячейках `c1`,..., `c12` будут порождены вхождения, в `c13`,..., `c18` — столбцевые метки, а в `c19`,..., `c23` — строчные метки.

4.4.2. Группы меток

Пусть все столбцевые метки, порожденные из ячеек одной строки, принадлежат одной категории и каждая строка соответствует отдельной категории (рис. 4.13, *в*). Тогда столбцевые метки можно сгруппировать в анонимные категории следующим образом:

```
when
cell c1: tag == "head"
cell c2: tag == "head", rt == c1.rt
then
group c1.label with c2.label
```

Пусть аналогично строчные метки, происходящие из ячеек одного столбца, ассоциированы с одной категорией. Тогда их можно сгруппировать в анонимные категории следующим образом:

```
when
cell c1: tag == "stub"
cell c2: tag == "stub", cl == c1.cl
then
group c1.label with c2.label
```

В результате применения приведенных правил к таблице (рис. 4.13, *в*) метки будут сгруппированы в четыре категории следующим образом: {a, b}, {c, d}, {e, f} и {g, h}.

4.4.3. Заданные категории

Значения некоторой категории могут задаваться в виде наборов естественно-языковых и регулярных выражений в формате YAML[24]. Например, категория `YEAR`, ограниченная диапазоном значений от 1995 до 2015, может быть описана следующим образом:

```
# category YEAR
name: YEAR
constraints:
- "199[5-9]"
- "200[0-9]"
- "201[0-5]"
```

Метки (1995,. . . , 2015) могут быть ассоциированы с данной категорией следующим образом:

```
when
category c: name == "YEAR"
label l: c.match(value)
then
set category c to l
```

4.4.4. Упоминание категорий внутри таблицы

Таблица может содержать упоминания категорий. В представленном случае (рис. 4.14, *а*) угловик содержит упоминание двух категорий: первая адресует столбцевые, а вторая — строчные метки. Одно правило создает категорию из первого токена текста угловика и ассоциирует с ней столбцевые метки:

[24] `https://yaml.org`

B \ A	a1	a2	a3
b1	1	2	3
b2	4	5	6

а

	a	b
c		
··c1		
····c11	1	2
····c12	3	4
··c2		
····c21	5	6
d		
··d1		
····d11	7	8

б

	a		b	
	c	d	c	d
e				
g	1	2	3	4
h	5	6	7	8
f				
i	9	10	11	12
j	13	14	15	16

в

Рисунок 4.14 — Сценарии применения CRL-правил: угловик содержит упоминания двух категорий — `A` и `B`: первой принадлежат столбцевые метки (`a1`, `a2`, `a3`), а второй — строчные метки (`b1`, `b2`) (*а*); иерархия строчных меток (`c`,..., `c21`, `d`,..., `d11`) представлена визуально с помощью отступов (··) (*б*); перерезы `e` и `f` разделяют таблицу по строкам на несколько частей (*в*)

```
when
cell corner: cl == 1, rt == 1
label l: cell.cl > corner.cr
then
set category token(corner, 0) to l
```

Другое правило аналогично создает категорию из второго токена текста угловика и ассоциирует с ней строчные метки:

```
when
cell corner: cl == 1, rt == 1
label l: cell.rt > corner.rb
then
set category token(corner, 1) to l
```

4.4.5. Иерархии строчных меток

Одной из распространенных практик является использование в тексте таблиц отступов от левого края, для того чтобы показать существующие родительско-дочерние связи между строчными метками (рис. 4.14, *б*). Следующее

правило, используя предположение о том, каждый уровень вложенности увеличивает отступ на два пробела, восстанавливает иерархию строчных меток в первом столбце:

```
when
cell c1: cl == 1
cell c2: cl == 1, rt > c1.rt, indent == c1.indent + 2
no cells: cl == 1, rt > c1.rt, rt < c2.rt, indent <= c1.indent
then
set parent label c1.label to c2.label
```

4.4.6. Перерезы

Таблица может быть разделена на части по строкам так называемыми *перерезами* (рис. 4.14, *в*). Предполагается, что такой перерез содержит метку, адресующую вхождения, размещенные в строках той части, которую он озаглавливает. Как показано на рис. 4.14, *в*, один перерез адресует вхождения `1`,..., `8`, а другой — `9`,..., `16`. Следующее правило помечает такие перерезы:

```
when
cell c: cl == 1, cr == owner.numOfCols
then
set tag "cut-in" to c
```

4.4.7. Структурированные ячейки

Одна ячейка может содержать несколько значений одновременно. В первом случае таблица (рис. 4.15, *а*) дублирует метки на двух языках. Предполагая, что каждая ячейка, размещенная в первой строке или первом столбце, включает два токена, первый из которых соответствует метке на одном языке, а второй — метке на другом, можно записать следующее правило генерации меток:

```
when
cell c: cl == 1 || rt == 1, !blank
```

	α **а**	β **б**
γ **в**	1	2
δ **г**	3	4

а

C1	C2
a = 1	b = 2
c = 3	d = 4
e = 7	f = 8
g = 10	h = 11

б

Рисунок 4.15 — Сценарии применения CRL-правил: структурированные ячейки содержат двуязычные метки (*а*); каждая ячейка данных содержит одновременно метку и связанное вхождение в виде пары ключ–значение (*б*)

```
then
new label token(c, 0) to c
new label token(c, 1) to c
```

Для того чтобы связать вхождение с обеими метками (рис. 4.15, *а*), можно воспользоваться следующим правилом:

```
when
cell c: containsLabel()
entry e: cell.rt == c.rt || cell.cl == c.cl
then
add label c.label[0] to e
add label c.label[1] to e
```

Во втором случае каждая ячейка данных содержит одновременно метку и связанное вхождение в виде пары *ключ–значение* (рис. 4.15, *б*). Следующее правило порождает метку из *ключа* и связанное с ней вхождение из *значения* пары:

```
when
cell c: rt > 1
then
new label c as left(c, '=')
new entry c as right(c, '=')
add label c.label to c.entry
```

4.4.8. Сноски

В зависимости от целевого представления можно по-разному интерпретировать сноски. В таблице (рис. 4.16, *а*) сноски `x` и `y` связаны с вхождениями `1` и `2` посредством ссылок `*` и `**`, соответственно. Представленное ниже правило выполняет следующие действия: порождает вхождение из ячейки со ссылкой; порождает метку из соответствующей сноски в подвале, ассоциируя ее с категорией, названной `FOOTNOTES`; связывает данное вхождение с меткой:

```
when
cell footer: onLastRow, fn: text
cell c: text matches ".+\\*+", ref: right(text, "\\d+")
then
new entry c as left(c, "\\*+")
add label between(fn, ref, '\n') from "FOOTNOTES" to c.entry
```

В другом случае (рис. 4.16, *б*) сноски `u` и `v` относятся к меткам `b` и `f` соответственно. Представленное ниже правило порождает метку, конкатенируя ее с текстом соответствующей сноски:

```
when
cell footer: onLastRow, fn : text
cell c: text matches ".+\\*+", ref: right(text, "\\d+")
then
new label c as left(c, "\\*+") + between(fn, ref, '\n')
```

	a		b	
	c	d	c	d
e	1*	2**	3	4
f	5	6	7	8
g	9	10	11	12
* x ** y				

а

	a		b*	
	c	d	c	d
e	1	2	3	4
f**	5	6	7	8
g	9	10	11	12
* u ** v				

б

Рисунок 4.16 — Сценарии применения CRL-правил: сноски сопровождают вхождения (*а*) и метки (*б*)

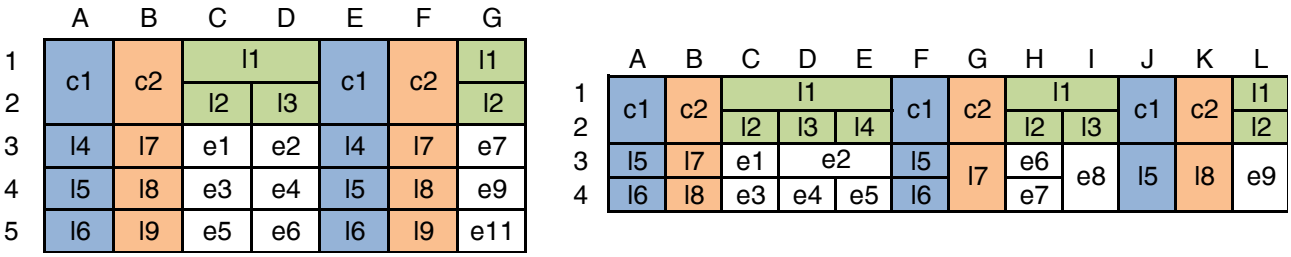


Рисунок 4.17 — Сценарии применения CRL-правил: раскраска функциональных регионов ячеек, при которой каждый цвет соответствует отдельной категории

4.4.9. Раскраска функциональных регионов ячеек

Таблицы, демонстрируемые на рис. 4.17, имеют раскраску функциональных регионов ячеек. Предполагается, что каждый цвет соответствует отдельной категории. Раскраска может использоваться для того, чтобы восстановить вхождения, метки и категории. В следующем правиле выбираются ячейки, выделенные синим цветом, они используются для порождения меток, ассоциированных с категорией `BLUE`:

```
when
cell c: style.bgColor == Color.BLUE
then
new label c
set category "BLUE" to c.label
```

4.5. Исследование пользовательской разработки CRL-правил

Для того чтобы оценить, насколько CRL подходит для разработки правил анализа и интерпретации таблиц конечными пользователями, выполнено следующее исследование. Всего в исследовании приняли участие 10 специалистов — практиков в области программной инженерии и управления данными. Каждый из них имел опыт программирования как минимум на одном язы-

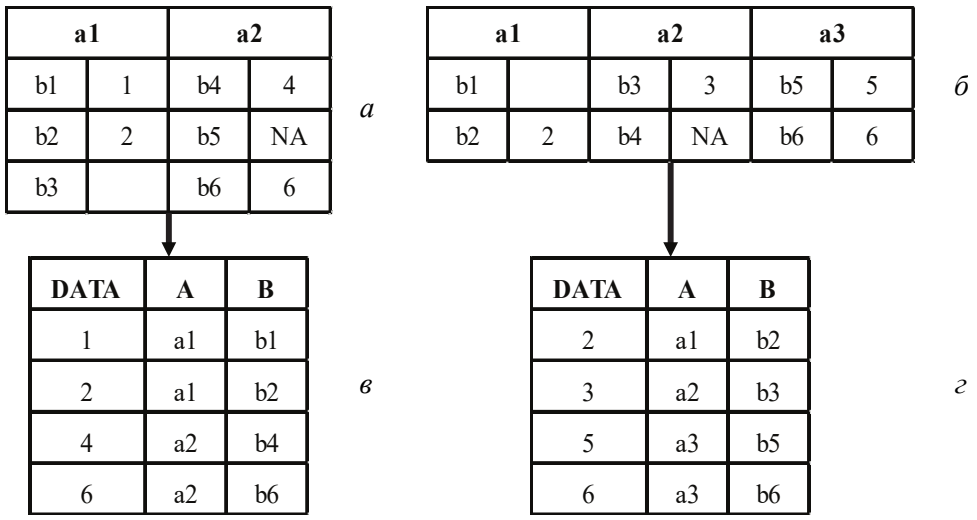


Рисунок 4.18 — Сценарии пользовательского исследования: исходные таблицы ($а, б$) и соответствующие им целевые наборы записей ($в, г$)

ке общего назначения. При этом только двое использовали в своей практике языки правил.

Участники были ознакомлены с принципами разработки CRL-правил и их применения в 40-минутном практическом занятии. После чего каждому участнику было предложено самостоятельно выполнить задание, целью которого являлось разработать набор правил для трансформации исходных таблиц с общими свойствами (рис. 4.18, $а, б$) в целевое представление (рис. 4.18, $в, г$). Исходные таблицы удовлетворяют следующим ограничениям: $1, \ldots, n$ — вхождения; $a_1, \ldots, a_m$ — столбцевые метки категории A; $b_1, \ldots, b_k$ — строчные метки категории B. Следует отметить, что представленная компоновка взята из реальных таблиц, используемых для сбора информации об электрическом и техническом оборудовании в одной энергетической компании.

Задание предполагало разработку правил, чтобы выполнить семь следующих действий: очистка данных; генерация вхождений; генерация меток; ассоциирование вхождений с столбцевыми метками; ассоциирование вхождений с метками строк; категоризация столбцевых меток; категоризация строч-

(1)
```
when cell $c: text == "NA"
then set text "" to $c
```

(2)
```
when cell $c: (cl % 2) == 0, !blank
then new entry $c
```

(3)
```
when cell $c: (cl % 2) == 1
then new label $c
```

(4)
```
when
  entry $e
  label $l: cell.cr == $e.cell.cr
then add label $l to $e
```

(5)
```
when
  entry $e
  label $l: cell.rt == $e.cell.rt, cell.cl == $e.cell.cl - 1
then add label $l to $e
```

(6)
```
when label $l: cell.rt == 1
then set category "A" to $l
```

(7)
```
when label $l: cell.rt > 1
then set category "B" to $l
```

Рисунок 4.19 — Эталонные CRL-правила пользовательского исследования

ных меток. Эталонный набор из семи правил, реализующих данные действия, показан на рис. 4.19.

Участники предложили свои наборы правил. Только трое не допустили никаких ошибок. Все ошибки были сделаны в левой части правил. Синтаксические ошибки были вызваны некорректной записью Java-выражений в условиях правил: оператора остатка (`mod` вместо `%`) — четыре случая; оператора сравнения равенства (`=` вместо `==`) — три; пар скобок — один. Семантические ошибки были вызваны следующими причинами: пропущенные ограничения, необходимые для правильного связывания записей с метками строк, — шесть случаев; неопределенный идентификатор — четыре; избыточные ограничения, препятствующие правильному запросу фактов, — два; использование координаты `cl` вместо `cr` — два случая. Следует отметить, что два участника допустили около 87 % (7/8) от всех синтаксических ошибок и 64 % (9/14) от всех семантических. Часть допущенных ошибок была исправлена после неудачного запуска процесса исполнения правил.

Полученные результаты показали, что предлагаемая платформа разработки ПО может использоваться экспертами в области управления и анализа данных. Квалифицированные пользователи, включая программных инженеров и аналитиков данных, могут разрабатывать наборы правил для конкретных задач трансформации таблиц. Вероятно, что использование визуального

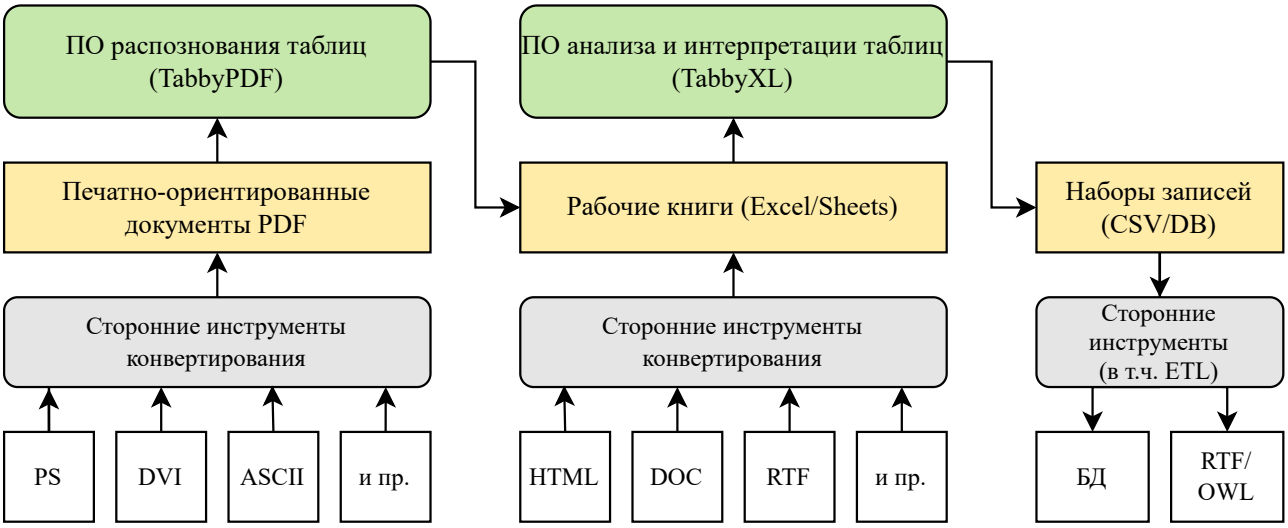


Рисунок 4.20 — Единое технологическое решение автоматизации извлечения данных из документных таблиц

редактора с подсветкой синтаксиса и автодополнением или редактирование при помощи диалогов может снизить количество синтаксических ошибок.

4.6. Выводы

Разработанный КИС составлен ПО для РТ на основе метода, представленного в главе 2 (TabbyPDF) и АИТ на основе метода, представленного в главе 3 (TabbyXL). Следует отметить, что представленные решения TabbyPDF и TabbyXL могут использоваться как совместно, так и независимо друг от друга (рис. 4.20). В случае совместного применения они обеспечивают АПТ полного цикла, а в случае независимого — ограничиваются соответствующими им задачами. Более того, самостоятельную ценность имеют отдельные программные реализации предложенных ранее методик, в том числе сегментации страниц, фильтрации кандидатных случаев, а также автоматизации трансферного обучения ИНС обнаружения таблиц. Они могут самостоятельно использоваться в сторонних решениях. Исходный код реализованного ПО опубликован в открытом доступе под свободными лицензиями (приложение Г).

Результат, представленный в данной главе

Комплекс инструментальных средств, реализующий теоретические основы, предлагаемые в предыдущих главах. По сравнению с конкурентными решениями реализованный процесс распознавания таблиц НПОД (PDF) дополнен этапами предобработки — сегментации страницы документа и постобработки — фильтрации кандидатных случаев. Процесс извлечения данных из таблиц РК впервые реализован на основе пользовательского программирования правил (в отличие от аналогов предположения о структуре таблиц отделены от моделей представления и алгоритмов обработки).

Публикации

Представленный результат опубликован в ряде рецензируемых изданий, в том числе журналах [1, 2, 5, 6, 50], а также материалах всероссийских и международных мероприятий [23, 27, 32, 34, 36–38, 40, 43]. Получены свидетельства о государственной регистрации программ для ЭВМ [13–18].

Глава 5

Экспериментальные результаты

Глава представляет результаты оценки производительности реализованных решений, а также их количественное и качественное сравнение с имеющимися аналогами. Приводятся показатели оценки производительности, используемые в рассматриваемых экспериментах и сравнениях (раздел 5.1). Излагаются результаты решений распознавания таблиц НПОД на базе ПО TabbyPDF, полученные с помощью известных тестов: ICDAR-2013, SciTSR и IAIS (разделы 5.2 и 5.3). Излагаются результаты решений анализа и интерпретации таблиц РК на базе ПО TabbyXL, полученные с помощью собственных тестов на базе известных коллекций данных: Troy200 и SAUS200 (разделы 5.4 и 5.5). В конце главы делаются выводы (раздел 5.6).

5.1. Показатели оценки производительности

Рассматриваемая оценка производительности и количественное сравнение с аналогами базируется на измерениях *точности* (P), *полноты* (R) и F*-меры* (F_1), которые определяются следующим образом:

$$P = \frac{tp}{tp + fp},\ R = \frac{tp}{tp + fn},\ F_1 = 2\frac{P \cdot R}{P + R},$$

где tp — количество истинно-положительных, fp — ложноположительных и fn — ложноотрицательных исходов, значения которых вычисляются в результате сопоставления результатов (набора однотипных объектов R), произведенных тестируемым решением, с эталонными данными теста производительности (набором однотипных объектов GT). Исход сравнения некоторого объекта из R считается *истинно-положительным*, если есть идентичный ему объект в наборе GT и *ложноположительным* — в противном случае. Исход

сравнения некоторого объекта из GT считается *ложноотрицательным*, если среди объектов набора R нет идентичного ему.

5.2. Оценка решений распознавания таблиц

Реализованные решения распознавания таблиц в PDF-документах оценивались с помощью известного теста производительности ICDAR-2013 [189]. Рассмотрим подробнее используемую методику (раздел 5.2.1) и полученные результаты (раздел 5.2.2) оценки.

5.2.1. Методика оценки производительности

Используется известная методика, предложенная M. Göbel и др. [188], которая обеспечивает оценку производительности как обнаружения таблиц так и распознавания ячеек. В первом случае сопоставляются печатаемые символы внутри предсказанных и эталонных ограничивающих рамок таблиц, а во втором — отношения соседства содержимого предсказанных и эталонных непустых ячеек. В первом случае исход считается истинно-положительным, если печатаемый символ присутствует как в предсказанной, так и в эталонной таблице, ложноположительным, если он есть только в предсказанной, и ложноотрицательным, если — только в эталонной. Аналогично во втором случае исход считается истинно-положительным, если отношение соседства содержимого двух непустых ячеек присутствует как в предсказанной, так и в эталонной таблице, ложноположительным, если оно есть только в предсказанной, и ложноотрицательным, если — только в эталонной.

Рисунок 5.1 демонстрирует пример сопоставления предсказанных и эталонных отношений соседства содержимого непустых ячеек. Подсчет исходов

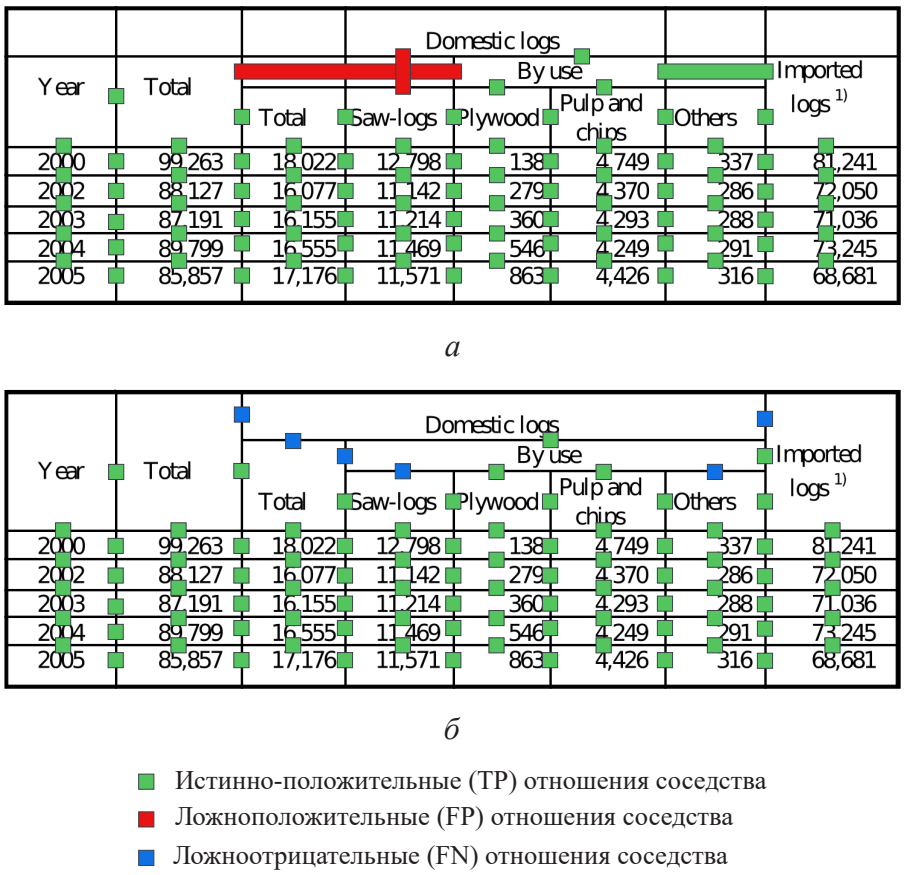


Рисунок 5.1 — Методика оценки производительности M. Göbel и др.: сопоставление результатов распознавания ячеек ($а$) с эталонными данными ($б$)

на данном примере приводит к следующим значениям:

$$tp = 86,\ fp = 2 \text{ и } fn = 6,$$

$$P = 0,9773\ (86/88),\ R = 0,9348\ (86/92) \text{ и } F_1 = 0,9556.$$

5.2.2. Результаты оценки производительности

Результаты получены на предметно-независимом тесте производительности ICDAR-2013 [189]. В нем случайным образом собраны 67 PDF-документов, содержащих всего 156 таблиц вперемежку с текстом и рисунками. Примеры сопровождаются подготовленными вручную эталонными данными в формате XML. Для каждой таблицы определены эталонные координаты

Таблица 5.1 — Оценка производительности решений распознавания таблиц на тесте ICDAR-2013

Показатели	**Обнаружение таблиц**			**Сегментация таблиц**[1]	
	СТС	ОО	ОО+Ф	СПП	АКС
P_{doc}	0,7605	0,8651	0,9703	0,9180	0,9499
R_{doc}	0,8172	0,9795	0,9795	0,9121	0,9233
F_1	0,7878	0,9187	**0,9748**	0,9150	**0,9364**

[1] Используются эталонные ограничивающие рамки таблиц.

Оцениваются следующие варианты: СТС — сопоставление табличных строк; ОО — обнаружение объектов с помощью ИНС; ОО+Ф — ОО с фильтрацией кандидатных случаев; СПП — сегментация пробельного пространства; АКС — анализ компонент связности.

ограничивающей рамки и множество печатаемых символов внутри нее, а также структура ее ячеек, включая их содержимое и отношения соседства.

Сравнение результатов обнаружения и распознавания структуры таблиц с эталонными данными выполняется автоматически с помощью известной реализации соответствующей методики [312]. Количества исходов подсчитываются для каждого тестового документа в отдельности. Средние значения точности (P_{doc}) и полноты (R_{doc}), рассчитанные среди всех документов, принимаются за результат тестирования:

$$P_{doc} = \frac{P_1 + \cdots + P_n}{n} \text{ и } R_{doc} = \frac{R_1 + \cdots + R_n}{n},$$

где P_i — точность и R_i — полнота, измеренная на i-м документе. Результаты оценки производительности решений обнаружения и сегментации таблиц, предлагаемых в главе 2, представлены в табл. 5.1.

Выполненная оценка производительности показала, что решение на основе обнаружения объектов (ОО) дает лучшие результаты по сравнению с решением на основе сопоставления табличных строк (СТС) — $F_1 = 0,9187$ у первого против $F_1 = 0,7878$ у второго. Несмотря на то что полнота (R_{doc})

обнаружения таблиц с помощью ОО достаточно высокая — 0,9795, точность (P_{doc}) остается низкой — 0,8651 из-за большого количества ложноположительных предсказаний ИНС. Более продвинутое решение (ОО+Ф), дополненное фильтрацией кандидатных таблиц, позволило исключить часть ложноположительные случаев. В результате удалось значительно поднять точность (P_{doc}) до значения 0,9703. (Следует отметить, что для реализации вариантов решений на основе СТС, СПП и АКС использовался синтез однострочных блоков текста, а для реализации варианта на основе ОО+Ф — синтез многострочных блоков текста.)

5.3. Сравнение с аналогами распознавания таблиц

Многие исследования со схожей целью предоставляют результаты оценки производительности предлагаемых ими методов на известных тестах, включая ICDAR-2013 [189], SciTSR [128] и IAIS [75], что позволяет сравнивать их между собой. Рассмотрим количественное (раздел 5.5.1) и качественное (раздел 5.5.2) сравнение с этими аналогами.

5.3.1. Количественное сравнение

Сравнение реализованных решений распознавания таблиц с имеющимися аналогами выполнено на трех тестах производительности. На предметно-независимом тесте ICDAR-2013 [189] показано, что TabbyPDF дает результаты, близкие к лучшим академическим аналогам, немного уступая некоторым индустриальным продуктам (табл. 5.2 и 5.3).

Схожие выводы следуют из сравнения с имеющимися аналогами на двух предметно-ориентированных тестах производительности, а именно SciTSR [128] и IAIS [75]. Оба теста также базируются на методике M. Göbel и др. [188], но с расчетом макросредних и микросредних значений точности и полноты

Таблица 5.2 — Сравнение с аналогами по обнаружению таблиц на тесте ICDAR-2013

Решение	P_{doc}	R_{doc}	F_1	**Решение**	P_{doc}	R_{doc}	F_1
GTE [429]	0,990	0,998	0,993	**ОО**	0,865	0,980	0,919
FineReader*	0,973	0,997	0,985	Nurminen* [312]	0,921	0,908	0,914
Kavasidis [234]	0,981	0,975	0,978	Acrobat*	0,937	0,874	0,904
ОО+Ф	**0,970**	**0,980**	0,978	TableBank	0,800	0,991	0,886
DeepDeSRT	0,974	0,962	0,968	Klampfl [244]	0,825	0,918	0,869
TableNet	0,970	0,963	0,966	**СТС**	**0,761**	**0,817**	**0,788**
OmniPage*	0,957	0,964	0,961	Yildiz* [409]	0,640	0,853	0,731
Tran [385]	0,964	0,952	0,958	Stoffel* [370]	0,754	0,699	0,725
Silva* [364]	0,929	0,983	0,955	Liu* [277]	0,884	0,336	0,486
Hao [198]	0,922	0,972	0,946	KYTHE*	0,367	0,460	0,408
Nitro*	0,940	0,932	0,936	Fang* [172]	0,750	0,270	0,397

[1] Результаты предоставлены M. Göbel и др. [189].

Ост. пояснения см. табл. 5.1.

следующим образом:

$$P_{macro} = \frac{P_1 + \cdots + P_n}{n},\ R_{macro} = \frac{R_1 + \cdots + R_n}{n},$$

$$P_{micro} = \frac{tp_1 + \cdots + tp_n}{tp_1 + fp_1 + \cdots + tp_n + fp_n},\ R_{micro} = \frac{tp_1 + \cdots + tp_n}{tp_1 + fn_1 + \cdots + tp_n + fn_n},$$

где P_i и R_i, tp_i, fp_i и fn_i — значения, полученные на i-й таблице.

Тест SciTSR [128] включает 15.000 таблиц в формате PDF, собранных из научных статей репозитория arXiv[1]. Из них 12.000 предназначены для настройки решений распознавания таблиц, в том числе обучения ИНС, и 3.000 — для оценки производительности. Предоставляются эталонные данные, а именно физические структуры таблиц в формате JSON, сформирован-

[1] `https://arxiv.org`

Таблица 5.3 — Сравнение с аналогами по распознаванию таблиц на тесте ICDAR-2013

Решение	Сегментация таблиц[1]			**Решение**	Распознавание таблиц[2]		
	P_{doc}	R_{doc}	F_1		P_{doc}	R_{doc}	F_1
Nurminen*	0,951	0,941	0,946	FineReader*	0,871	0,884	0,877
АКС	**0,950**	**0,923**	**0,936**	OmniPage*	0,846	0,838	0,842
СПП	**0,918**	**0,912**	**0,915**	**ОО+Ф/АКС**	**0,849**	**0,824**	**0,839**
DeepDeSRT	0,959	0,874	0,914	Nurminen* [312]	0,869	0,808	0,837
Klampfl	0,766	0,782	0,774	Tab.IAIS [305]	0,918	0,762	0,833
Silva* [364]	0,614	0,640	0,627	**СТС/СПП**	**0,834**	**0,830**	**0,832**
KYTHE*	0,570	0,481	0,522	Acrobat*	0,816	0,726	0,769
				Nitro*	0,846	0,679	0,754
				Silva* [364]	0,687	0,705	0,696
				Yildiz* [409]	0,575	0,595	0,585

[1] Без обнаружения таблиц: используются эталонные ограничивающие рамки.

[2] Автоматическое обнаружение и сегментация таблиц.

[3] ОО+Ф/АКС — на основе ИНС обнаружения объектов, фильтрации кандидатных случаев и анализа компонент связности.

[4] СТС/СПП — на основе сопоставления строк и сегментации пробельного пространства.

* Результаты предоставлены M. Göbel и др. [189].

Ост. пояснения см. табл. 5.1.

ные из исходного LaTeX-представления. Соответсвующая коллекция примеров включает подмножество сложных случаев — SciTSR-COMP, в каждом из которых есть объединенные ячейки. К таким случаям относится около 24 % от всех таблиц.

Поскольку тест SciTSR предоставляет только таблицы, а остальное содержимое документа отсутствует, то он может применяться только для оценки производительности решений сегментации таблиц. Результаты количественного сравнения, опубликованные в работе [128], представлены в табл. 5.4.

Таблица 5.4 — Сравнение с аналогами по распознаванию таблиц на тесте SciTSR

Решение	**SciTSR**			**SciTSR-COMP**		
	P_{micro}	R_{micro}	F_1	P_{micro}	R_{micro}	F_1
SEM* [425]	0,977	0,965	0,971	0,968	0,947	0,957
GraphTSR** [128]	0,936	0,931	0,934	0,943	0,925	0,934
TabStruct-Net* [332]	0,927	0,913	0,920	0,909	0,882	0,895
TabbyPDF**	**0,914**	**0,910**	**0,912**	**0,869**	**0,841**	**0,855**
DeepDeSRT** [354]	0,898	0,897	0,897	0,811	0,813	0,812
Acrobat**	0,829	0,796	0,812	0,796	0,737	0,765
	P_{macro}	R_{macro}	F_1	P_{macro}	R_{macro}	F_1
GraphTSR** [128]	0,959	0,948	0,953	0,964	0,945	0,955
TabbyPDF**	**0,926**	**0,920**	**0,921**	**0,892**	**0,872**	**0,882**
DeepDeSRT** [354]	0,906	0,887	0,890	0,863	0,831	0,846
Acrobat**	0,930	0,784	0,851	0,901	0,717	0,798

* Результаты предоставлены Z. Zhang и др. [425].

** Результаты предоставлены Z. Chi и др. [128].

Следует отметить, что Z. Chi и др. [128] исследовали только одно из предлагаемых в диссертации решений, основанное на сопоставлении строк и сегментации пробельного пространства (СТС и СПП). В отличие от ИНС [128, 332, 425], заранее обученных на наборе данных SciTSR, предлагаемое нами решение не настраивалось под работу именно с такими тестовыми таблицами. Тем не менее оно показало результаты, близкие к тем, которые дают специализированные ИНС модели (табл. 5.4).

Тест IAIS [75] состоит из 863 научных статей в формате PDF, собранных из электронного архива PMC[2]. Они содержат 1650 таблиц, среди них

[2] `https://www.ncbi.nlm.nih.gov/pmc`

Таблица 5.5 — Сравнение с аналогами по распознаванию таблиц на тесте IAIS

Решение	**Одновариантная оценка**			**Многовариантная оценка**		
	P_{micro}	R_{micro}	F_1	P_{micro}	R_{micro}	F_1
FineReader*	0,5991	0,5949	0,5970	0,6060	0,6088	0,6074
Tab.IAIS* [305]	0,6675	0,4676	0,5499	0,6601	0,4804	0,5561
TabbyPDF*	**0,5020**	**0,5460**	**0,5231**	**0,5383**	**0,5611**	**0,5494**
Tabula (LM)* [312]	0,6303	0,1296	0,2150	0,5469	0,1316	0,2122
Tabula (SM)* [312]	0,1001	0,3504	0,1557	0,1051	0,3618	0,1629

* Результаты предоставлены T. Adams и др. [75].

628 сложных случаев, содержащих объединенные ячейки. Эталонные данные, представленные в формате XML, описывают физическую структуру таблиц.

Тест IAIS определяет два способа оценки производительности: одновариантный, при котором каждый пример представлен единственным вариантом (PDF-документом); многовариантный, при котором пример может быть представлен несколькими вариантами (PDF-документами). В последнем случае значения tp, fp и fn величин вычисляется для каждого варианта примера, после чего выбирается лучший результат, именно он включается в подсчет микро-средних значений точности и полноты распознавания отношений соседства ячеек. Результаты количественного сравнения приводятся в табл. 5.5.

Следует отметить, что ни одно из протестированных решений не показало высоких результатов (табл. 5.5). Тестирование касалось единственного варианта решения TabbyPDF, на базе сопоставления строк и сегментации пробельного пространства (СТС и СПП). Оно показало близкий результат к академическому аналогу Tab.IAIS [305] и превзошло другое — Tabula [312], являющееся наиболее известным среди доступных аналогов.

Таблица 5.6 — Сравнение с аналогами распознавания таблиц по качественным характеристикам

Метод	**Подход**	**Формат**	**Не требуется**		**PDL-специфика**				
			OCR	Разграфка	Т	Ш	Л	ОТ	ПП
DeepDeSRT	ГО	РИ	-	+	-	-	-	-	-
GraphTSR*	П/ГО	ИР	+	+	+	+	+	+	+
pdf2table	П	ИР	+	+	+	-	-	-	-
Tab.IAIS	П	РИ	-	-	-	-	+	-	-
TabbyPDF	**П/ГО**	**ИР**	+	+	+	+	+	+	+
TableNet	П/ГО	РИ	-	+	-	-	-	-	-
Tabula	П	ИР	+	+	+	-	+	-	-
Tran et al.	П	РИ	-	-	-	-	+	-	-

П — правила, ГО — глубокое обучение; РИ — растровые изображения, ИР — PDL-инструкции. Поддерживаемая PDL-специфика: Т — машиночитаемый текст, Ш — шрифтовые свойства, Л — линейки разграфки (в том числе восстанавливаемые из растра), ОТ — порядок отрисовки текста, ПП — следы перемещения пера.

* Базируется на извлечении блоков текста из PDF-документа, выполняемого TabbyPDF.

5.3.2. Качественное сравнение

Качественное сравнение с аналогами распознавания таблиц приводится в табл. 5.6. Следует отметить, что представленное здесь сравнение охватывает только часть конкурентных методов, которые в целом дают общую картину отличий предлагаемого метода от аналогов по качественным характеристикам. PDL-специфичная информация, такая как порядок отрисовки текста в графическом контексте, следы перемещения пера и пр., используется только предлагаемым в диссертационной работе решением TabbyPDF, а также сторонним аналогом GraphTSR [128]. Однако сам этот аналог базируется на извлечении блоков текста из PDF-документов, выполняемого TabbyPDF.

Остальные аналоги используют только часть доступной PDL-информации, а именно текст и линейки, или вовсе не применяют ее, прибегая к растеризации источников.

5.4. Оценка решений анализа и интерпретации таблиц

Оценка производительности решений анализа и интерпретации таблиц выполнена с помощью двух предметно-ориентированных тестов: Troy200 и SAUS200. Оба были разработаны в рамках диссертации на основе двух известных коллекций реальных таблиц, собранных случайным образом с веб-ресурсов англоязычной государственной статистики. Прежде всего вводится формат эталонных данных (раздел 5.4.1) и методика оценки производительности (раздел 5.4.2), общие для обоих тестов, а также методику предобработки данных коллекции SAUS200 (раздел 5.4.3). Затем рассматриваются тестовые задачи, включая описания соответствующих коллекций примеров, тестируемые решения и полученные результаты (разделы 5.4.4 и 5.4.5).

5.4.1. Эталонные данные

Эталонные данные описывают функциональные элементы и их связи, тем самым позволяя выполнять оценку как ролевого, так и структурного анализа таблиц. В рамках диссертационного исследования они были подготовлены в едином машиночитаемом формате (приложение Г). Каждой исходной таблице из тестовой коллекции сопоставлена пара наборов вхождений `ENTRIES` и меток `LABELS`. Первый из них перечисляет вхождения в виде троек: значение, провенанс (адрес ячейки в исходной таблице), множество связанных с ней меток. Второй из них перечисляет метки в виде троек: значение, провенанс, родительская метка при наличии. Ниже демонстрируются два примера: фрагмент набора вхождений `ENTRIES`:

AWARD AND RECIPIENCY STATUS	2001			
	ALL CUSTODIAL PARENTS			
	Total		Mothers	Fathers
	Number	Percent distribution		
unit indicator	(1,000)		(1,000)	(1,000)
Total	13 383	(X)	11 291	2 092
With child support agreement or award \1	7 916	(X)	7 110	807
Supposed to receive payments in year shown	6 924	100,0	6 212	712
Actually received payments in year shown	5 119	73,9	4 639	480
Received full amount	3 099	44,8	2 821	278
Received partial payments	2 020	29,2	1 818	202
Did not receive payments in year shown	1 804	26,1	1 573	232
Child support not awarded	5 466	(X)	4 181	1 285
MEAN INCOME AND CHILD SUPPORT	(Dollars)		(Dollars)	
Received child support payments in year shown:				
Mean total money income (dollars)	29 008	(X)	28 258	36 255
Mean child support received (dollars)	4 274	(X)	4 274	4 273
Received the full amount due:				
Mean total money income (dollars)	32 338	(X)	31 734	38 479
Mean child support received (dollars)	5 665	(X)	5 655	5 768
Received partial payments:				
Mean total money income (dollars)	23 899	(X)	22 865	33 199
Mean child support received (dollars)	2 141	(X)	2 132	2 219

Рисунок 5.2 — Использование цветовой разметки таблицы для генерации эталонных данных: желтый соответствует угловику, красный — шапке, синий — боковику, зеленый — перерезам, серый — телу. Каждая градация синего цвета соответствует отдельному уровню вложенности иерархии меток в ячейках боковика. Демонстрируется фрагмент размеченной таблицы 0556 из коллекции SAUS200.

```
ENTRY   PROVENANCE   LABELS
243     T11          "2002 [B11]", "balance [T4]"
2871    S11          "2002 [B11]", "imports [S4]"
```

и фрагмент парного набора меток `LABELS`:

```
LABEL           PROVENANCE          PARENT
balance         T4                  other ict goods [R3]
imports         S4                  other ict goods [R3]
```

Следует отметить, что сами эталонные данные были получены следующим образом. Прежде всего эксперты скорректировали заголовки исходных таблиц в тех случаях, когда машиночитаемые ячейки отличались от человекочитаемых. В результате физическая структура каждого случая стала полностью соответствовать его визуальному представлению. Затем уже скорректированные таблицы были раскрашены с помощью цветовой разметки. Эксперты присвоили заранее заданный цвет фона каждой функциональной области ячеек, как показано на рис. 5.2. При наличии иерархии меток боковика используются градации синего. Каждая градация соответствует определенному уровню вложенности меток боковика. Значения градаций родительских и дочерних меток различаются на одну условную единицу, как показано на рис. 5.2. Затем были разработаны CRL-правила анализа и интерпретации таких таблиц по их цветовой разметке. В результате их исполнения из размеченных таблиц были автоматически сгенерированы наборы записей в канонической форме с соответствующими им наборами вхождений `ENTRIES` и меток `LABELS`.

5.4.2. Методика оценки производительности

В обоих случаях рассчитываются значения точности, полноты и F-меры извлечения функциональных элементов и связей между ними суммарно на всех таблицах. При тестировании результатов анализа и интерпретации таблиц для каждого исходного примера генерируется не только соответствующий набор записей в канонической форме, но также и пара наборов записей: `ENTRIES` и `LABELS`, в предложенном формате 5.4.1. Предсказанные вхождения и метки, а также связи типа «вхождение–метка» и «метка–метка», сопоставляются с эталонами. Исход считается истинно-положительным, если функциональный элемент / связь присутствует как в предсказанных, так

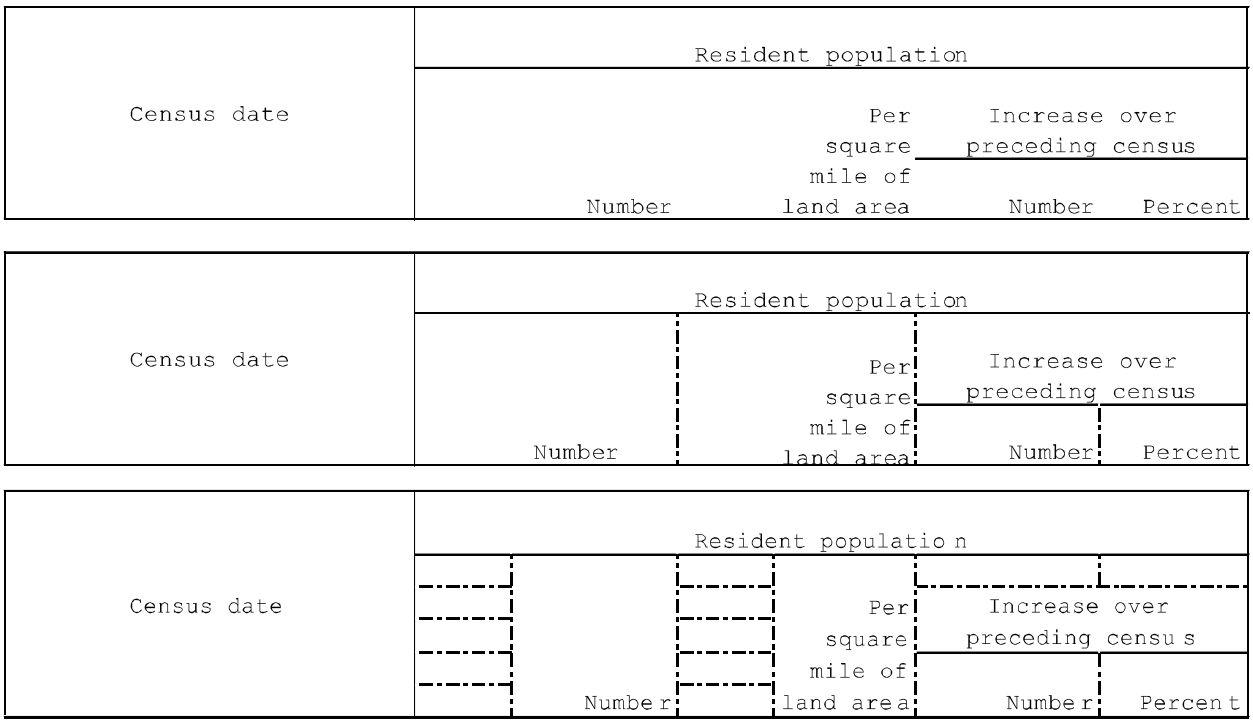


Рисунок 5.3 — Некорректная физическая структура таблицы: визуальное представление ($а$); человекочитаемые ячейки ($б$); машиночитаемые ячейки ($в$). (Сплошные линии соответствуют видимым, а штрихпунктирные — невидимым границам ячеек)

и в эталонных данных, ложноположительным, если он есть только в предсказанных, и ложноотрицательным, если — только в эталонных. Настоящая методика реализована в виде ПО (приложение Г), что позволяет выполнять оценку производительности тестируемого решения автоматически.

5.4.3. Предобработка тестовых данных

Изучение тестовых примеров коллекции SAUS200 показало, что примерно в половине случаев невозможно считать корректную физическую структуру ячеек заголовка — угловика и шапки, так как представленная визуально разграфкой (рис. 5.3, $а$) человекочитаемая структура ячеек (рис. 5.3, $б$) не совпадает с машиночитаемой (рис. 5.3, $в$). (Одна человекочитаемая ячейка может быть физически представлена несколькими машиночитаемыми.)

Некорректная физическая структура препятствует корректному восстановлению логической структуры. Например, наличие пустых ячеек может приводить к ошибкам связывания функциональных элементов. Для того чтобы улучшить качество анализа и интерпретации таблиц, тестируемое решение следует предварить предобработкой тестовых данных. Настоящее исследование предлагает решение, которое в дополнение к тестируемому набору правил выполняет коррекцию физической структуры заголовка. В результате она может быть приближена к его визуальному представлению.

Предобработка базируется на предположении K. W. Broman и K. H. Woo [103] о том, что заголовок не может иметь пустых ячеек. Когда хотя бы одна из четырех границ пустой ячейки является невидимой, то она должна быть объединена с соседней ячейкой. Дополнительное предположение состоит в том, что каждая строка многострочного текста может размещаться в отдельной однострочной ячейке. Когда сохраняется форматирование текста (горизонтальное выравнивание, шрифт), то они должны сливаться.

Будем считать, что две соседние ячейки с общей границей могут объединяться тогда и только тогда, когда выполняются следующие ограничения: общая граница является невидимой; общая сторона является горизонтальной (вертикальной), то они расположены в одинаковых столбцах (строках); общая сторона является горизонтальной (вертикальной), то их левые и правые (верхние и нижние) границы попарно являются, либо видимыми, либо невидимыми. Пусть пара соседних ячеек удовлетворяет перечисленным выше ограничениям, тогда будем считать, что они составляют одну объединенную ячейку в следующих случаях: по крайней мере одна из них пустая; при общей горизонтальной стороне, их тексты имеют одинаковое форматирование (горизонтальное выравнивание, шрифт).

Предобработка выполняется следующим образом. Прежде всего сливаются непустые соседние ячейки с учетом приведенных выше условий (рис. 5.4,

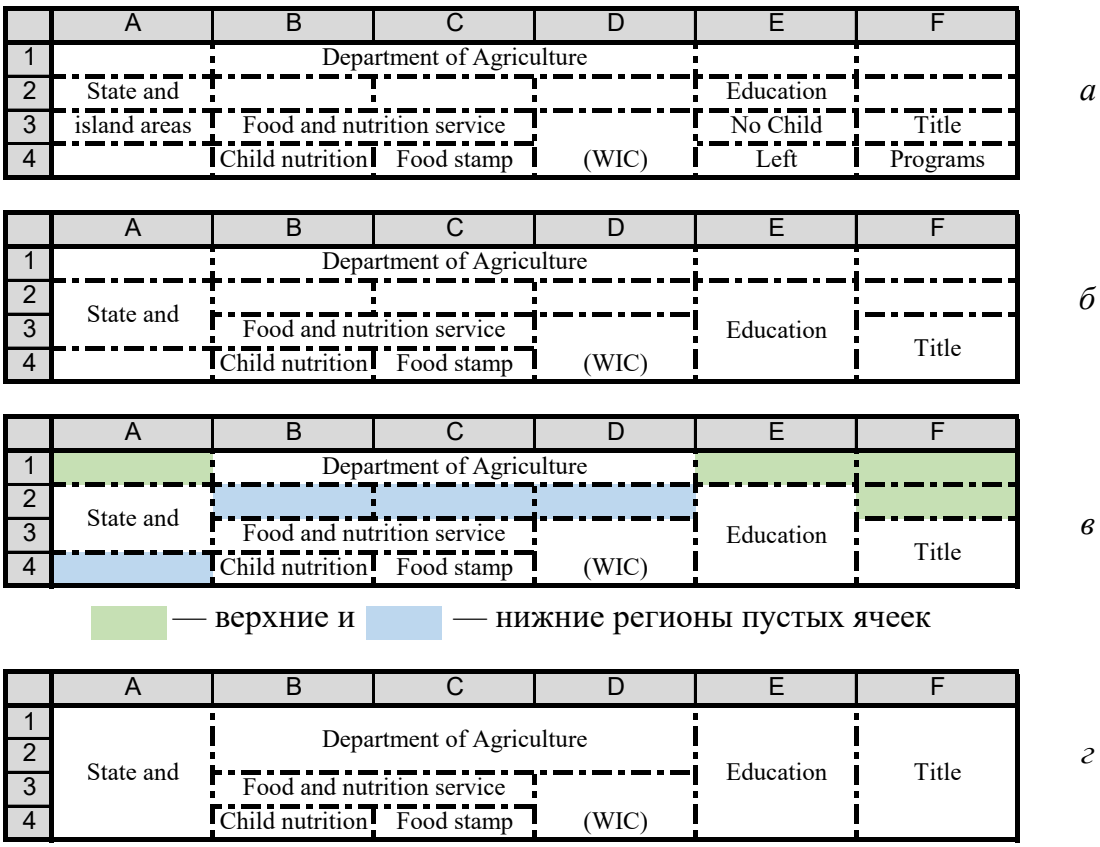


Рисунок 5.4 — Коррекция физической структуры таблицы: исходные ячейки заголовка (*а*); после объединения непустых ячеек (*б*); определение регионов пустых ячеек (*в*); после объединения пустых ячеек (*г*)

а, б). Затем непустые ячейки рассматриваются в порядке их размещения слева направо сверху вниз. Для каждой из них рассматривается два региона пустых ячеек, удовлетворяющих приведенным выше ограничениям: один из которых размещен непосредственно над ней, а другой — под ней (рис. 5.4, *в*). При этом компоновка ячеек такого региона строго соответствует решетке. Сама непустая ячейка и пустые ячейки соответствующих ей регионов объединяются (рис. 5.4, *г*). Например, две непустых ячейки `A2` и `A3` сперва сливаются в `A2:A3`; затем для нее определяются два региона пустых ячеек `A1` и `A4`; в результате они составляют ячейку `A1:A4` (рис. 5.4, *а*–*г*).

Экспериментальные результаты предлагаемого решения предобработки таблиц коллекции SAUS200 приводятся в табл. 5.7. Исходные заголовки суммарно включали 8028 ячеек, а корректные заголовки — только 3768. Ав-

Таблица 5.7 — Экспериментальные результаты предобработки таблиц SAUS200

Количество ячеек в заголовках	
до коррекции	8028
после ручной коррекции	3768
после автоматической коррекции	3795
Показатели	
доля корректно восстановленных ячеек	93,2 %
доля корректно объединенных ячеек	97,5 %
доля корректно восстановленных заголовков	95,5 %

томатическая коррекция заголовков сократила количество ячеек с 8028 до 3795. Из них 93,2 % соответствовали корректным случаям. В результате, доля корректно объединенных ячеек составила 97,5 %, а доля корректно восстановленных заголовков — 95,5 % (только 9 случаев из 200 остались частично некорректными). Таким образом, можно утверждать, что предобработка существенно сократила количество тех случаев, когда машиночитаемые ячейки отличаются от человекочитаемых.

5.4.4. Тестовая задача Troy

Данная тестовая задача состоит в том, чтобы восстановить логическую структуру таблиц из коллекции Troy200[3]. Исходные таблицы преимущественно следуют общим ограничениям. Целевое представление соответствует канонической форме. Предлагается решение тестовой задачи в виде единственного набора правил анализа и интерпретации таблиц. Выполняется оценка производительности предлагаемого решения.

[3] `https://tc11.cvc.uab.es/datasets/Troy_200_1`

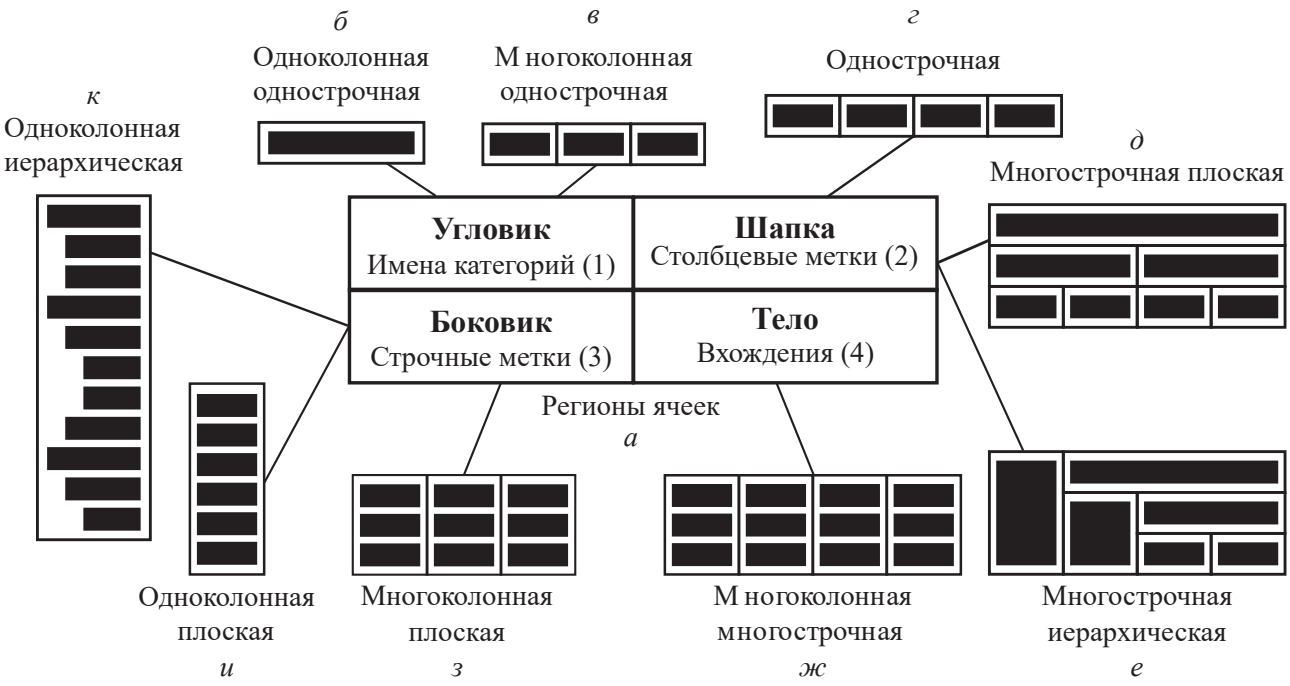


Рисунок 5.5 — Варианты компоновки таблиц из коллекции Troy200

Коллекция тестовых примеров

Коллекция таблиц Troy200 [304] собрана в рамках исследовательского проекта Tango[4] [384]. Всего 200 примеров были выбраны случайным образом из 10 различных веб-сайтов государственной статистики (преимущественно англоязычных). Ее первая версия была получена в результате конвертирования исходных HTML-таблиц в формат PK Excel с сохранением шрифтового форматирования. Именно она используется в настоящем исследовании.

Каждая тестовая таблица может быть сопоставлена некоторой комбинацией из вариантов компоновок функциональных регионов ячеек, как показано на рис. 5.5. Доли случаев использования каждого из этих вариантов приводятся в табл. 5.8. (Некоторые примеры демонстрируются в приложении Б.1.) Большая часть таблиц данной коллекции соответствуют следующим ограничениям:

[4] `https://tango.byu.edu`

Таблица 5.8 — Распределение вариантов компоновок таблиц в коллекции Troy200

Регион ячеек	**Вариант компоновки**	**Доля**
Угловик	Одноколонная однострочная (рис. 5.5, *б*)	94,5 %
	Многоколонная однострочная (рис. 5.5, *в*)	5,5 %
Шапка	Однострочная плоская (рис. 5.5, *г*)	65,5 %
	Многострочная плоская (рис. 5.5, *д*)	26 %
	Многострочная иерархическая (рис. 5.5, *е*)	8,5 %
Боковик	Одноколонная иерархическая (рис. 5.5, *к*)	47,5 %
	Одноколонная плоская (рис. 5.5, *и*)	47 %
	Многоколонная плоская (рис. 5.5, *з*)	5,5 %
Тело	Многоколонная многострочная (рис. 5.5, *ж*)	100 %

- Каждая из них содержит четыре функциональных региона ячеек: угловик, шапку, боковик и тело, содержащие имена категорий, столбцевые метки, строчные метки и вхождения соответственно.
- Каждый регион ячеек реализует один из вариантов компоновок, как показано на рис. 5.5.
- Каждая непустая ячейка угловика содержит имя категории.
- Каждая непустая ячейка шапки содержит столбцевую метку.
- Каждая непустая ячейка боковика содержит строчную метку.
- Каждая непустая ячейка тела содержит вхождение.
- Многострочная компоновка шапки (рис. 5.5, *д*, *е*) образует иерархию меток: дочерние метки соответствуют вложенным ячейкам, родительские — охватывающим.
- Одноколонная компоновка боковика (рис. 5.5, *к*) может образовывать иерархию строчных меток одним из следующих способов: каждый уровень вложенности меток добавляет один дополнительный отступ, рав-

ный двум пробелам; знак дефиса в начале метки указывает, что она является дочерней; текст, выделенный жирным шрифтом, может обозначать родительскую метку.

- Вхождения являются либо числами, либо специальными значениям ($\#$, X, F, NA и пр.).

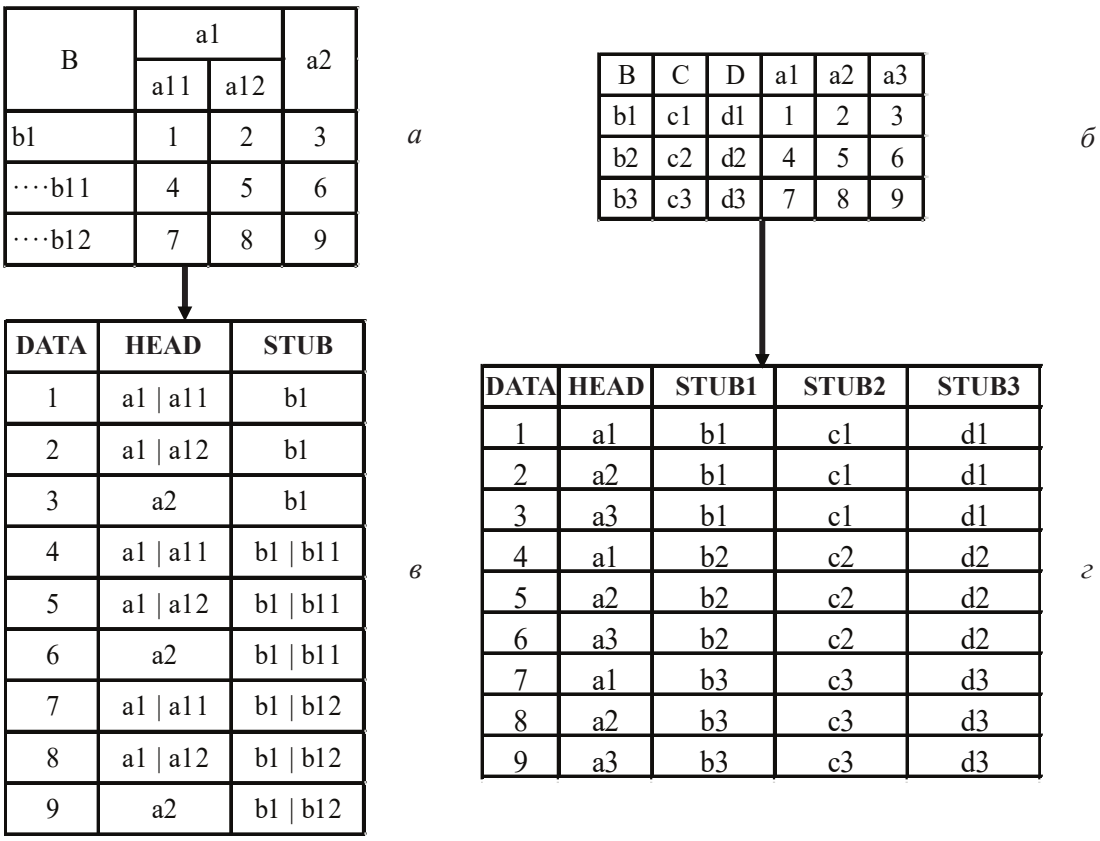
Целевое представление

Предполагается, что данные, восстановленные в результате анализа и интерпретации таблиц из коллекции Troy200, могут быть представлены в канонической форме, как показано на рис. 5.6. Все столбцевые метки добавляются в одну иерархическую категорию — поле `HEAD`. Все строчные метки одного столбца добавляются в отдельную иерархическую категорию — поля: `STUB1`,..., `STUBn`.

Тестируемое решение CRL-16

Тестируемое решение реализовано в виде 16 CRL-правил:

- *Правила 1–3* исключают из текста специальные значения: $\#$, X, F, NA и пр., заменяя их на соответствующие числовые значения при необходимости.
- *Правило 4* создает столбцевые метки из ячеек первой строки шапки (рис. 5.5, *г*–*е*).
- *Правило 5* создает и связывает столбцевые метки в случае многострочной компоновки шапки (рис. 5.5, *д*, *е*). Предполагается, что если ячейка c из i-й строки охватывает несколько столбцов с ячейками $c_1, \ldots, c_n$ из $(i+1)$-й строки, тогда ячейке c соответствует родительская метка, а ячейкам $c_1, \ldots, c_n$ — дочерние метки.



B	a1		a2
	a11	a12	
b1	1	2	3
····b11	4	5	6
····b12	7	8	9

а

B	C	D	a1	a2	a3
b1	c1	d1	1	2	3
b2	c2	d2	4	5	6
b3	c3	d3	7	8	9

б

DATA	HEAD	STUB
1	a1 | a11	b1
2	a1 | a12	b1
3	a2	b1
4	a1 | a11	b1 | b11
5	a1 | a12	b1 | b11
6	a2	b1 | b11
7	a1 | a11	b1 | b12
8	a1 | a12	b1 | b12
9	a2	b1 | b12

в

DATA	HEAD	STUB1	STUB2	STUB3
1	a1	b1	c1	d1
2	a2	b1	c1	d1
3	a3	b1	c1	d1
4	a1	b2	c2	d2
5	a2	b2	c2	d2
6	a3	b2	c2	d2
7	a1	b3	c3	d3
8	a2	b3	c3	d3
9	a3	b3	c3	d3

г

Рисунок 5.6 — Сценарии тестируемого решения CRL-16: исходные таблицы ($а, б$) и их канонические формы ($в, г$)

- *Правило 6* создает строчные метки из ячеек первого столбца боковика (рис. 5.5, $з$–$к$).
- *Правило 7* создает строчные метки в случае многоколонной компоновки боковика (рис. 5.5, $з$). Предполагается, что столбцы боковика не содержат числовых значений.
- *Правило 8* создает вхождения из ячеек тела (рис. 5.5, $ж$). Предполагается, что строки тела расположены ниже строк шапки, а столбцы тела содержат только числовые значения.
- *Правило 9* заменяя дефисы на пробельные отступы в случае одноколонной иерархической компоновки боковика (рис. 5.5, $к$).
- *Правило 10* связывает строчные метки в случае одноколонной иерархической компоновки боковика (рис. 5.5, $к$). Предполагается, что если

ячейка c расположена выше ячеек $c_1, \ldots, c_n$, текст которых выделен одним пробельным отступом относительно текста c, тогда ячейке c соответствует родительская метка, а ячейкам $c_1, \ldots, c_n$ — дочерние метки.

- *Правило 11* связывает строчные метки в случае одноколонной иерархической компоновки боковика (рис. 5.5, *к*). Предполагается, что если ячейка c, текст которой выделен полужирным шрифтом, расположена выше ячеек $c_1, \ldots, c_n$, текст которых выделен обычным шрифтом, тогда ячейке c соответствует родительская метка, а ячейкам $c_1, \ldots, c_n$ — дочерние метки. (Исключения делаются для меток `Total`, `All` и `I alt`.)
- *Правила 12–14* добавляют метки в категории: `HEAD`, `STUB1`,..., `STUBn` соответственно.
- *Правило 15* связывает вхождение с терминальной меткой, если они порождены из ячеек одного столбца.
- *Правило 16* связывает вхождение с каждой меткой, если они порождены из ячеек одной строки.

Реализация настоящего набора правил на предметно-ориентированном языке CRL опубликована в составе архивного материала (приложение Г).

Результаты оценки производительности

Качество результатов тестируемого решения CRL-16 измерено с помощью предложенной методики (раздел 5.4.2) путем их автоматического сопоставления с эталонными данными. Результаты ролевого анализа таблиц включают значения точности и полноты извлечения вхождений и меток по отдельности и вместе. Аналогично, результаты структурного анализа таблиц составлены значениями точности и полноты извлечения связей типа «вхождение–метка» и «метка–метка» по отдельности и вместе (табл. 5.9).

Таблица 5.9 — Оценка производительности решения анализа и интерпретации таблиц (CRL-16) на тесте Troy200

Показа-	**Функциональные элементы**			**Связи**		
тели	вхождения	метки	все	ВМ	ММ	все
P	1,0000	0,9365	0,9849	0,9966	0,9784	0,9956
R	0,9813	0,9979	0,9850	0,9773	0,9389	0,9751
F_1	0,9906	0,9662	**0,9849**	0,9869	0,9582	**0,9852**

Извлекаются связи двух типов: ВМ — «вхождение–метка» и ММ — «метка–метка».

Всего 25 из 200 тестовых таблиц были обработаны с ошибками: из них 1249 ложноотрицательных и 488 ложноположительных случаев. Следует отметить, что одна из таблиц (поименованная как C10082 в наборе данных Troy200) содержала нечисловые значения вхождений. Поскольку тестируемый набор правил допускал исключительно числовые значения вхождений, обработка данной таблицы привела к значительным ошибкам: 948 ложноотрицательных и 316 ложноположительных случаев, что составило около 73 % от всех ошибок. В остальных 24 таблицах ошибки происходили при распознавании строчных или столбцевых меток, как показано на рис. 5.7. Данные случаи можно классифицировать в соответствии с их причинами.

Всего можно указать пять типов причин (рис. 5.7, *а*), которые вызвали ошибки распознавания строчных меток в настоящем эксперименте, а именно следующие: *наличие перереза* — родительская метка представлена в единственной непустой ячейке некоторой строки; *некорректная компоновка бокового* — части значения одной метки могут быть размещены в нескольких соседних ячейках; *некорректная индентация* — некоторый уровень иерархии представлен отступами разной длинны; *некорректное форматирование* — полужирный шрифт выделяет метку, не являющуюся родительской; *отсутствие форматирования/индентации* — метки дублируются, но уровни иерар-

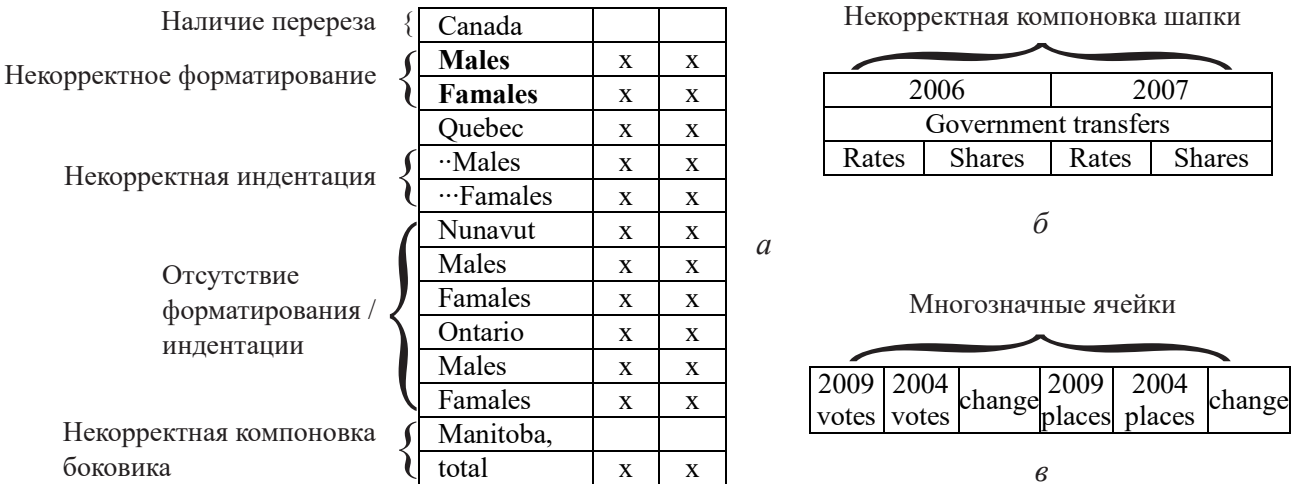


Рисунок 5.7 — Причины ошибок тестируемого решения CRL-16: в боковике ($а$) и шапке ($б$, $в$)

хии меток не выделяются, ни шрифтовым форматированием, ни индентацией. Аналогично, можно указать два типа причин (рис. 5.7, $б$, $в$), которые вызвали ошибки распознавания столбцевых меток в настоящем эксперименте: *некорректная компоновка шапки*: охватывающая ячейка расположена ниже вложенных ячеек; *многозначные ячейки*: ячейка содержит более одной метки.

Экспериментальные результаты показывают, что компоновка, форматирование и индентация не всегда могут быть интерпретированы однозначным образом. Одни и те же приемы могут использоваться для того, чтобы выразить разный смысл. Более того, некоторые приемы могут применяться некорректно. Распределение случаев ложных срабатываний при распознавании меток по типам причин демонстрируется в табл. 5.10. Среди 24 таблиц, обработанных с ошибками, только в одной встречались ошибки двух типов причин (некорректная индентация и компоновка боковика). В каждой из остальных таблиц проявлялись ошибки единственного типа причин.

Таблица 5.10 — Распределение ошибок тестируемого решения CRL-16 по типам причин их возникновения

Типы причин ошибок	Кол-во таблиц	*FN*	*FP*
Ошибки распознавания строчных меток	21	153	84
Наличие перереза	10	34	3
Некорректная компоновка боковика	4	48	41
Некорректная индентация	4	30	14
Некорректное форматирование	2	30	26
Отсутствие форматирования/индентации	2	11	0
Ошибки распознавания столбцевых меток	3	148	88
Некорректная компоновка шапки	2	130	70
Многозначные ячейки	1	18	18
Всего	24	301	172

FN — количество ложноотрицательных, а *FP* — ложноположительных случаев.

5.4.5. Тестовая задача SAUS

Данная тестовая задача состоит в том, чтобы восстановить логическую структуру таблиц из коллекции SAUS200[5]. Исходные таблицы преимущественно следуют общим ограничениям. Целевое представление соответствует канонической форме. Предлагается решение тестовой задачи в виде единственного набора правил анализа и интерпретации таблиц. Выполняется оценка производительности предлагаемого решения.

Коллекция тестовых примеров

Коллекция таблиц SAUS200 содержит 200 примеров в формате PK Excel. Они выбраны случайным образом из корпуса таблиц государственной стати-

[5] `http://dbgroup.eecs.umich.edu/project/sheets/datasets.html`

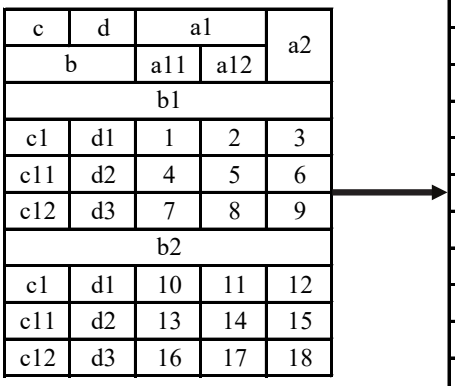
Characteristic	$del $del Selected characteristic $del	Total	Ever told had asthma	
			Number \1	Percent \2
Total \3	**<chgrow;bold>Total \3**	**73,728**	**9,605**	**13.1**
SEX \4	<lp;4q>SEX \4<c>			
Male	Male	37,686	5,550	14.8
Female	Female	36,042	4,055	11.3

Рисунок 5.8 — Предварительная очистка таблиц SAUS200: фрагмент таблицы с удаляемым столбцом, содержащим служебную информацию (изначально столбец являлся скрытым, но в данном примере он раскрыт и выделен серым)

стики США — SAUS[6] (The 2010 Statistical Abstract of the United States). (Z. Chen и M. Cafarella [117] представили исходную коллекцию из 1369 таблиц SAUS, а позднее G. Nagy [304] опубликовал случайную выборку из 200 примеров данной коллекции — SAUS200.) Многие из исходных примеров имели скрытые столбцы, которые обычно содержали служебную информацию (рис. 5.8). Содержательно они не влияли на понимание самих таблиц. В рамках данного исследования они были исключены, чтобы избежать необходимости их учета в процессе анализа и интерпретации таблиц.

Следует отметить, что тестовые таблицы из корпуса SAUS по своей структуре схожи с примерами из ранее рассмотренной тестовой коллекции Troy200 (раздел 5.4.4). Однако в целом таблицы двух коллекций следуют разным приемам компоновки и форматирования. В частности, отличия таблиц коллекции SAUS200 состоят в следующем: наличие перерезов — ячеек, пересекающих шапку или тело по нескольким столбцам (рис. Б.5); метки, порожденные из перерезов, могут образовывать иерархию (рис. Б.6); составная компоновка, при которой столбцы боковика чередуются со столбцами тела (рис. Б.7).

[6] https://www.commerce.gov/bureaus-and-offices/census



c	d	a1		a2
b		a11	a12	
b1				
c1	d1	1	2	3
c11	d2	4	5	6
c12	d3	7	8	9
b2				
c1	d1	10	11	12
c11	d2	13	14	15
c12	d3	16	17	18

DATA	HEAD	CUT-IN	STUB1	STUB2
1	a1 | a11	b1	c | c1	d1
2	a1 | a12	b1	c | c1	d1
3	a2	b1	c | c1	d1
4	a1 | a11	b1	c | c1 | c11	d2
5	a1 | a12	b1	c | c1 | c11	d2
6	a2	b1	c | c1 | c11	d2
7	a1 | a11	b1	c | c1 | c12	d3
8	a1 | a12	b1	c | c1 | c12	d3
9	a2	b1	c | c1 | c12	d3
10	a1 | a11	b2	c | c1	d4
11	a1 | a12	b2	c | c1	d4
12	a2	b2	c | c1	d4
13	a1 | a11	b2	c | c1 | c11	d5
14	a1 | a12	b2	c | c1 | c11	d5
15	a2	b2	c | c1 | c11	d5
16	a1 | a11	b2	c | c1 | c12	d6
17	a1 | a12	b2	c | c1 | c12	d6
18	a2	b2	c | c1 | c12	d6

а *б*

Рисунок 5.9 — Сценарий тестируемого решения CRL-18: исходная таблица (*а*) и ее каноническая форма (*б*)

Целевое представление

В качестве целевого представления принимается каноническая форма следующего вида (рис. 5.9). Все столбцевые метки, исключая порожденные из перерезов, добавляются в одну иерархическую категорию — поле `HEAD`, а метки перерезов в другую — поле `CUT-IN`. Все строчные метки одного столбца составляют отдельную иерархическую категорию — поля: `STUB1`,. . . , `STUBn`.

Тестируемое решение CRL-18

Отличия исходных/целевых данных двух тестовых задач не позволяют эффективно применить набор правил тестовой задачи Troy (CRL-16) к анализу и интерпретации таблиц из корпуса SAUS. В качестве тестируемого решения предложен другой набор из 18 CRL-правил:

- *Правило 1* удаляет символы разрыва строки из текста ячейки.

- *Правила 2-3* создают и категоризируют метку в ячейке перереза.
- *Правила 4-5* создают и категоризируют метку в ячейке угловика.
- *Правило 6* создает и категоризирует метку в ячейке перереза.
- *Правило 7* создает и категоризирует метку в ячейке шапки.
- *Правило 8* создает вхождение в ячейке тела.
- *Правила 9-13* связывает две метки из боковика в родительско-дочернюю пару (каждое правило для отдельного приема компоновки боковика).
- *Правило 14* связывает вхождение и метку из боковика.
- *Правило 15* связывает две метки из шапки в родительско-дочернюю пару.
- *Правило 16* связывает две метки из перерезов в родительско-дочернюю пару.
- *Правило 17* связывает вхождение с меткой из шапки.
- *Правило 18* связывает вхождение с меткой из перереза.

Реализация настоящего набора правил на предметно-ориентированном языке CRL опубликована в составе архивного материала (приложение Г).

Результаты оценки производительности

Качество результатов тестируемого решения CRL-18 измерено с помощью предложенной методики (раздел 5.4.2) путем их автоматического сопоставления с эталонными данными. Оценка производительности получена на трех вариантах тестовых таблиц коллекции SAUS: исходные без коррекции, автоматически и вручную скорректированные. Первый вариант показывает эффективность тестируемого решения без предобработки, второй — совместно с автоматической предобработкой (раздел 5.4.3), и третий — совместно с ручной предобработкой. Полученные результаты приводятся в табл. 5.11.

Таблица 5.11 — Оценка производительности решения анализа и интерпретации таблиц (CRL-18) на тесте SAUS200

Показатели	**Функциональные элементы**			**Связи**		
	вхождения	метки	все	ВМ	ММ	все
Без предобработки						
P	0,8189	0,8207	0,8191	0,7668	0,7007	0,7638
R	0,8783	0,7544	0,8624	0,6896	0,6392	0,6874
F_1	0,8476	0,7862	**0,8402**	0,7262	0,6685	**0,7236**
С автоматической предобработкой						
P	0,8310	0,8571	0,8342	0,8108	0,7716	0,8092
R	0,8791	0,8617	0,8769	0,8379	0,7557	0,8342
F_1	0,8544	0,8594	**0,8550**	0,8241	0,7636	**0,8215**
С ручной предобработкой						
P	0,9420	0,9446	0,9423	0,9275	0,8636	0,9248
R	0,9928	0,9360	0,9856	0,9549	0,8391	0,9498
F_1	0,9667	0,9403	**0,9635**	0,9410	0,8512	**0,9371**

Извлекаются связи двух типов: ВМ — «вхождение–метка» и ММ — «метка–метка».

Представленные результаты подтверждают, что качество ролевого и структурного анализа таблиц в значительной степени зависит от корректности заголовка таблицы. В частности, автоматическая коррекция улучшила полноту распознавания связей «вхождение–метка» на 15 % и «метка–метка» на 12 %. Сравнительная оценка производительности на трех вариантах тестовых таблиц показала, что при работе с исходными данными без предобработки большая часть ошибок происходит именно из-за некорректной физической структуры. В целом можно утверждать, что предлагаемый набор правил (раздел 5.4.5) является эффективным решением для анализа и интерпретации таблиц корпуса SAUS с корректной физической структурой.

Таблица 5.12 — Результаты дополнительных тестовых задач

Источник	Количество					
	таблиц	ячеек	вхождений	меток	связей между метками	DRL-правил
JAPAN_STAT	15	1088	734	257	102	10
AEROFLOT	13	2047	727	321	167	16
BOEING	21	2156	964	470	196	14
CHINA_STAT	18	7216	4180	862	551	12
CHEVRON	7	812	268	141	89	12
USDA_NASS	7	1553	1175	313	174	16
TOBACCO	16	2844	2195	508	335	10

5.4.6. Дополнительные тестовые задачи

В приведенных ранее тестовых задачах (Troy200/SAUS200) качество работы тестируемых решений не достигает 100 % в силу противоречивости приемов оформления таблиц. Однако для случаев, когда все таблицы имеют однотипную компоновку и форматирование, такое качество может быть достигнуто. Последнее подтверждается экспериментально следующим образом. Сформирована коллекция из семи тестовых задач; в каждой из них источником данных является единственный электронный документ (табл. 5.12), а именно статистический или финансовый отчет. Данное ограничение обеспечивает однозначность приемов оформления таблиц, применяемых в рамках одной тестовой задачи. Во всех случаях необходимо восстановить канонические формы тестовых таблиц. Для каждой задачи разработан набор DRL-правил в качестве решение. Оценивалась производительность извлечения вхождений, меток и связей между метками. Установлено, что во всех случаях точность и

полнота составили 100 %, т. е. все вхождения, метки и связи между метками были извлечены абсолютно корректно.

5.5. Сравнение с аналогами анализа и интерпретации таблиц

Среди известных методов со схожей целью только следующие имеют дело с РК: Tango[7] [166], Senbazuru[8] [120] и HUSS [400]. Рассмотрим количественное (раздел 5.5.1) и качественное (раздел 5.5.2) сравнение с перечисленными аналогами.

5.5.1. Количественное сравнение

Результаты количественного сравнения с аналогами Senbazuru и HUSS приводятся в табл. 5.13. Предлагаемые наборы CRL-правил дают значительно лучшие результаты по сравнению со решением Senbazuru, немного уступая лидеру, а именно недавней разработке HUSS. Следует отметить, что последняя в основном лучше справляется с восстановлением родительско-дочерних связей между ячейками заголовка на коллекции SAUS200. Изучение исходных данных показало наличие там разнообразных приемов визуального оформления иерархических заголовков в боковике. Некоторый уровень относительно соответствующего ему подуровня может быть выделен отступом или выступом произвольной длины, выравниванием по центру, шрифтом, пустой строкой, ключевыми словами и пр. При этом перечисленные приемы оформления могут быть двусмысленными, а число уровней вложенности может доходить до десятка. Большая часть допущенных ошибок была вызвана тем, что не были учтены все варианты применения перечисленных приемов.

[7] `https://tango.byu.edu`

[8] `https://dbgroup.eecs.umich.edu/project/sheets`

Таблица 5.13 — Сравнение с аналогами по анализу и интерпретации таблиц на тестах Troy200 и SAUS200

Показатели		**Troy200**			**SAUS200**		
		Senbazuru	**TabbyXL**	HUSS	Senbazuru	**TabbyXL**	HUSS
P	BM	-	0,98	0,99	-	0,96	0,99
	MM	0,90	0,97	0,99	0,88	0,96	0,98
R	BM	-	0,97	0,98	-	0,95	0,98
	MM	0,89	0,93	0,99	0,88	0,78	0,92
F_1	BM	-	**0,97**	0,98	-	**0,95**	0,99
	MM	0,89	**0,95**	0,99	0,88	**0,86**	0,95

Извлекаются связи двух типов: BM — «вхождение–метка» и MM — «метка–метка».
Представленные данные опубликованы авторами HUSS [400].

Допустимо предположить, что качество результатов можно улучшить за счет совершенствования тестируемого решения.

Сравнение с третьим аналогом Tango выполнено на коллекции Troy200 [304]. Его авторы сводят поставленную задачу к поиску так называемых *критических ячеек*, которые делят таблицу на четыре функциональных региона: угловик, шапку, боковик и тело. В качестве методики оценки производительности G. Nagy и др. [165, 166, 301] предлагают измерять количество истинно-положительных предсказаний таких критических ячеек. Исключив из 200 примеров коллекции Troy200 один тривиальный случай (содержит единственный столбец с данными), они показали, что доля истинно-положительных предсказаний критических ячеек у их решения достигает значения — 98,99 % (197/199).

Несмотря на то что сравниваемые решения Tango и TabbyXL оценивались разными методиками производительности, полученные результаты можно сопоставить следующим образом. Исполнение набора CRL-правил сопоста-

вило каждой ячейке заданный дескриптор, соответствующий функциональному региону: шапке, боковику и телу. Это позволило вывести долю случаев, в которых таблицы были разделены на функциональные регионы корректно. Данная оценка эквивалентна доле истинно-положительных предсказаний критических ячеек. В результате исполнения правил всего 7 из 200 тестовых таблиц было обработано с ошибками ролевого анализа. Однако только в двух случаях функциональные регионы были разделены некорректно. Таким образом, доля корректных случаев разделения таблиц на функциональные регионы у решения TabbyXL составила 99,50 % (198/199), что незначительно лучше по сравнению с аналогом Tango.

5.5.2. Качественное сравнение

Качественное сравнение с аналогами анализа и интерпретации таблиц приводится в табл. 5.14. Следует отметить, что применение конкурентных методов ограничено частным случаем, а именно таблицами англоязычной государственной статистики. Рассматриваемая ими модель предполагает, что любая таблица строго делится на четыре функциональных региона: угловик, шапка, боковик и тело. Следует отметить, что данное предположение выполняется далеко не всегда даже в используемых ими тестовых примерах. Будучи ориентированными на наличие заданных функциональных регионов, они не поддерживают произвольность структуры. Хотя ими выполняется анализ иерархических заголовков, однако ни один из них не касается структурированности самих ячеек. Более того, реализация конкурентных методов основана на императивном программировании и обучении с учителем классификаторов (CRF/SVM).

В отличие от аналогов, предлагаемое решение (TabbyXL) выполняет обработку данных посредством пользовательских правил. Они реализуются с

Таблица 5.14 — Сравнение с аналогами по анализу и интерпретации таблиц по качественным характеристикам

Характеристики	Tango	Senbazuru	HUSS	**TabbyXL**
Ролевой анализ	+	+/-*	+	+
Структурный анализ	+	+	+	+
Предлагаемый подход	П	МО	П	**П**
Сложность реализации	ИП	ИП/ОУ	ИП	**ДП**
Произвольность структуры:				
свободная компоновка	-	-	-	+
структурированность ячеек	-	-	-	+
иерархичность заголовков	+/-**	+	+	+

* Только распознавание функциональных типов строк.

** Только иерархия верхнего заголовка (шапки).

П — правила, МО — машинное обучение.

И/ДП — императивное/декларативное программирование; ОУ — обучение с учителем.

помощью декларативного программирования, упрощенного проблемно-ориентированным языком правил CRL. Специфические ограничения выносятся в CRL-правила, позволяющие порождать целевые алгоритмы. Другим преимуществом предлагаемого метода является поддержка свободной компоновки и структурированности ячеек. В отличие от аналогов, используемая им модель не ограничивает структуру таблицы заданными функциональными регионами атомарных ячеек. Напротив, допускается произвольное размещение функциональных элементов в структурированных ячейках.

5.6. Выводы

Оценка производительности, выполненная на соревновательном наборе данных ICDAR-2013 [189], показывает, что лучшие результаты предлагаемое в диссертации решение полного цикла достигает именно в комбинации с ИНС. При этом невысокая точность ИНС может компенсироваться фильтрацией предсказаний ИНС на основе классификации графовых представлений содержимого кандидатных таблиц. Следует отметить, что качество предсказаний ИНС в общем случае зависит от данных, на которых она была обучена. Например, ИНС, обученная на научных статьях, необязательно обеспечит требуемое качество при работе с технической документацией или финансовыми отчетами. Основное достоинство предлагаемого в диссертации решения состоит в том, что оно может обходиться без ИНС. В зависимости от конечных целей предлагаемое решение может задействовать ИНС или ограничиться методиками, основанными на правилах. При этом тестирование демонстрирует, что и в таком случае оно способно давать приемлемые результаты. Качественное сравнение подтверждает, что конкурентные методы распознавания таблиц либо работают с растром, либо не в полной мере используют PDL-специфику НПОД.

Решение тестовых задач Troy и SAUS продемонстрировало возможности применения правил анализа и интерпретации таблиц. Результаты оценки производительности, выполненной на соответствующих тестах Troy200 и SAUS200, показывает эффективность тестируемых решений, основанных на предлагаемом в диссертации методе. Представлено сравнение с тремя аналогами: Tango, Senbazuru и HUSS. Все они нацелены на извлечение реляционных данных из таблиц PK, ограниченных заданной компоновкой функциональных регионов атомарных ячеек. При количественном сравнении решения на основе TabbyXL дают близкие к аналогам результаты на известных кол-

лекциях примеров: Troy200 и SAUS200. Однако из качественного сравнения с ними следует, что предлагаемый в диссертации метод автоматизации анализа и интерпретации таблиц является единственным, в котором структура документной таблицы не ограниченна типичной компоновкой и атомарностью ячеек. Даже применяемые коллекции примеров содержат не только атомарные, но и структурированные ячейки. Однако аналоги не рассматривают последние вовсе. В отличие от них пользовательские CRL-правила подходят для обоих видов ячеек. Кроме того, по сравнению с конкурентными методами, обеспечивается создание более простых по форме решений. На представленных примерах показано, что 16–18 CRL-правил могут заменить специализированные алгоритмы, реализованные с помощью императивного программирования и обучения с учителем. Важным преимуществом CRL-правил по сравнению с алгоритмами и моделями рассматриваемых аналогов является их доступность для модификации.

Результаты, представленные в данной главе

- Экспериментальные данные, полученные на основе известных тестов ICDAR-2013, SciTSR и IAIS, для обоснования основных результатов диссертационной работы: предлагаемого метода автоматизации распознавания таблиц НПОД (глава 2) и его реализации в виде ПО TabbyPDF (глава 4).
- Два теста производительности решений анализа и интерпретации таблиц РК на основе известных коллекций примеров Troy200 и SAUS200.
- Экспериментальные данные, полученные на основе собственных тестов Troy200 и SAUS200, для обоснования основных результатов диссертационной работы: предлагаемого метода автоматизации анализа и интерпретации таблиц РК (глава 3) и его реализации в виде ПО TabbyXL (глава 4).

Публикации

Представленные результаты опубликованы в ряде рецензируемых изданий, в том числе журналах [1, 2, 6–8], а также материалах всероссийских и международных мероприятий [24, 29, 31, 32, 36, 41]. Разработанные тесты производительности Troy200 и SAUS200 опубликованы в открытом доступе под свободными лицензиями [71, 72].

Глава 6

Применение в прикладных задачах

Глава посвящена применению предлагаемых в диссертации методов и реализованных на их основе программных средств в прикладных задачах. Рассматриваются следующие случаи применения: анализ технических (раздел 6.1), финансовых (раздел 6.2) и научных документов (раздел 6.3); интеграция данных государственной статистики (раздел 6.4); интеграция данных медиапланирования (раздел 6.5); конструирование онтологии предметной области (раздел 6.6); кросс-контекстный обмен бизнес-документами (раздел 6.7). В конце главы делаются выводы (раздел 6.8).

6.1. Анализ технических документов

Настоящее применение касалось *паспортов безопасности химической продукции*. Данный вид технической документации предоставляет информацию, относящуюся к безопасности и гигиене труда при использовании различных химических веществ и продуктов, их содержащих. Текущий формат таких паспортов стандартизирован на международном уровне. «Согласованная на глобальном уровне система классификации и маркировки химических веществ»[1] определяет стандартную спецификацию паспортов. Структурно каждый паспорт состоит из 16 разделов и их подразделов, порядок которых не меняется; он также может сопровождаться приложением, содержащим сценарии воздействия описываемого вещества.

Сегодня такая техническая документация широко применяется во всем мире. Многие компании предлагают услуги по управлению ею, обеспечивая

[1] `https://unece.org/ru/ghs-rev7-2017`

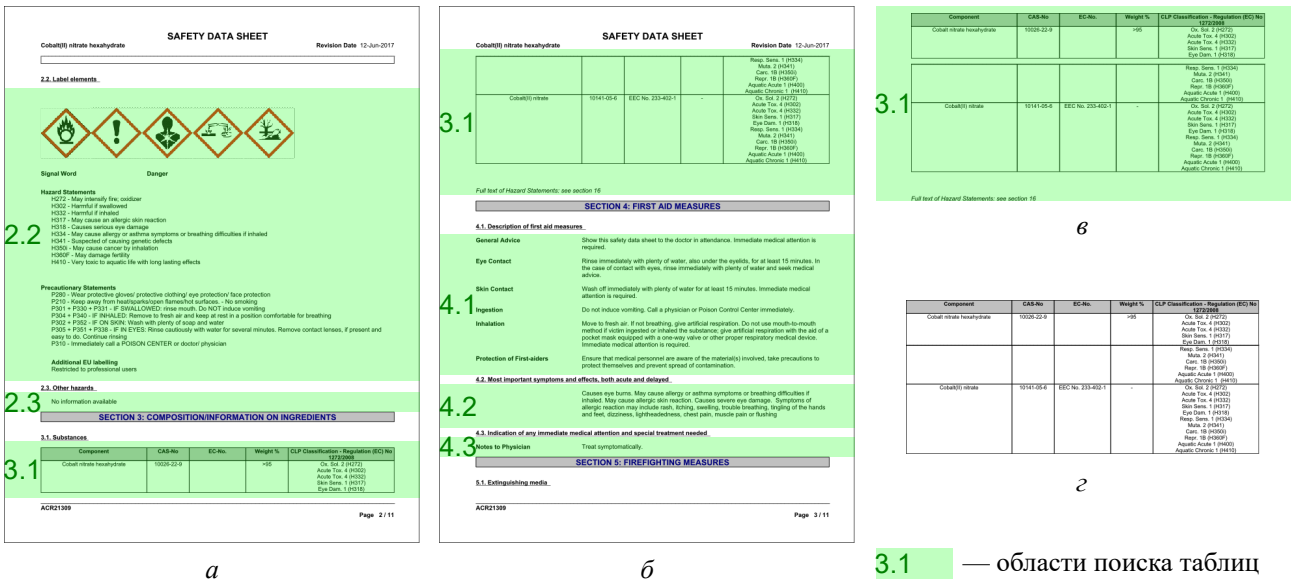


Рисунок 6.1 — Анализ технической документации: извлечение таблицы, размещенной на двух соседних страницах паспорта безопасности химической продукции: выделение областей поиска таблиц внутри разделов исходного документа (*а б*); объединение областей поиска таблиц внутри одного раздела (*в*); и распознавание структуры таблицы (*г*)

актуальность и доступность паспортов для своих клиентов. Некоторые юрисдикции обязывают регулярно ее обновлять; когда появляется новая информация, конкретные паспорта пересматриваются без промедления. Как правило, паспорта распространяются в виде PDF-документов. Каждый из них может содержать неразмеченные таблицы. Сбор информации из паспортов и ввод их в целевую базу данных нуждается в автоматизации обнаружения и распознавания структуры таблиц.

Опыт реализации решений анализа паспортов безопасности химической продукции показал, что доступные инструменты распознавания PDF-таблиц малоэффективны. Для того чтобы обеспечить более высокое качество работы реализуемых решений, потребовалось разработать специализированное ПО, опирающееся на специфику представления такой информации. Исходный код TabbyPDF был положен в основу специализированного ПО анализа паспортов. В частности, реализация метода анализа компоновки документа, предложенного в главе 2, позволила собрать блоки текста (заголовки разделов,

абзацы текста, ячейки таблиц и др.). При этом анализ компоновки документа был дополнен обнаружением заголовков 16 стандартных разделов паспортов и их подразделов, а также верхних и нижних колонтитулов. В результате страница документа сегментируется на области поиска, внутри которых выполняется обнаружение таблиц. Каждая область поиска идентифицируется некоторым разделом. В том случае, когда несколько областей поиска принадлежат одному разделу, они объединяются в одну (выполняется отрисовка соответствующих фрагментов страниц исходного документа в общую целевую страницу). Распознавание самой таблицы внутри области поиска базируется на методе, изложенном в главе 2. На рис. 6.1 показан пример распознавания таблицы, размещенной на двух соседних страницах, но при этом в одной области поиска исходного паспорта.

6.2. Анализ финансовых документов

Данное применение состояло в автоматизации процесса извлечения данных из документных таблиц, представленных в отчетах *оценки кредитного риска*. Кредитная организация (банк) анализирует такие отчеты, чтобы оценить кредитоспособность потенциальных корпоративных заемщиков. Существенная часть фактов, необходимых для принятия решения о кредитовании, представлены в таблицах. Исходные отчеты предоставляются в виде изображений документов. С помощью стороннего ПО оптического распознавания символов они переводятся в редактируемые документы формата Word, при этом восстанавливается физическая структура таблиц. Однако извлечение фактов нуждается также и в логической структуре таблиц.

Рассматривалась задача наполнения целевой базы фактов результатами сопоставления ключевых слов с именованными сущностями, упоминаемыми в таблицах отчетов оценки кредитного риска (рис. 6.2). Искомые клю-

Title	Name	Age	Stake	Years in industry	Guarantee obtained (Y/N)	PrB/PvB Customer? (Y/N)
Parthner	Mr. Apurva Shah	40+	70%	20+	N	N
Parthner	Mr. Ashit Shah	35+	30%	20+	N	N

Рисунок 6.2 — Анализ финансовых документов: исходная таблица из отчета оценки кредитного риска, в которой первый столбец содержит ключевое слово «*parthner*», а второй — именованные сущности «*Apurva Shah*» и «*Ashit Shah*»

Ключевое слово	Сущность	Тип	Провенанс
subsidiar	China Consolidated Company Ltd.	ORGANIZATION	10
client	Axis Bank	ORGANIZATION	7
partner	OWNERSVinati Saraf	PERSON	11
management	Social Risk Management	ORGANIZATION	2
md	Vinod Saraf.Mr	PERSON	11
competitor	Co Borrower	ORGANIZATION	6
director	Praveena Mehta	PERSON	8
director	Aruna Mukesh Mehta	PERSON	8
country	Sudan	LOCATION	1
country	India	LOCATION	3

Рисунок 6.3 — Анализ финансовых документов: наполнение целевой базы фактов результатами извлеченных именованных сущностей

чевые слова перечислены в словаре, причем каждое из них сопоставлено некоторому из заданных классов именованных сущностей: организаций, персон, местоположений, продуктов, количеств и др. Например, ключевые слова: «*subsidiar*», «*ceo*» и «*plant*», могут быть сопоставлены заданным классам следующим образом: `<subsidiar, organization>`, `<ceo, person>` и `<plant, location>`. Необходимо найти пары вида *<ключевое слово, именованная сущность>*, где оба относятся к одному классу именованных сущностей, например, `<subsidiar, Ecu-Line Mediterranean Ltd.>`, `<ceo, C. J. Shah>` и `<plant, USFDA>`. Предполагается, что ключевое слово и именованная сущность составляют такую пару, если соответствующие им ячейки связаны между собой некоторым логическим отношением.

Реализация решения поставленной задачи включает ПО извлечения данных из таблиц, которое адаптировало отдельные части исходного кода ПО

TabbyXL для восстановления связей между ячейками. При этом распознавание именованных сущностей выполняется с помощью моделей обработки естественного языка — OpenNLP[2] и Stanford NLP[3]. Восстановленная логическая структура позволяет выбрать пары ячеек по заданному ключевому слову. Результаты сопоставления ключевых слов с распознанными именованными сущностями сохраняются в целевой базе фактов (рис. 6.3).

6.3. Анализ научной литературы

Данная прикладная задача состоит в разработке решений распознавания таблиц, ориентированных на анализ научной литературы формата PDF. Группа исследователей (Z. Chi и др.) из Пекинского технологического университета разработала новую архитектуру ИНС для распознавания структуры таблицы — GraphTSR [128]. В качестве входных данных GraphTSR ИНС принимает граф, составленный из блоков текста, извлекаемых из PDF-документа с помощью ПО TabbyPDF. ПО TabbyPDF также включено в качестве одного из базовых решений наряду с GraphTSR, DeepDeSRT и др. в предметно-ориентированный тест производительности решений распознавания таблиц — SciTSR. Предлагаемая ими коллекция примеров собрана из материалов, опубликованных в репозитории arXiv[4]. Каждый тестовый пример сопровождается набором блоков текста, извлеченных из источников с помощью ПО TabbyPDF, в формате JSON; фрагмент такого представления блоков текста показан ниже:

```
{"chunks": [{"pos": [147.96600341796875, 205.49998474121094,
475.7929992675781, 480.4206237792969], "text": "Probability"},...]}
```

Другая группа исследователей (T. Adams и др.) из «Fraunhofer institute for algorithms and scientific computing SCAI» (Германия) рассматривает задачу интеллектуального анализа биомедицинской литературы в области невро-

[2] https://opennlp.apache.org

[3] https://nlp.stanford.edu

[4] https://arxiv.org

логических расстройств в рамках проекта Human Brain Pharmacome[5] (HBP) [75]. Последний нацелен на поиск эффективной профилактической терапии болезни Альцгеймера. Строится механистическая модель, представляющая химические соединения, действующие на мозг человека, их молекулярные мишени и механизмы действия. Она наполняется данными, извлекаемыми из тематических научных статей открытого архива PubMed Central[6] (PMC).

T. Adams и др. [75] указывают, что в доступных источниках, включая сопроводительные материалы, необходимые количественные данные в основном предоставляются в виде таблиц. Сами источники — PDF-документы, часто неразмеченные. Большой массив исходной неструктурированной информации, представляющей интерес для исследователей проекта HBP (только PMC содержит более 7 миллионов статей и 3,5 миллионов таблиц оценочно) делает практически невозможным ее приведение к структурированному формату вручную. Для того чтобы решить поставленную задачу, прежде всего требуется автоматизировать извлечение таблиц из таких источников.

С целью поддержки развития специализированных инструментов распознавания таблиц в PDF-документах архива PMC, направленных на интеллектуальный анализ биомедицинской литературы в области неврологических расстройств, авторы работы [75] реализовали тест производительности[7], где эталонные данные основаны на тематических статьях из архива PMC. Они задействовали ПО TabbyPDF в качестве одного из двух базовых решений. В результате обеспечивается количественное сравнение соответствующих им аналогов.

[5] https://www.scai.fraunhofer.de/en/projects/Human-Brain-Pharmacome.html

[6] https://www.ncbi.nlm.nih.gov/pmc

[7] https://github.com/mnamysl/benchmarking_table_recogn

6.4. Интеграция данных государственной статистики

Предлагаемое решение анализа и интерпретации таблиц TabbyXL использовалось в следующих приложениях: создании статистического атласа (раздел 6.4.1), наполнении базы данных статистических показателей (раздел 6.4.2), сборе бизнес-данных (раздел 6.5). Во всех перечисленных случаях требовалось автоматически восстанавливать логическую структуру исходных таблиц и приводить ее к некоторой конической форме. Рассмотрим случаи применения разработанного ПО ниже.

6.4.1. Создание тематических слоев электронной карты

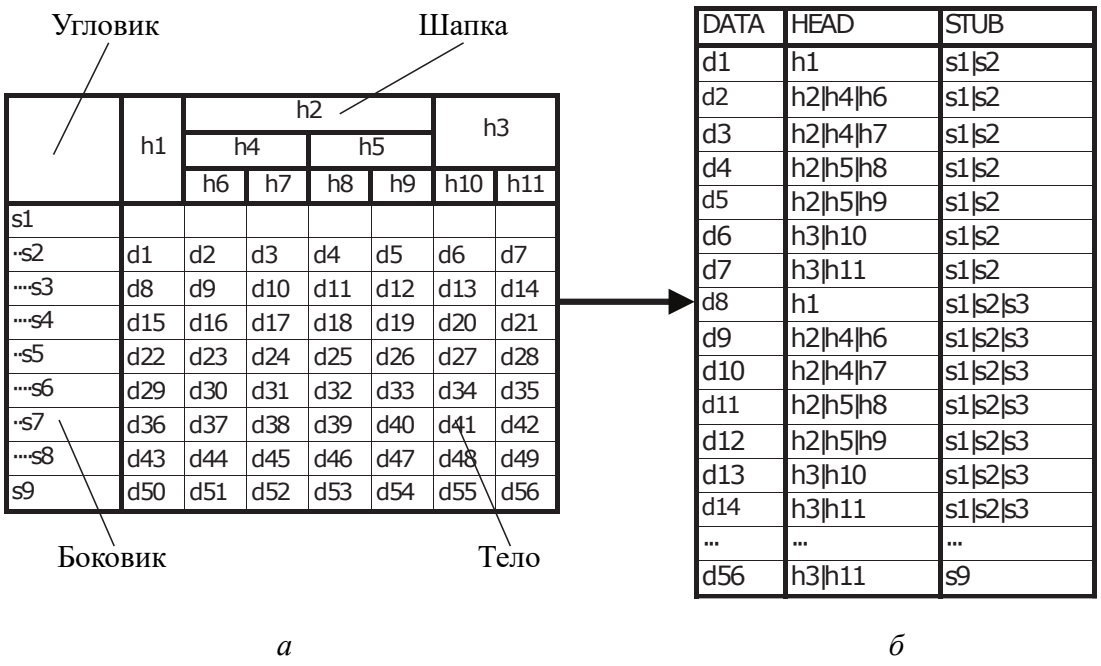
Данное приложение нацелено на создание тематических слоев электронной карты Иркутской области с помощью табличных данных статистических отчетов, публикуемых «Территориальным органом Федеральной службы государственной статистики по Иркутской области[8]» (Иркутскстатом). Исходные данные — значения статистических показателей в разрезе пространства (районов Иркутской области) и времени (месяцев, кварталов и годов); распространяются в формате Office Open XML[9]. Тематические слои электронной карты представляются в целевой базе данных геоинформационной системы Геопортал[10], развиваемой в ИДСТУ СО РАН. Для того чтобы автоматизировать ввод необходимой статистической информации в целевую базу данных, предлагается разработать ПО трансформации исходных таблиц (рис. 6.4, *а*) к канонической форме (рис. 6.4, *б*) на основе TabbyXL.

В качестве решения поставленной задачи предложен набор CRL-правил (приложение В.1). С помощью TabbyXL из них сгенерирован исходный код ПО трансформации исходных таблиц в наборы записей канонической формы.

[8] `https://38.rosstat.gov.ru`

[9] `https://www.ecma-international.org/publications-and-standards/standards/ecma-376`

[10] `http://cris.icc.ru`



а *б*

Рисунок 6.4 — Создание тематических слоев электронной карты: компоновка типичной таблицы отчетов Иркутскстата (*а*) и соответствующая каноническая форма (*б*)

Последние загружаются в целевое представление с помощью стандартных средств материализованной интеграции данных.

Тестирование предлагаемого решения выполнено на четырех отчетах Иркутскстата по охране атмосферного воздуха и населения Иркутской области. Всего обработано 450 таблиц, из них автоматически извлечено 51414 ячеек, 29050 вхождений, 12274 меток, 58094 пар «вхождение–метка», 2824 пар «метка–метка» и 12274 пар «метка–категория». Выполнена демонстрационная загрузка извлекаемых значений статистических показателей в целевую базу данных с геокодированием наименований районов Иркутской области. Визуализация полученных тематических слоев электронной карты реализована с помощью Геопортала[11] (рис. 6.5).

В результате в интересах правительства Иркутской области было автоматизировано создание тематических слоев электронной карты администра-

[11] `http://cris.icc.ru`

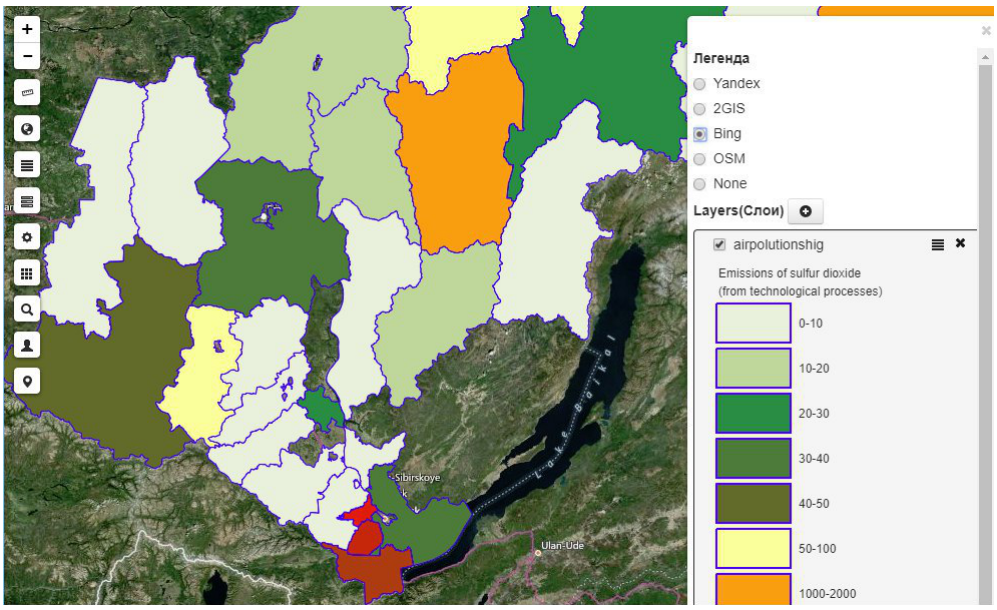


Рисунок 6.5 — Создание тематических слоев электронной карты: снимок экрана Геопортала с электронной карты Иркутской области (демонстрируемый тематический слой сформирован из данных, извлеченных из отчетов Иркутскстата)

тивно-территориального деления Иркутской области. (Работа была поддержана РФФИ, грант № 17-47-380007.)

6.4.2. Наполнение базы данных информационно-аналитической системы

В ИДСТУ СО РАН под руководством А. Е. Хмельнова развивается инструментальная платформа MDAttr для создания информационно-аналитических систем (далее ИАС) с функциями OLAP[12]. ИАС, разработанные на основе MDAttr, могут наполняться данными, которые предварительно извлекаются из статистических отчетов. В настоящей работе рассматривается один из таких случаев. Ставится задача наполнения базы данных ИАС Института национального развития Монголии статистическими сведениями по социально-экономическому положению территорий Монголии. Источником исходных данных выступают предоставленные этой организацией РК (рис. 6.6).

[12]Оперативная аналитическая обработка (англ. Online Analytical Processing).

	A	B	C	D	E	F	G	H	I	J	K
1											
2									**$NAME**	Table 1. …	
3	**$START**								**$MEASURE**	bln.tog	
4			**2003**				…	**2011**			
5			*Total, bln.tog*	*Share to total, %*				*Total, bln.tog*			
6				*Agriculture*	…	*Services*	…		…	*Services*	
7		***TOTAL***	**1479,7**	**20,6**		**54**		**10829689,6**		**52**	
8		***Western regi***	**118,5**	**63,5**		**32,2**		**455942,3**		**41,4**	
9		*Bayan-Olgii*	27,3	59,3		36,7		91968,9		51,9	
10		*Govi-Altai*	14,6	62		39,1		60036,6		49,4	
11		…					…			…	
12		*Khentii*	25,9	74,8		20,3		99575,3		29,1	
13		***Ulaanbaatar***	**861,5**	**0,8**	…	**73**	…	**6991314,8**	…	**68,8**	
14											**$END**

Рисунок 6.6 — Наполнение базы данных информационно-аналитической системы: исходная таблица со статистическими сведениями по социально-экономическому положению территорий Монголии; используются теги разметки: `$START` и `$END` — границы таблицы внутри рабочего листа; `$NAME` — наименование показателя и `$MEASURE` — единица измерения табличных данных

В качестве решения поставленной задачи предложен набор CRL-правил (приложение В.2). Дополнительно с помощью словарей регулярных выражений описаны диапазоны значений следующих категорий: *время* — 1990, 1991, и др.; *аймаки* — Bayan-Olgii, Govi-Altai и др.; *отрасли* — Agriculture, Industry и др.; *агрегации* — Total, Average и др. Значения двух других категорий — *наименований показателей* и *единиц измерений* — считываются из заголовков таблиц. Предполагается, что в процессе интерпретации таблицы каждая ее метка должна быть ассоциирована с одной из перечисленных категорий.

Заголовки исходных таблиц предварительно размечаются следующими тегами: `$NAME` указывает наименование показателя `$MEASURE` — единицу измерения табличных данных. С помощью данной разметки формируются диапазоны значения двух категорий — *наименований показателей* и *единиц измерений*. Диапазоны значений остальных категорий загружаются из словарей в формате YAML[13]. Полученные категории добавляются в рабочую память

[13] `https://yaml.org`

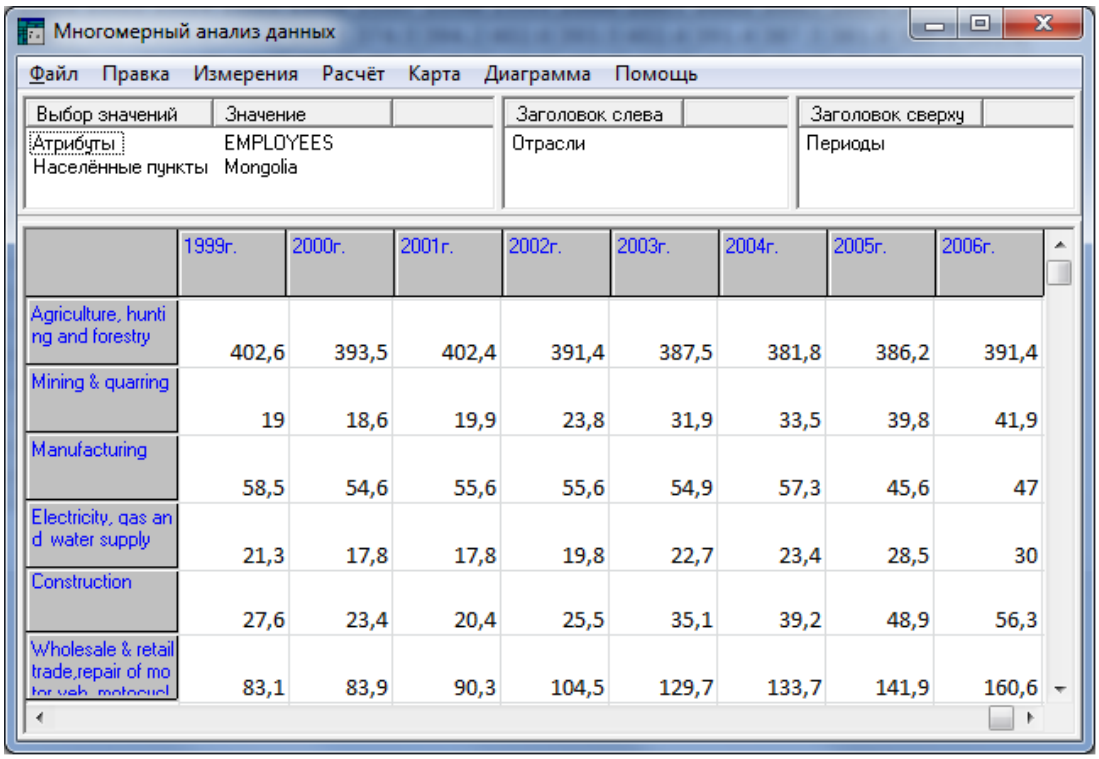


Рисунок 6.7 — Наполнение базы данных информационно-аналитической системы: снимок экрана MDAttr со статистическими сведениями по социально-экономическому положению территорий Монголии, загруженными из таблиц РК с помощью TabbyXL

системы исполнения правил в качестве начальных фактов. TabbyXL выполняет трансформацию исходных таблиц в наборы записей канонической формы, управляемую представленными правилами (приложение В.2). При этом данные, связанные с категорией агрегаций, не включаются в формируемые результаты. Полученные наборы записей загружаются в целевое представление с помощью стандартных средств материализованной интеграции данных.

В рамках международного сотрудничества с Академией наук Монголии статистическими сведениями наполнена база данных ИАС Института национального развития Монголии. Всего в результате исполнения представленных правил было автоматически извлечено более 15000 не агрегированных значений данных (вхождений) из 40 таблиц в формате Excel. Каждое из них было ассоциировано с пятью метками в пяти категориях: время, аймаки, отрасли, наименования показателей и единицы измерений. В результате наполнения

целевой базы данных ИАС (рис. 6.7) был обеспечен многомерный анализ социально-экономического положения территорий Монголии предметными специалистами. (Работа была поддержана РФФИ, грант № 16-57-44034.)

6.5. Интеграция данных медиапланирования

Сложившаяся практика медиапланирования рекламных кампаний широко применяет РК для представления так называемых медиапланов — расписаний показов рекламных материалов, в которых указываются расценки на размещение, даты выхода, форматы, продолжительность размещения, сроки подачи, ожидаемые результаты и пр. У крупного рекламодателя обычно имеется большое количество подрядчиков, размещающих рекламу в различных каналах коммуникации. Перед каждым из этих подрядчиков ставится план по бюджету и достигаемым целевым показателям. Они используют разные табличные шаблоны представления медиапланов, рассчитанные на удобство заполнения и последующий анализ.

Специалисты по медиапланированию сталкиваются с необходимостью сбора данных по планируемым затратам на рекламные размещения из медиапланов, соответствующих разным табличным шаблонам, в единую ИАС, с помощью которой выполняется оценка эффективности рекламных кампаний, в частности, сопоставление плановых и фактических показателей. Исходные медиапланы предоставляются в форматах РК Excel/Sheets. Один рабочий лист может включать несколько таблиц, которые часто имеют объединенные ячейки, изменяющийся от месяца к месяцу состав столбцов, строки с итогами. Их структура не позволяет загрузить их напрямую в ИАС с помощью стандартных средств интеграции данных. Тем не менее они могут предварительно приводиться к канонической форме с помощью CRL-правил, а затем

уже загружаться в ИАС. Решение поставленной задачи требует подготовки набора CRL-правил для каждого из имеющихся шаблонов.

В результате применения ПО TabbyXL удалось автоматизировать сбор данных медиапланов в маркетинговом агентстве «Адвентум Консалтинг»[14]. Данной организацией разработано веб-ориентированное ПО управления запусками и результатами сбора данных. Реализована следующая функциональность: хранение CRL-правил в едином хранилище данных; поиск исходных РК Excel/Sheets для извлечения данных; запуск CRL-правил по расписанию; хранение результатов применения CRL-правил в едином хранилище данных; предоставление доступа к результатам применения CRL-правил; оповещение пользователей об ошибках применения CRL-правил.

6.6. Конструирование онтологии предметной области

Данное приложение рассматривается на примере автоматизации конструирования онтологии предметной области *экспертизы промышленной безопасности*. В соответствии с российским законодательством (ФЗ-116[15]), такая экспертиза необходима для реализации процедуры подтверждения соответствия состояния технического оборудования требованиям промышленной безопасности. Одним из источников данных при наполнении целевой онтологии служат таблицы, содержащиеся в отчетах экспертиз промышленной безопасности (рис. 6.8, *а*). Они описывают результаты технического диагностирования, расчет долговечности и остаточного ресурса и пр. Их оформление и содержимое определяются корпоративными стандартами предприятий, предоставляющих услуги экспертизы промышленной безопасности. Настоящее исследование ограничивается отчетами, предоставленными Иркутским

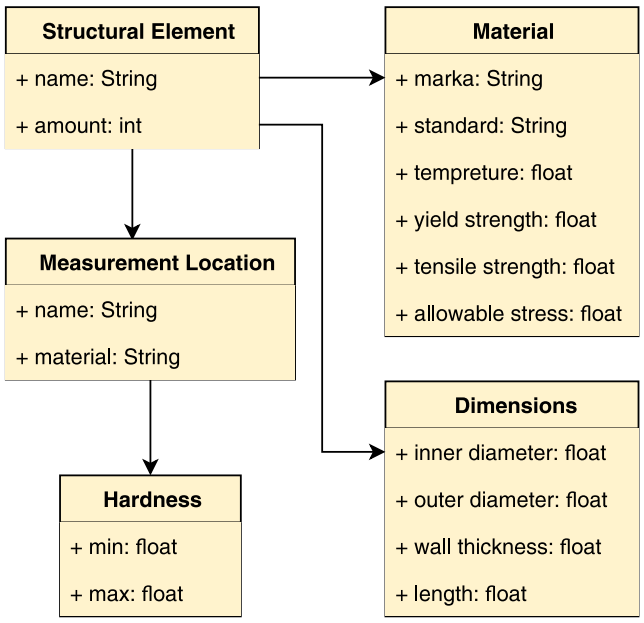
[14] `https://www.adventum.ru`

[15] `http://www.kremlin.ru/acts/bank/11232`

	A	B	C
1		Value	
2	Parameter	Actual	Design
3	Environment	Gas-oil mixture	Gas-oil mixture
4	Pressure, Mpa	32	32
5	Operating environment temperature, C	300	300

	A	B	C
1	Parameter	Denotation	Value
2	Operating pressure, Mpa	P	32
3	Design pressure, Mpa	Pd	32
4	Operating environment temperature, C	T	45
5	Design temperature of the wall, C	Td	50
6	Operating environment	Circulating gas	

а



б

Рисунок 6.8 — Конструирование онтологии предметной области: примеры исходных таблиц отчетов экспертизы промышленной безопасности (*а*); пример концептуальной модели, сгенерированной из 23 таблиц, описывающих инспектируемые объекты: испаритель, теплообменник и др. (*б*)

институтом химического и нефтехимического машиностроения[16] (Иркутск-НИИхиммаш).

Создание концептуальных моделей (рис. 6.8, *б*) выполняется с помощью системы управления базами знаний — Personal Knowledge Base Designer[17] (PKBD), развиваемой в ИДСТУ СО РАН под руководством А. Ю. Юрина. Данное решение обеспечивает визуальное моделирование баз знаний на основе правил конечными пользователями (предметными экспертами, аналитиками данных и др.). Для того чтобы воспользоваться табличными данными отчетов как одним из источников предметных знаний при построении концептуальных моделей, разработан специализированный модуль расширения PKBD. Обеспечивается интеграция на уровне данных с приложениями кано-

[16] `http://hm.irk.ru`

[17] `https://knowledge-core.ru`

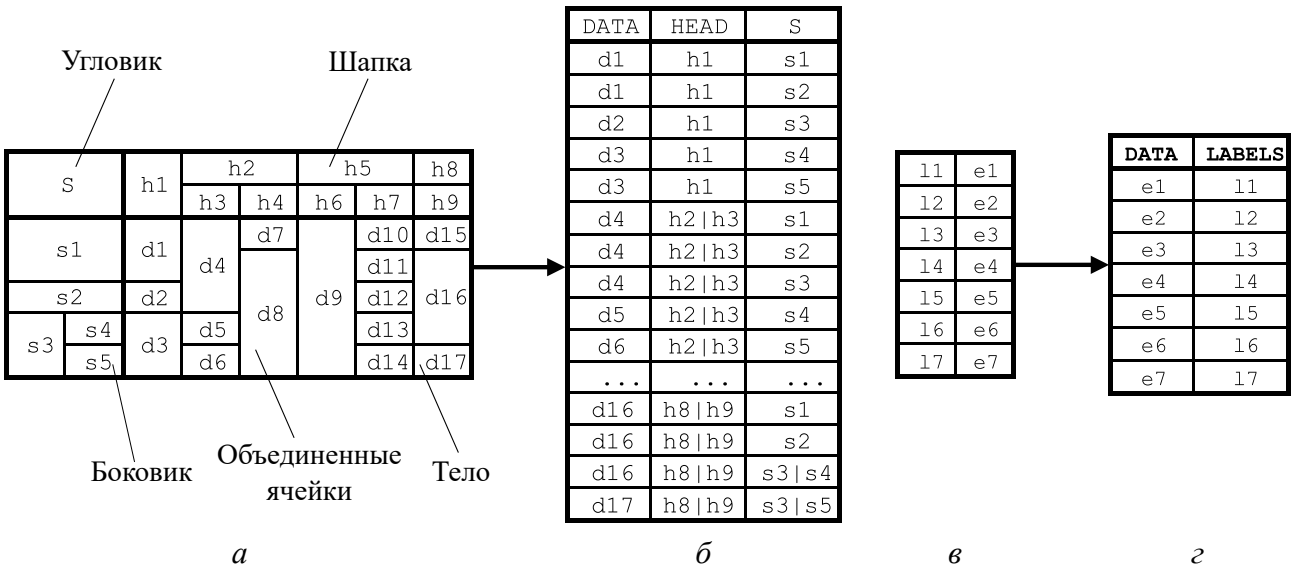


Рисунок 6.9 — Конструирование онтологии предметной области: таблица первого компоновочного типа (*а*) и ее каноническая форма (*б*); таблица второго компоновочного типа (*в*) и ее каноническая форма (*г*)

никализации данных из таблиц PK, разработанными на основе платформы TabbyXL. Модуль расширения генерирует фрагменты модели из упоминаний взаимосвязанных сущностей, извлеченных из исходных таблиц. Полученные фрагменты могут быть агрегированы в общую онтологию предметной области посредством операций их уточнения, слияния и разделения.

Исходные данные представлены в виде таблиц PK. Все таблицы относятся к одному из двух компоновочных типов (рис. 6.9, *а*, *в*). Для каждого из них предложен отдельный набор CRL-правил (приложение В.4). С помощью TabbyXL разработано ПО каноникализации данных таблиц PK, управляемой этими правилами. В результате запуска ПО формируются наборы записей в канонической форме (рис. 6.9, *б*, *г*). Из них генерируются фрагменты модели. Они комбинируются в единую концептуальную модель, которая загружается в базу знаний под управлением PKBD.

Для тестирования реализованного решения подготовлена коллекция примеров ISI-161[18], включающая 161 таблицу из шести отчетов ИркутскНИИ-

[18] https://doi.org/10.17632/8zdymg4y96.1

Химмаша. С помощью исполнения CRL-правил (приложение В.4) тестовые таблицы автоматически приводятся к канонической форме. Всего из них было извлечено 429 сущностей, включая 59 классов (понятий), 338 атрибутов (свойств) и 32 ассоциации (отношения). Агрегирование фрагментов в целевую модель сократила их до 242 сущностей, включая 25 классов, 196 атрибутов и 21 ассоциацию. Подробное описание и шаги воспроизведения данного эксперимента опубликованы в открытом доступе (приложение Г).

Полученные фрагменты сопоставлены предметными экспертами с эталонной моделью, предоставленной ИркутскНИИХиммашем. Установлено, что только 17 % (69 из 400) сущностей эталонной модели присутствуют также и в целевой. Следует отметить, что решение практических вопросов экспертизы промышленной безопасности требует дополнить целевую модель сущностями моделей описания динамики технических состояний. В рассматриваемом случае такое дополнение позволило увеличить совпадение целевой и эталонной модели до 24 % (106 из 400 сущностей). Эксперты подтвердили, что предложенное решение может применятся для конструирования начальной онтологии данной предметной области.

6.7. Кросс-контекстный обмен бизнес-документами

S. Yang и др. [35] рассматривают задачу организации кросс-контекстного обмена электронными бизнес-документами. Предполагается, что участники некоторой электронной торговой площадки создают бизнес-документы (опросные листы, счета на оплату, коммерческие предложения и др.) на основе собственных шаблонов. В таких документах таблицы используются для представления сведений об организациях, продуктах, условиях оплаты и пр.

Определим, что под контекстом понимается совокупность знаний, влияющих на корректное понимание смысла передаваемой информации. Слож-

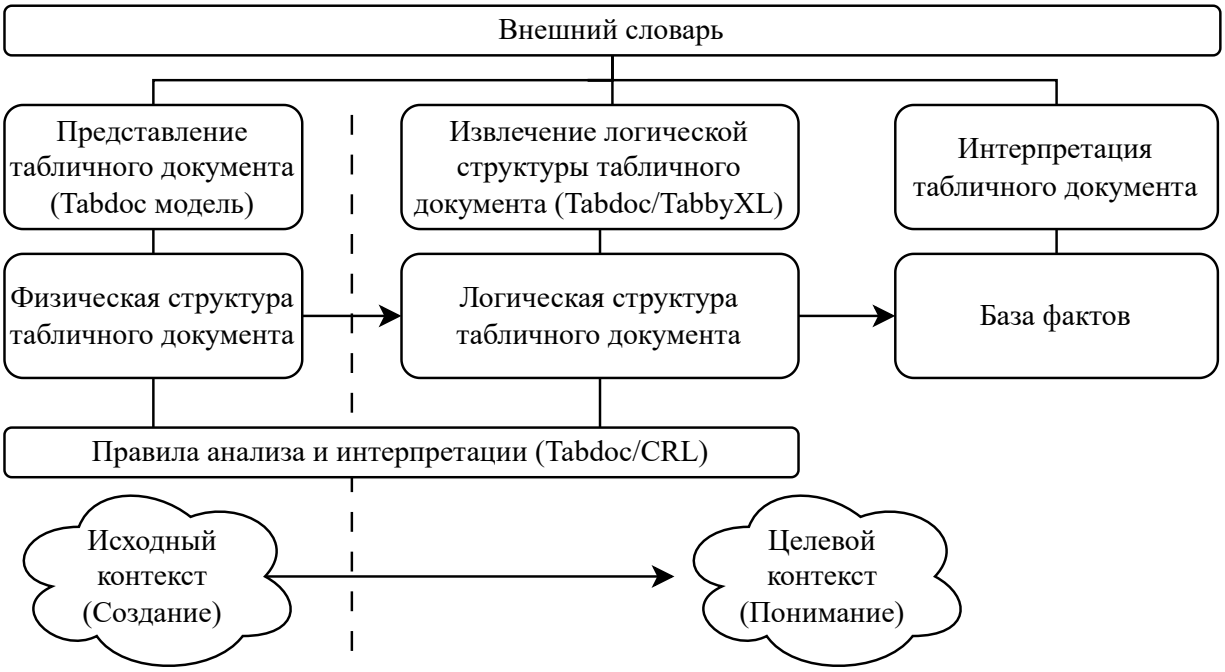


Рисунок 6.10 — Технология кросс-контекстного обмена табличными бизнес-документами Tabdoc

ность обмена бизнес-информацией связана с тем, что будучи подготовленной в одном контексте, она затем должна интерпретироваться в другом контексте. Например, понятие *FY2023* может означать фискальный год, начинающийся 1 января 2023 года, в одном контексте, и начинающийся 1 октября 2023 года — в другом. Участники обмена обладают автономностью в создании шаблонов. В результате применяются самые разнообразные формы представления бизнес-документов со сложной табличной структурой. В сложившейся практике они не сопровождаются явной семантикой, позволяющей интерпретировать их в соответствии с тем, как было задумано их авторами. Это ограничивает возможности их автоматической обработки при попадании в другие контексты; может приводить к неправильной трактовке содержащихся в них понятий.

S. Yang и др. [35] предлагают метод организации кросс-контекстного обмена бизнес-документами, называемую Tabdoc (рис. 6.10). Она обеспечивает согласованную интерпретацию бизнес-документов в гетерогенных кон-

текстах. Композиционно Tabdoc включает несколько компонентов, одним из которых является так называемый *метод семантического извлечения*. Данный компонент предназначен для последовательного выполнения двух задач: *анализа таблиц* — восстановление функциональных элементов таблицы (вхождений и меток) и связей между ними (пар типа «вхождение–метка» и «метка–метка»); *интерпретации таблиц* — создание групп функциональных элементов и определение семантических отношений между такими группами. Основное решение первой задачи адаптировало метод TabbyXL к управляемому правилами анализу и интерпретации таблиц.

Ввиду отсутствия подходящего набора эталонных данных, собранных из бизнес-документов, тестирование предлагаемого решения анализа таблиц выполнялось на коллекции Troy200, размеченной в рамках настоящего исследования [71]. Исследовались две альтернативные реализации: первая была предложена в рамках разработки метода Tabdoc, вторая основана на решении тестовой задачи Troy посредством TabbyXL. Тестирование показало в целом сопоставимые результаты обоих реализаций. Однако качество распознавания пар вида «метка–метка», демонстрируемое реализацией на базе TabbyXL, значительно лучше, чем у оригинальной реализации Tabdoc: F_1-мера составила 95,85 % у TabbyXL против 64,25 % у Tabdoc.

6.8. Выводы

Практическая применимость методов, предлагаемых в настоящей работе, показана на восьми прикладных задачах, пять из которых решались в научных и три в индустриальных проектах. Следует отметить, что из восьми проектов только три выполнялось в ИДСТУ СО РАН, а остальные — в сторонних организациях.

Перечислим научные проекты, выполненные в ИДСТУ СО РАН: реализовано конструирование онтологии предметной области из табличных данных отчетов экспертизы промышленной безопасности (РНФ № 18-71-10001); в интересах правительства Иркутской области реализовано создание тематических слоев электронной карты административно-территориального деления Иркутской области (РФФИ № 17-47-380007); в рамках международного сотрудничества с Академией наук Монголии статистическими данными наполнена база данных ИАС Института национального развития Монголии (РФФИ № 16-57-44034). Научные проекты сторонних организаций включают следующие. Группа исследователей (Z. Chi и др.) из Пекинского технологического университета (Пекин, Китай) разработала новую архитектуру ИНС для распознавания структуры таблицы — GraphTSR. В качестве входных данных GraphTSR ИНС принимает граф, составленный из блоков текста, извлекаемых из PDF-документа с помощью ПО TabbyPDF. Группа исследователей (T. Adams и др.) из Fraunhofer Institute for Algorithms and Scientific Computing SCAI[19] (Бонн, Германия) [75] включила TabbyPDF в качестве базового решения в свой тест производительности методов распознавания таблиц, представленных в биомедицинской литературе. Группа исследователей (S. Yang и др. [35]) из Гуанчжоуского университета (Гуанчжоу, Китай) предложила метод кросс-контекстного обмена табличными документами, адаптировавший предлагаемый в настоящей работе метод автоматизации анализа и интерпретации таблиц на основе исполнения правил.

Перечислим индустриальные проекты, применявшие результаты настоящей работы. Американская компания «CloudSDS Inc.»[20] использовала исходный код TabbyPDF при реализации ПО анализа технической документации — паспортов безопасности химической продукции. Индийская компания

[19] `https://www.scai.fraunhofer.de/en.html`

[20] `https://cloudsds.com`

«Mosaik Risk Solutions Inc.»[21] использовала исходный код TabbyXL при реализации ПО анализа бизнес-документов — отчетов оценки кредитного риска (раздел 6.2). Российское маркетинговое агентство «Адвентум Консалтинг»[22] использовало ПО TabbyXL для автоматизации сбора данных медиапланов, представленных в виде РК. Оно также разработало собственное ПО интеграции данных на базе ПО TabbyXL. Документы, подтверждающие использование разработанного ПО в перечисленных проектах, приведены в приложении Д.

Результат, представленный в данной главе

Описание случаев практического применения теоретических основ и комплекса инструментальных средств, предложенных в предыдущих главах, как в научных, так и в индустриальных проектах. Первые включают примеры реализации тестов производительности извлечения таблиц из научной литературы, создание тематических слоев электронной карты, наполнения база данных ИАС, конструирование онтологии предметной области, методологическое обеспечение кросс-контекстного обмена бизнес-документами. Вторые включают примеры реализации ПО анализа технических и финансовых документов, а именно паспортов безопасности химической продукции и отчетов оценки кредитного риска соответственно, а также автоматизации ввода данных из медиапланов.

Публикации

Результат, представленный в данной главе, опубликован в ряде рецензируемых изданий, в том числе в журналах [5,6,50,53] и материалах всероссийских и международных мероприятий [23, 25–27, 30, 32, 34–38, 40, 43].

[21] `http://www.mosaikrisk.com`

[22] `https://www.adventum.ru`

Заключение

В диссертационной работе созданы методы и комплекс инструментальных средств АПТ, совокупность которых составляет теоретические и технологические основы решения проблемы упрощения разработки прикладного программного обеспечения извлечения данных из документных таблиц за счет поддержки произвольности табличной структуры. Получены следующие **основные результаты**:

1. Усовершенствована структура совокупности задач АПТ [3], в которой согласована терминология, сложившаяся в родственных направлениях исследований: компьютерном зрении, управлении данными, информационном поиске и «Семантической паутине». По сравнению с имеющимися формулировками АПТ, она является более релевантной за счет согласованной терминологии и многоуровневой декомпозиции.
2. Предложен метод автоматизации распознавания таблиц НПОД [2,4,12], т. е. конвертирования их в редактируемый формат РК. Впервые данный процесс базируется на использовании PDL-специфичной информации НПОД. Показано, как ее можно задействовать для анализа компоновки страниц, обнаружения и сегментации таблиц НПОД.
3. Создана модель таблицы, обеспечивающая представление табличной информации в процессах АПТ [1,7]. В отличие от известных аналогов, она не ограничивает структуру таблицы предопределенными типами компоновки, атомарностью ячеек и плоскими заголовками.
4. Предложен метод автоматизации анализа и интерпретации таблиц редактируемого формата РК [1, 8–11]. Впервые данный процесс реализуется посредством пользовательских правил. Обеспечена поддержка произвольности структуры таблицы: свободной компоновки, структурированности ячеек и иерархичности заголовков.

5. Создан проблемно-ориентированный язык правил анализа и интерпретации таблиц [7]. В отличие от формальных языков общего назначения, он позволяет сфокусироваться исключительно на реализации логики соответствующих этапов АПТ.
6. Разработан комплекс инструментальных средств [5, 6, 13–18], реализующий предлагаемые теоретические основы. По сравнению с другими доступными инструментами АПТ он обеспечивает конвертирование таблиц НПОД в редактируемый формат РК с возможностью пользовательской коррекции результатов и извлечение наборов записей в канонической форме из полученных таблиц РК с помощью правил, предоставляемых пользователями.

Представленные в пятой главе экспериментальные результаты показали соответствие количественных оценок предлагаемых методов текущему мировому уровню по данной тематике (на разных тестах F_1-мера составила от 54,9 % до 92,1 % на этапе РТ и от 93,7 % до 98,5 % на этапе АТ). При этом качественно преодолеваются некоторые ограничения, присущие современному состоянию исследования, главным из которых является отсутствие в конкурентных методах поддержки свободной компоновки таблиц и структурированности ячеек.

Теоретические и технологические решения, разработанные в рамках диссертации, могут быть положены в основу создания новых информационных технологий и программных продуктов сбора и интеграции данных из документных таблиц. Сталкиваясь с задачами АПТ на практике, предметные специалисты и разработчики обычно прибегают к инструментальным средствам общего назначения, предлагая собственные реализации однотипных задач. В сравнении с последними, предлагаемые решения могут позволить сократить затраты на разработку целевого ПО.

Список сокращений и условных обозначений

АПТ — Автоматизированное понимание таблиц.

ГЗ — Граф знаний.

ДТ — Документная таблица.

ИАС — Информационно-аналитическая система.

ИНС — Искусственная нейронная сеть.

КИС — Комплекс инструментальных средств.

ОМТ — Объектная модель таблицы.

ПО — Программное обеспечение.

НПОД — Нередактируемый печатно-ориентированный документ.

ASCII — Американский стандартный код для обмена информацией.

CRL — Язык правил анализа и интерпретации таблиц.

CSV — Значения, разделенные запятыми.

HTML — Язык разметки гипертекста.

OCR — Оптическое распознавание символов.

OWL — Язык описания онтологий.

PDF — Переносимый формат документов.

PDL — Язык описания страниц (англ. Page Description Language).

RDF — Среда описания ресурса.

RTF — Обменный формат документов (англ. Rich Text Format).

SPARQL — Язык запросов к данным, представленным по модели RDF.

XML — Расширяемый язык разметки.

Список литературы

Публикации автора, в которых представлены основные научные результаты диссертационной работы

В журналах, рекомендованных ВАК

1. Шигаров А.О. Извлечение реляционных данных из произвольных таблиц электронных документов редактируемых форматов на основе пользовательских правил // *Вычислительные технологии*. 2025. Т. 30, № 3. С. 127–144.
2. Шигаров А.О. Распознавание таблиц неаннотированных PDF-документов на основе использования PDF-специфичных свойств // *Вычислительные технологии*. 2024. Т. 29, № 6. С. 125–146.
3. Shigarov A. Table understanding: problem overview // *WIREs Data Mining and Knowledge Discovery*. 2023. 13(1), e1482
4. Шигаров А.О., Парамонов В.В. Сегментация текста неразмеченных PDF-документов // *Вычислительные технологии*. 2022. Т. 27, № 5. С. 69–78.
5. Yurin A., Dorodnykh N., Shigarov A. Semi-automated formalization and representation of the engineering knowledge extracted from spreadsheet data // *IEEE Access*. 2021. 9. 157468–157481
6. Shigarov A., Khristyuk V., Mikhailov A. TabbyXL: Software platform for rule-based spreadsheet data extraction and transformation // *SoftwareX*. 2019. 10. 100270

7. Shigarov A., Mikhailov A. Rule-based spreadsheet data transformation from arbitrary to relational tables // *Information Systems*. 2017. 71. 123–136
8. Shigarov A. Table understanding using a rule engine // *Expert Systems with Applications*. 2015. 42(2), 929–937
9. Шигаров А.О., Бычков И.В., Парамонов В.В., Белых П.В. Анализ и интерпретация произвольных таблиц на основе исполнения CRL-правил // *Вычислительные технологии*. 2015. Т. 20, № 6. С. 87–112.
10. Шигаров А.О. Восстановление логической структуры таблиц из неструктурированных текстов на основе логического вывода // *Вычислительные технологии*. 2014. Т. 19, № 1. С. 87–99.
11. Шигаров А.О., Бычков И.В., Ружников Г.М. и др. Система трансформации таблиц // *Информационные технологии и вычислительные системы*. 2013. № 3. С. 15–26.
12. Shigarov A., Fedorov R. Simple algorithm page layout analysis // *Pattern Recognition and Image Analysis*. 2011. 21(2), 324–327.

Свидетельства о регистрации программ для ЭВМ

13. Шигаров А.О. TABBYXL: программный комплекс извлечения данных из таблиц рабочих книг форматов Excel/Sheets: Сви-о о гос. регистрации программы для ЭВМ № 2024682185 от 17.09.2024. М.: Роспатент, 2024.
14. Шигаров А.О., Парамонов В.В. HEADRECOG: программа распознавания структуры заголовков таблиц рабочих книг (модуль расширения программного комплекса TABBYXL): Сви-о о гос. регистрации программы для ЭВМ № 2024682187 от 17.09.2024. М.: Роспатент, 2024.
15. Шигаров А.О., Михайлов А.А. TABBYPDF: программный комплекс распознавания таблиц нередактируемых печатно-ориентированных доку-

ментов формата PDF: Сви-о о гос. регистрации программы для ЭВМ № 2024682186 от 18.09.2024. М.: Роспатент, 2024.

16. Бондарев А.И., Шигаров А.О. Веб-сервис каноникализации произвольных электронных таблиц (CELLS WebSSDC): Сви-о о гос. регистрации программы для ЭВМ № 2016614889 от 11.05.2016. М.: Роспатент, 2016.
17. Шигаров А.О., Парамонов В.В., Белых П.В. и др. CELLS spreadsheet unstructured tabular data transformation (SUTDT): Сви-о о гос. регистрации программы для ЭВМ № 2015661685 от 03.11.2015. М.: Роспатент, 2015.
18. Шигаров А.О., Михайлов А.А., Алтаев А.А., Бурлаков А.С. CELLS untagged PDF table extraction (UPDTE): Сви-о о гос. регистрации программы для ЭВМ № 2015662978 от 08.12.2015. М.: Роспатент, 2015.

Главы в монографиях

19. Инфраструктура информационных ресурсов и технологии создания информационно-аналитических систем территориального управления / И. В. Бычков [и др.]; под ред. И. В. Бычкова; Ин-т динамики систем и теории управления имени В. М. Матросова СО РАН. – Новосибирск: Изд-во СО РАН, 2016. – 242 с.
20. Интеграция информационно-аналитических ресурсов и обработка пространственных данных в задачах управления территориальным развитием / И. В. Бычков [и др.]; под ред. И. В. Бычкова; Ин-т динамики систем и теории управления имени СО РАН. – Новосибирск: Изд-во СО РАН, 2012. – 369 с.

В прочих изданиях

21. Kostyleva O., Paramonov V., Shigarov A., Vetrova V. Towards comparison of table type taxonomies // *Proc. 45th Int. ICT and Electronics Conv.* 2022. P. 1461–1465.
22. Shigarov A., Dorodnykh N., Yurin A. et al. From web-tables to a knowledge graph: prospects of an end-to-end solution // *Proc. 4th W. on Information Technologies: Algorithms, Models, Systems.* 2021. CEUR WS. V. 2984. P. 23–33.
23. Shigarov A., Dorodnykh N., Mikhailov A. et al. Table extraction, analysis, and interpretation: the current state of the TabbyDOC project // *Proc. 4th W. on Information Technologies: Algorithms, Models, Systems.* 2021. CEUR WS. V. 2984. P. 11–22.
24. Paramonov V., Shigarov A., Vetrova V. Rule driven spreadsheet data extraction from statistical tables: case study // *Proc. 27th Int. Conf. on Inf. and Softw. Technol.* 2021. CCIS 1486. P. 84–95.
25. Dorodnykh N., Shigarov A., Yurin A. Using the semantic annotation of web table data for knowledge base construction // *Proc. 4th Int. Conf. on Artificial Intelligence and Cloud Computing.* 2021. P. 122–129.
26. Dorodnykh N., Yurin A., Shigarov A., Turdakov D. Ontology engineering at the assertion level based on semantic annotation of tabular data // *Proc. 2021 Ivannikov Memorial Workshop.* 2021. P. 28–34.
27. Cherepanov I., Mikhailov A., Shigarov A., Paramonov V. On automated workflow for fine-tuning deep neural network models for table detection in document images // *Proc. 43rd Int. ICT and Electronics Conv.* 2020. P. 1130–1133.

28. Mikhailov A., Shigarov A., Rozhkov E., Cherepanov I. On graph-based verification for PDF table detection // *Proc. 2020 Ivannikov ISPRAS Open Conf.* 2020. P. 91–95.
29. Paramonov V., Shigarov A., Vetrova V. Table header correction algorithm based on heuristics for improving spreadsheet data extraction // *Proc. 26th Int. Conf. on Inf. and Softw. Technol.* 2020. CCIS 1283. P. 147–158.
30. Dorodnykh N., Yurin A., Shigarov A. Conceptual model engineering for industrial safety inspection based on spreadsheet data analysis // *Proc. 6th Int. Conf. on Model. and Develop. of Intelli. Sys.* 2020. CCIS 1126. P. 51–65.
31. Paramonov V., Shigarov A., Vetrova V., Mikhailov A. Heuristic algorithm for recovering a physical structure of spreadsheet header // *Proc. 40th Int. Conf. on Inf. Sys. Architect. and Technol.* 2019. AISC 1050. P. 140–149.
32. Shigarov A., Khristyuk V., Mikhailov A., Paramonov V. TabbyXL: rule-based spreadsheet data extraction and transformation // *Proc. 25th Int. Conf. on Inf. and Softw. Technol.* 2019. CCIS 1078. P. 59–75.
33. Shigarov A., Cherepanov I., Cherkashin E. et al. Towards end-to-end transformation of arbitrary tables from untagged portable documents (PDF) to linked data // *Proc. 2nd W. on Information Technologies: Algorithms, Models, Systems.* 2019. CEUR WS. V. 2463. P. 1–12.
34. Shigarov A., Khristyuk V., Mikhailov A., Paramonov V. Software development for rule-based spreadsheet data extraction and transformation // *Proc. 42nd Int. ICT and Electronics Conv.* 2019. P. 1132–1137.
35. Yang S., Wei R., Shigarov A. Semantic interoperability for electronic business through a novel cross-context semantic document exchange approach // *Proc. ACM Symp. on Document Engineering.* 2018. V. 28. P. 1–10.

36. Shigarov A., Altaev A., Mikhailov A. et al. TabbyPDF: Web-based system for PDF table extraction // *Proc. 24th Int. Conf. on Inf. and Softw. Technol.* 2018. CCIS 920. P. 257–269.

37. Shigarov A., Khristyuk V., Paramonov V. et al. Toward framework for development of spreadsheet data extraction systems // *Proc. 1st W. on Information Technologies: Algorithms, Models, Systems.* 2018. CEUR WS. V. 2221. P. 90–96.

38. Бычков И.В., Парамонов В.В., Шигаров А.О. и др. TabbyXL: система трансформации данных из произвольных электронных таблиц в реляционную форму // *Труды XVI Всеросс. конф. «Распределенные информационно-вычислительные ресурсы. Наука – цифровой экономике» (DICR).* 2017. Новосибирск, Россия. С. 150–156.

39. Парамонов В.В., Шигаров А.О. Идентификация вычисляемых значений в слабоструктурированных табличных документах // *Материалы межд. конф. «Информационные технологии и системы».* 2017. С. 256–257.

40. Шигаров А.О., Алтаев А.А., Михайлов А.А. TabbyPDF: Извлечение таблиц из неразмеченных PDF документов // *Материалы XVIII Всеросс. конф. молод. учен. по Математическому моделированию и информационным технологиям.* 2017. С. 96.

41. Shigarov A., Mikhailov A., Altaev A. Configurable table structure recognition in untagged PDF documents // *Proc. 2016 ACM Symp. on Document Engineering.* 2016. P. 119–122.

42. Shigarov A., Paramonov V., Belykh P., Bondarev A. Rule-based canonicalization of arbitrary tables in spreadsheets // *Proc. 22nd Int. Conf. on Inf. and Softw. Technol.* 2016. CCIS 639, P. 78–91.

43. *Shigarov A., Mikhailov A., Altaev A.* Web tool for heuristic table structure recognition in untagged PDF documents // *Proc. 18th Int. conf on Data Analytics and Management in Data Intensive Domains (DAMDID/RCDL).* 2016. P. 346–348.
44. Shigarov A., Paramonov V. CRL: a rule language for analysis and interpretation of arbitrary tables // *Selected Papers XVII Int. Conf. on Data Analytics and Management in Data Intensive Domains.* 2015. CEUR-WS. V. 1536. P. 22–29.
45. Шигаров А.О. CRL: Язык правил анализа и интерпретации таблиц // *Тезисы докладов III Росс.-Монг. конф. молод. уч. по Математическому моделированию, вычислительно-информационным технологиям и управлению.* 2015. С. 76.
46. Shigarov A., Bychkov I. From unstructured to structured tabular data using a rule engine // *Proc. 5th Int. W. on Comp. Sci. and Eng.* 2015. P. 85–91.
47. Shigarov A. Rule-based table analysis and interpretation // *Proc. 21st Int. Conf. on Inf. and Softw. Technol.* 2015. CCIS 538. P. 175–186.
48. Шигаров А.О. Автоматизированное понимание таблиц на основе системы исполнения правил // *Труды 16-й Всеросс. конф. «Электронные библиотеки: перспективные методы и технологии, электронные коллекции» (RCDL).* 2014. С. 383–390.
49. Шигаров А.О. Извлечение информации из таблиц с использованием системы исполнения правил // *Труды IV Всеросс. конф. «Математическое моделирование и вычислительно-информационные технологии в междисциплинарных научных исследованиях».* 2014. С. 73.
50. Шигаров А.О., Бычков И.В., Ружников Г.М. и др. Проект интеллектуальной системы извлечения табличной информации из неструктурированных текстов // *Вестн. Бурят. гос. ун-та. Математика, информатика.* 2013. № 9. С. 110–118.

51. *Шигаров А.О.* Анализ компоновки таблиц из источников неструктурированной текстовой информации // *Труды XVIII Всеросс. конф. «Информационные и математические технологии в науке и управлении»*. 2013. Т. 2. С. 39–46.

52. Шигаров А.О. Интеллектуальная система извлечения табличной информации из неструктурированных текстов // Тезисы докладов III Всеросс. конф. «Математическое моделирование и вычислительно-информационные технологии в междисциплинарных научных исследованиях». 2013. С. 63.

53. Ветров А.А., Фереферов Е.С., Хмельнов А.Е., Шигаров А.О. Формирование хранилища данных для систем MDAttr // *Вестн. Бурят. гос. ун-та. Математика, информатика.* 2012. № 9. С. 22–28.

54. Шигаров А.О. Логический анализ компоновки таблиц из слабоструктурированных источников с использованием экспертных знаний // *Тезисы докладов XIII Всеросс. конф. молод. учен. по математическому моделированию и информационным технологиям.* 2012. С. 54.

55. Шигаров А.О., Федоров Р.К. Алгоритм обнаружения пустого места на странице документа // *Современные технологии. Системный анализ. Моделирование*. 2011. № 3(31). С. 42–46.

56. Шигаров А.О., Хмельнов А.Е., Федоров Р.К. Преобразование слабоструктурированной табличной информации к реляционному представлению // *Тезисы докладов Всеросс. конф. «Математическое моделирование и вычислительно-информационные технологии в междисциплинарных научных исследованиях»*. 2011. С. 133.

57. Shigarov A.O., Hmelnov A.E. Analisis and segmentation of tables from electronic unstructured documents // *Zbornik radova konferencije «Matematičke i informacione tehnologije»*. 2010. P. 373–376.

58. Shigarov A., Fedorov R. An algorithm for page segmentation // *Proc. 10th Int. Conf. on Pattern Recognition and Image Analysis: New Information Technologies*. 2010. P. 351–354.

59. Shigarov A., Bychkov I., Ruzhnikov G., Khmel'nov A. A method of table detection in metafiles // *Pattern Recognition and Image Analysis*. 2009. 19(4), 693–697.

60. Бычков И.В., Ружников Г.М., Хмельнов А.Е., Шигаров А.О. Эвристический метод обнаружения таблиц в разноформатных документах // *Вычислительные технологии*. 2009. Т. 14, № 2. С. 58–73.

61. Шигаров А.О. Технология извлечения табличной информации из электронных документов разных форматов // *Современные технологии. Системный анализ. Моделирование*. 2009. № 3(23). С. 97–102.

62. Шигаров А.О. Автоматизированная система извлечения табличной информации из метафайлов // *Труды XIV Всеросс. конф. «Информационные и математические технологии в науке и управлении»*. 2009. Т. 2. С. 218–224.

63. Хмельнов А.Е., Шигаров А.О. Метод извлечения таблиц из неформатированного текста // *Вычислительные технологии*. 2008. Т. 13, спец. выпуск 1. С. 93–101.

64. Бычков И.В., Ружников Г.М., Хмельнов А.Е., Шигаров А.О. Метод обнаружения таблиц в метафайлах // *Современные технологии. Системный анализ. Моделирование*. 2008. Спецвыпуск. С. 47–51.

65. Bychkov I., Hmelnov A., Ruzhnikov G., Shigarov A. A method for table detection in metafiles // *Proc. 9th Int. Conf. on Pattern Recognition and Image Analysis: New Information Technologies*. 2008. V. 1. P. 66–69.

66. Хмельнов А.Е., Шигаров А.О. Сегментация страницы документа для обнаружения таблиц // *Труды XIII Всеросс. конф. «Информационные и математические технологии в науке и управлении»*. 2008. Т. 2. С. 244–251.
67. Шигаров А.О. Метод обнаружения таблиц в метафайлах // *Материалы Школы-семинара молод. учен. «Информационные технологии и моделирование социальных эколого-экономических систем»*. 2008. С. 58–61.
68. Хмельнов А.Е., Шигаров А.О. Метод извлечения статистических таблиц из неформатированного текста // *Труды XII Всеросс. конф. «Инф. и мат. технол. в науке и управлении»*. 2007. Т. 2. С. 91–99.
69. Хмельнов А.Е., Шигаров А.О. Извлечение таблиц из неформатированного текста // *Доклады XIII Всеросс. конф. «Математические методы распознавания образов»*. 2007. С. 551–553.
70. Хмельнов А.Е., Шигаров А.О. Извлечение статистических таблиц из неформатированного текста // *Материалы IX Школы-семинара «Мат. модел. и инф. технол.»*. 2007. С. 167–169.

Тесты производительности

71. Shigarov A. TabbyXL: dataset for the performance evaluation of a software platform for rule-based spreadsheet data extraction and transformation / Mendeley Data. 2019. URL: `https://doi.org/10.17632/ydcr7mcrtp.6`.
72. Shigarov A., Paramonov V., Khristyuk V. Spreadsheet data extraction from real-world tables of SAUS: case study / Figshare. 2021. URL: `https://doi.org/10.6084/m9.figshare.14371055.v2`.

Публикации других авторов

73. Abraham R., Erwig M. Header and unit inference for spreadsheets through spatial analyses // *Proc. IEEE S. on Vis. Lang. Hum. Cen. C.* 2004. P. 165–172.
74. Abraham R., Erwig M. UCheck: A spreadsheet type checker for end users // *J. Visual Lang. Comput.* 2007. V. 18, No. 1. P. 71–95.
75. Adams T., Namysl M., Kodamullil A.T. et al. Benchmarking table recognition performance on biomedical literature on neurological disorders // *Bioinformatics.* 2021. V. 38, No. 6. P. 1624–1630.
76. Adelfio M., Samet H. Schema extraction for tabular data on the web // *Proc. VLDB Endowment.* 2013. V. 6, No. 6. P. 421–432.
77. Adiga D., Bhat S.A., Shah M.B., Vyeth V. Table structure recognition based on cell relationship, a bottom-up approach // *Proc. Int. Conf. on Recent Advances in Natural Language Processing.* 2019. P. 1–8.
78. Agarwal M., Mondal A., Jawahar C.V. CDeC-Net: Composite deformable cascade network for table detection in document images // *Proc. 25th Int. Conf. on Pattern Recognition.* 2021. P. 9491–9498.
79. Andriyanov N., Dementiev V., Tashlinskiy A. Detection of objects in the images: from likelihood relationships towards scalable and efficient neural networks // *Computer Optics.* 2022. V. 46, No. 1. P. 139–159.
80. Arif S., Shafait F. Table detection in document images using foreground and background features // *Proc. Digital Image Computing: Techniques and Applications.* 2018. P. 1–8.

81. van Assem M., Rijgersberg H., Wigham M., Top J. Converting and annotating quantitative data tables // *Proc. 9th Int. Semantic Web Conf.: Part I*. 2010. LNCS 6496. P. 16–31.
82. Astrakhantsev N., Turdakov D., Vassilieva N. Semi-automatic data extraction from tables // *Selected Papers of the 15th All-Russian Scientific Conference on Digital Libraries: Advanced Methods and Technologies, Digital Collections*. 2013. P. 14–20.
83. Auer S., Dietzold S., Lehmann J. et al. Triplify: Light-weight linked data publication from relational databases // *Proc. 18th Int. Conf. on World Wide Web*. 2009. P. 621–630.
84. Badame S., Dig D. Refactoring meets spreadsheet formulas // *Proc. IEEE Int. Conf. on Software Maintenance*. 2012. P. 399–409.
85. Badaro G., Saeed M., Papotti P. Transformers for tabular data representation: a survey of models and applications // *Transactions of the Association for Computational Linguistics*. 2023. V. 11, P. 227–249.
86. Bai K., Mitra P., Giles C.L., Liu Y. Automatic extraction of table metadata from digital documents // *Proc. 6th ACM/IEEE-CS Joint Conf. on Digital Libraries*. 2006. P. 339–340.
87. Baimuratov I., Turygin D., Shilin I., Pliukhin D., Mouromtsev D. Extraction of requirement bases from domain normative documents and classifiers with application to the russian building code // *Lobachevskii Journal of Mathematics*. 2023. V. 44, No. 1. P. 97–110.
88. Balakrishnan S., Halevy A., Harb B. et al. Applying WebTables in practice // *Proc. 7th Conf. on Innovative Data Systems Research*. 2015.
89. Bansal A., Harit G., Roy S.D. Table extraction from document images using fixed point model // Proc. 2014 Indian Conf. on Computer Vision Graphics and Image Processing. 2014. V. 14. P. 67:1–67:8.

90. Barik T., Lubick K., Smith J. et al. Fuse: A reproducible, extendable, internet-scale corpus of spreadsheets // *Proc. IEEE/ACM 12th Working Conf. on Mining Software Repositories*. 2015. P. 486–489.
91. Barowy D., Gulwani S., Hart T., Zorn B. FlashRelate: Extracting relational data from semi-structured spreadsheets using examples // *Proc. 36th ACM SIGPLAN Conf. on Programming Language Design and Implementation*. 2015. P. 218–228.
92. Barowy Daniel W., Berger Emery D., Zorn Benjamin. ExceLint: Automatically finding spreadsheet formula errors // *Proc. ACM Program. Lang.* 2018. V. 2, No. OOPSLA. P. 148:1–148:26.
93. Bart E. Parsing tables by probabilistic modeling of perceptual cues // *Proc. 10th IAPR Int. W. on Document Analysis Systems*. 2012. P. 409–414.
94. Bhagavatula C.S., Noraset T., Downey D. Methods for exploring and mining tables on Wikipedia // *Proc. ACM SIGKDD Workshop on Interactive Data Exploration and Analytics*. 2013. P. 18–26.
95. Bhagavatula C.S., Noraset T., Downey D. TabEL: entity linking in web tables // *Proc. Semantic Web Conf.* 2015. LNCS 9366. P. 425–441.
96. Bhatt J., Hashmi K.A., Afzal M.Z., Stricker D. A survey of graphical page object detection with deep neural networks // *Applied Sciences*. 2021. V. 11, No. 12.
97. Binmakhashen G., Mahmoud S. Document layout analysis: a comprehensive survey // *ACM Computing Surveys*. 2019. V. 52, No. 6. P. 1–36.
98. Bonfitto S., Casiraghi E., Mesiti M. Table understanding approaches for extracting knowledge from heterogeneous tables // *WIREs Data Mining and Knowledge Discovery*. 2021. V. 11, No. 4. P. e1407.
99. Braunschweig K., Thiele M., Lehner W. From web tables to concepts: a semantic normalization approach // *Conceptual Modeling*. 2015. LNCS 9381. P. 247–260.

100. Braunschweig K. Recovering the semantics of tabular web data: Ph.D. thesis. Technischen Universität Dresden. 2015.

101. Breu H., Gil J., Kirkpatrick D., Werman M. Linear time Euclidean distance transform algorithms // *IEEE Transactions on Pattern Analysis and Machine Intelligence*. 1995. V. 17, No. 5. P. 529–533.

102. Van Breugel B., Van Der Schaar M. Position: Why tabular foundation models should be a research priority // Proc. 41st Int. Conf. on Machine Learning. 2024. V. 235. P. 48976–48993.

103. Broman K., Woo K. Data organization in spreadsheets // *The American Statistician*. 2018. V. 72, No. 1. P. 2–10.

104. Burdick D., Danilevsky M., Evfimievski A. et al. Table extraction and understanding for scientific and enterprise applications // *Proc. VLDB Endowment*. 2020. V. 13, No. 12. P. 3433–3436.

105. Cafarella M., Halevy A., Wang D.Z. et al. WebTables: Exploring the power of tables on the web // *Proc. VLDB Endowment*. 2008. V. 1, No. 1. P. 538–549.

106. Cafarella M., Halevy A., Wang D.Z. et al. Uncovering the relational web // *Proc. 11th Int. W. on Web and Databases*. 2008.

107. Cafarella M., Halevy A., Khoussainova N. Data integration for the relational web // *Proc. VLDB Endow.* 2009. V. 2, No. 1. P. 1090–1101.

108. Cafarella M., Halevy A., Madhavan J. Structured data on the Web // *Commun. ACM*. 2011. V. 54, No. 2. P. 72–79.

109. Cai Z., Vasconcelos N. Cascade R-CNN: High quality object detection and instance segmentation // *CoRR*. 2019. abs/1906.09756. URL: `http://arxiv.org/abs/1906.09756`.

110. Campbell-Kelly M., Croarken M., Flood R., Robson E. The History of Mathematical Tables / Oxford Univ. Press. 2003.

111. Casado-García Á., Domínguez C., Heras J. et al. The benefits of close-domain fine-tuning for table detection in document images // *Document Analysis Systems*. 2020. P. 199–215.

112. Cesarini F., Marinai S., Sarti L., Soda G. Trainable table location in document images // *Proc. Int. Conf. on Pattern Recognition*. 2002. V. 3. P. 236–240.

113. Chandran S., Kasturi R. Structural recognition of tabulated data // *Proc. 2nd Int. Conf. on Document Analysis and Recognition*. 1993. P. 516–519.

114. Che X., Yang H., Meinel C. Table detection from slide images // *Image and Video Technology*. 2016. LNCS 9431. P. 762–774.

115. Chen P.P.-S. The entity-relationship model – toward a unified view of data // *ACM Trans. Database Syst.* 1976. V. 1, No. 1. P. 9–36.

116. Chen H.-H., Tsai S.-C., Tsai J.-H. Mining tables from large scale HTML texts // *Proc. 18th Conf. on Computational Linguistics*. 2000. V. 1. P. 166–172.

117. Chen Z., Cafarella M. Automatic web spreadsheet data extraction // *Proc. 3rd Int. W. on Semantic Search Over the Web*. 2013. P. 1:1–1:8.

118. Chen Z., Cafarella M., Chen J. et al. Senbazuru: A prototype spreadsheet database management system // *Proc. VLDB Endow.* 2013. V. 6, No. 12. P. 1202–1205.

119. Chen Z., Cafarella M. Integrating spreadsheet data via accurate and low-effort extraction // *Proc. 20th ACM SIGKDD Int. Conf. on Knowledge Discovery and Data Mining*. 2014. P. 1126–1135.

120. Chen Z. Information extraction on para-relational data: Ph.D. thesis / University of Michigan. 2016.

121. Chen Z., Dadiomov S., Wesley R. et al. Spreadsheet property detection with rule-assisted active learning // *Proc. ACM Conf. on Information and Knowledge Management*. 2017. P. 999–1008.

122. Chen J., Jiménez-Ruiz E., Horrocks I., Sutton C. ColNet: Embedding the semantics of web tables for column type prediction // *Proc. 33rd AAAI Conf. on Artificial Intelligence and 31st Innovative Applications of Artificial Intelligence Conf. and 9th AAAI Symposium on Educational Advances in Artificial Intelligence*. 2019. P. 29–36.

123. Chen J., Jiménez-Ruiz E., Horrocks I., Sutton C. Learning semantic annotations for tabular data // *Proc. 28th Int. Joint Conf. on Artificial Intelligence*. 2019. P. 2088–2094.

124. Chen W., Chen J., Su Y. et al. Logical natural language generation from open-domain tables // *Proc. 58th Annual Meeting of the Association for Computational Linguistics*. 2020. P. 7929–7942.

125. Chen Z., Trabelsi M., Heflin J. et al. Table search using a deep contextualized language model // *Proc. 43rd Int. ACM SIGIR Conf. on Research and Development in Information Retrieval*. 2020. P. 589–598.

126. Chen X., Maniatis P., Singh R. et al. SpreadsheetCoder: Formula prediction from semi-structured context // *Proc. 38th Int. Conf. on Machine Learning*. 2021. PMLR 139. P. 1661–1672.

127. Chen L., Huang C., Zheng X., et al. TableVLM: Multi-modal pre-training for table structure recognition // *Proc. 61st Annual M. of the Association for Computational Linguistics*. 2023. V. 1. P. 2437–2449.

128. Chi Z., Huang H., Xu H.-D. et al. Complicated table structure recognition. *CoRR*. 2019. abs/1908.04729. URL: `https://arxiv.org/abs/1908.04729`.

129. Chu X., He Y., Chakrabarti K., Ganjam K. TEGRA: Table extraction by global record alignment // *Proc. 2015 ACM SIGMOD Int. Conf. on Management of Data*. 2015. P. 1713–1728.

130. Circi D., Khalighinejad G., Chen A. et al. How well do large language models understand tables in materials science? // *Integr Mater Manuf Innov*. 2024. V. 13. P. 669–687.

131. Clausner C., Antonacopoulos A., Pletschacher S. ICDAR2017 Competition on recognition of documents with complex layouts – RDCL2017 // *Proc. 14th IAPR Int. Conf. on Document Analysis and Recognition*. 2017. V. 1. P. 1404–1410.

132. Clausner C., Antonacopoulos A., Pletschacher S. ICDAR2019 Competition on recognition of documents with complex layouts – RDCL2019 // *Proc. 15th Int. Conf. Document Analysis and Recognition*. 2019. P. 1521–1526.

133. Cochez M., Ristoski P., Ponzetto S.P., Paulheim H. Global RDF vector space embeddings // *Proc. 16th Int. Semantic Web Conf.: Part I*. 2017. P. 190–207.

134. Codd E. F. A relational model of data for large shared data banks. *Commun. ACM*. 1970. V. 13, No. 6. P. 377–387.

135. Cohen W., Hurst M., Jensen L. A flexible learning system for wrapping tables and lists in HTML documents // *Proc. 11th Int. Conf. on World Wide Web*. 2002. P. 232–241.

136. Corrêa A.S., Zander P. Unleashing tabular content to open data // *Proc. 18th Int. Conf. on Digital Government Research*. 2017. P. 54–63.

137. Cortez E., Bernstein P.A., He Y., Novik L. Annotating database schemas to help enterprise search // *Proc. VLDB Endow.* 2015. V. 8, No. 12. P. 1936–1939.

138. Coüasnon B., Lemaitre A. Recognition of tables and forms // *Handbook of Document Image Processing and Recognition*. 2014. P. 647–677.

139. Cremaschi M., de Paoli F., Rula A., Spahiu B. A fully automated approach to a complete semantic table interpretation // *Future Generation Computer Systems*. 2020. V. 112. P. 478–500.

140. Crestan E., Pantel P. Web-scale table census and classification // *Proc. 4th ACM Int. Conf. on Web Search and Data Mining*. 2011. P. 545–554.

141. Cunha J., Saraiva J., Visser J. From spreadsheets to relational databases and back // *Proc. ACM SIGPLAN Workshop Partial Evaluation and Program Manipulation*. 2009. P. 179–188.

142. Cunha J., Fernandes J.P., Mendes J., Saraiva J. Embedding, evolution, and validation of model-driven spreadsheets // *IEEE Transactions on Software Engineering*. 2015. V. 41, No. 3. P. 241–263.

143. Cunha J., Erwig M., Mendes J., Saraiva J. Model inference for spreadsheets // *Automat. Softw. Eng.* 2016. V. 23, No. 3. P. 361–392.

144. Cutrona V., Bianchi F., Jiménez-Ruiz E., Palmonari M. Tough Tables: Carefully evaluating entity linking for tabular data // *Proc. 19th Int. Semantic Web Conf.: Part II*. 2020. LNCS 12507. P. 328–343.

145. Cutrona V., Chen J., Efthymiou V. et al. Results of SemTab 2021 // *Proc. Semantic Web Challenge on Tabular Data to Knowledge Graph Matching co-located with the 20th Int. Semantic Web Conf.* 2021. CEUR WS. V. 3103. P. 1–12.

146. Deckert F., Seidler B., Ebbecke M., Gillmann M. Table content understanding in SmartFIX // *Proc. 11th Int. Conf. on Document Analysis and Recognition*. 2011. P. 488–492.

147. Deng J., Dong W., Socher R. et al. ImageNet: A large-scale hierarchical image database // *Proc. 2009 IEEE Conf. on Computer Vision and Pattern Recognition*. 2009. P. 248–255.

148. Deng Y., Rosenberg D., Mann G. Challenges in end-to-end neural scientific table recognition // *2019 Int. Conf. on Document Analysis and Recognition*. 2019. P. 894–901.

149. Desai H., Kayal P., Singh M. TabLeX: A benchmark dataset for structure and content information extraction from scientific tables // *Proc. 16th Int. Conf. on Document Analysis and Recognition: Part II*. 2021. P. 554–569.

150. Dong X.L., Hajishirzi H., Lockard C., Shiralkar P. Multi-modal information extraction from text, semi-structured, and tabular data on the Web // *Proc. 26th ACM SIGKDD Int. Conf. on Knowledge Discovery & Data Mining*. 2020. P. 3543–3544.

151. Dong H., Liu S., Han S. et al. TableSense: Spreadsheet table detection with convolutional neural networks // *CoRR*. 2021. abs/2106.13500. URL: `https://arxiv.org/abs/2106.13500`.

152. Dong H., Cheng Z., He X., et al. Table pre-training: a survey on model architectures, pre-training objectives, and downstream tasks // *Proc. 31st Int. Joint Conf. on Artificial Intelligence*. 2022. P. 5426–5435.

153. Dong H., Wang Z. Large language models for tabular data: progresses and future directions // *Proc. 47th Int. ACM SIGIR Conf. on Research and Development in Information Retrieval*. 2024. P. 2997–3000.

154. Dorodnykh N., Yurin A. Towards a universal approach for semantic interpretation of spreadsheets data // Proc. 24th Int. S. on Database Engineering & Applications. 2020. P. 1–9.

155. Dorodnykh N., Yurin A. TabbyLD: a tool for semantic interpretation of spreadsheets data // *Proc. 7th Int. Conf. on Modelling and Development of Intelligent Systems*. 2021. CCIS 1341. P. 315–333.

156. Dou W., Xu C., Cheung S.C., Wei J. CACheck: Detecting and repairing cell arrays in spreadsheets // *IEEE Transactions on Software Engineering*. 2017. V. 43, No. 3. P. 226–251.

157. Douglas S., Hurst M., Quinn D. Using natural language processing for identifying and interpreting tables in plain text // *Proc. S. on Document Analysis and Information Retrieval*. 1995. P. 535–545.

158. Doush I.A., Pontelli E. Detecting and recognizing tables in spreadsheets // Proc. 9th IAPR Int. W. on Document Analysis Systems. 2010. P. 471–478.

159. Du L., Gao F., Chen X. et al. TabularNet: A neural network architecture for understanding semantic structures of tabular data // *Proc. 27th ACM SIGKDD Conf. on Knowledge Discovery & Data Mining*. 2021. P. 322–331.

160. Eberius J., Braunschweig K., Hentsch M. et al. Building the dresden web table corpus: a classification approach // *Proc. IEEE/ACM 2nd Int. S. on Big Data Computing*. 2015. P. 41–50.

161. Efthymiou V., Hassanzadeh O., Sadoghi M., Rodriguez-Muro M. Annotating web tables through ontology matching // *Proc. 11th Int. W. on Ontology Matching co-located with the 15th Int. Semantic Web Conf.* 2016. P. 229–230.

162. Efthymiou V., Hassanzadeh O., Rodriguez-Muro M., Christophides V. Matching web tables with knowledge base entities: from entity lookups to entity embeddings // *Proc. 16th Int. Semantic Web Conf.: Part I*. 2017. LNCS 10587. P. 260–277.

163. Elmeleegy H., Madhavan J., Halevy A. Harvesting relational tables from lists on the web // *The VLDB Journal*. 2011. V. 20, No. 2. P. 209–226.

164. Embley D.W., Hurst M., Lopresti D., Nagy G. Table-processing paradigms: a research survey // *Int. J. Doc. Anal. Recog.* 2006. V. 8, No. 2-3. P. 66–86.

165. Embley D.W., Seth S., Nagy G. Transforming web tables to a relational database // *Proc. 22nd Int. Conf. on Pattern Recognition*. 2014. P. 2781–2786.

166. Embley D.W., Krishnamoorthy M.S., Nagy G., Seth S. Converting heterogeneous statistical tables on the web to searchable databases // *Int. J. Doc. Anal. Recog.* 2016. V. 19, No. 2. P. 119–138.

167. Embley D.W. Relational model // *Encyclopedia of Database Systems*. 2016. P. 1–5.

168. Ermilov I., Ngomo A.-C.N. TAIPAN: Automatic property mapping for tabular data // *Knowledge Engineering and Knowledge Management*. 2016. LNCS 10024. P. 163–179.

169. Erwig M. Software engineering for spreadsheets // *IEEE Softw.* 2009. V. 26, No. 5. P. 25–30.

170. Everingham M., van Gool L., Williams C. et al. The Pascal visual object classes (VOC) challenge // *International Journal of Computer Vision.* 2010. V. 88, No. 2. P. 303–338.

171. Fabbri R., Costa L., Torelli J., Bruno O. 2D Euclidean distance transform algorithms: a comparative survey // *ACM Comput. Surv.* 2008. V. 40, No. 1.

172. Fang J., Gao L., Bai K. et al. A table detection method for multipage PDF documents via visual seperators and tabular structures // *Proc. 11th Int. Conf. on Document Analysis and Recognition.* 2011. P. 779–783.

173. Fang J., Tao X., Tang Z. et al. Dataset, ground-truth and performance metrics for table detection evaluation // *Proc. 10th IAPR Int. W. on Document Analysis Systems.* 2012. P. 445–449.

174. Fang J., Mitra P., Tang Z., Giles C.L. Table header detection and classification // *Proc. 26 AAAI Conf. on Artificial Intelligence.* 2012. V. 26, No. 1. P. 599–605.

175. Fayyaz N., Khusro S., Ullah S. Accessibility of tables in PDF documents // *Information Technology and Libraries.* 2021. V. 40, No. 3.

176. Fayyaz N., Khusro S., Imranuddin. Enhancing accessibility for the blind and visually impaired: presenting semantic information in PDF tables // *J. of King Saud Univ. – Comp. and Inf. Sci.*. 2023. V. 35, No. 7. P. 101617.

177. Fedorov P. E., Mironov A. V., Chernishev G. A. Russian web tables: A public corpus of web tables for Russian language based on Wikipedia // *Lobachevskii Journal of Mathematics.* 2023. V. 44, No. 1. P. 111–122.

178. Filby G. (Ed.). Spreadsheets in Science and Engineering. Springer. 1998.

179. Fumarola F., Weninger T., Barber R. et al. Extracting general lists from web documents: a hybrid approach // *Modern Approaches in Applied Intelligence.* 2011. P. 285–294.

180. Galkin M., Mouromtsev D., Auer S. Identifying web tables: supporting a neglected type of content on the web // *Knowledge Engineering and Semantic Web*. 2015. CCIS 518. P. 48–62.

181. Gao L., Yi X., Jiang Z. et al. ICDAR2017 competition on page object detection // *Proc. 14th IAPR Int. Conf. on Document Analysis and Recognition*. 2017. P. 1417–1422.

182. Gao L., Huang Y., Déjean H. et al. ICDAR 2019 competition on table detection and recognition (cTDaR) // *Proc. Int. Conf. on Document Analysis and Recognition*. 2019. P. 1510–1515.

183. Gatterbauer W., Bohunsky P. Table extraction using spatial reasoning on the CSS2 visual box model // *Proc. National Conference on Artificial Intelligence*. 2006. V. 2. P. 1313–1318.

184. Gatterbauer W., Bohunsky P., Herzog M. et al. Towards domain-independent information extraction from web tables // *Proc. 16th Int. Conf. on World Wide Web*. 2007. P. 71.

185. Ghasemi-Gol M., Pujara J., Szekely P. Tabular cell classification using pre-trained cell embeddings // *Proc. 2019 IEEE Int. Conf. on Data Mining*. 2019. P. 230–239.

186. Ghasemi-Gol M., Pujara J., Szekely P. Learning cell embeddings for understanding table layouts // *Knowledge and Information Systems*. 2021. V. 63, No. 1. P. 39–64.

187. Gilani A., Qasim S.R., Malik I., Shafait F. Table detection using deep learning // *Proc. 14th IAPR Int. Conf. on Document Analysis and Recognition*. 2017. V. 01. P. 771–776.

188. Göbel M., Hassan T., Oro E., Orsi G. A methodology for evaluating algorithms for table understanding in PDF documents // *Proc. ACM S. on Document Engineering*. 2012. P. 45–48.

189. Göbel M., Hassan T., Oro E., Orsi G. ICDAR 2013 table competition // *Proc. 12th Int. Conf. on Document Analysis and Recognition*. 2013. P. 1449–1453.

190. Green E., Krishnamoorthy M. Model-based analysis of printed tables // *Proc. 3rd Int. Conf. on Document Analysis and Recognition*. 1995. P. 214–217.

191. Gulwani S., Harris W., Singh R. Spreadsheet data manipulation using examples // *Commun. ACM*. 2012. V. 55, No. 8. P. 97–105.

192. Gupta A., Tiwari D., Khurana T., Das S. Table detection and metadata extraction in document images // *Smart Innovations in Communication and Computational Sciences*. 2019. AISC 851. P. 361–372.

193. Gurulingappa H., Mudi A., Toldo L. et al. Challenges in mining the literature for chemical information // *RSC Adv*. 2013. V. 3. P. 16194–16211.

194. Habibi M., Starlinger J., Leser U. DeepTable: a permutation invariant neural network for table orientation classification // *Data Mining and Knowledge Discovery*. 2020. V. 34, No. 6. P. 1963–1983.

195. Han L., Finin T., Parr C. et al. RDF123: From spreadsheets to RDF // *Proc. 7th Int. Semantic Web Conference*. 2008. LNCS 5318. P. 451–466.

196. Hancock B., Lee H., Yu C. Generating titles for web tables // *The World Wide Web Conf*. 2019. P. 638–647.

197. Handley J. Table analysis for multiline cell identification // *Document Recognition and Retrieval VIII*. 2000. V. 4307. P. 34–43.

198. Hao L., Gao L., Yi X., Tang Z. A table detection method for PDF documents based on convolutional neural networks // *Proc. 12th IAPR W. on Document Analysis Systems*. 2016. P. 287–292.

199. Harris W., Gulwani S. Spreadsheet table transformations from examples // *ACM SIGPLAN Notices*. 2011. V. 46, No. 6. P. 317–328.

200. Hashmi K.A., Liwicki M., Stricker D. et al. Current status and performance analysis of table recognition in document images with deep neural networks // *IEEE Access*. 2021. V. 9. P. 87663–87685.

201. Hashmi K.A., Stricker D., Liwicki M. et al. Guided table structure recognition through anchor optimization // *IEEE Access*. 2021. V. 9. P. 113521–113534.

202. Hashmi K.A., Pagani A., Liwicki M. et al. CasTabDetectoRS: Cascade network for table detection in document images with recursive feature pyramid and switchable atrous convolution // *Journal of Imaging*. 2021. V. 7, No. 10.

203. Hassan T., Baumgartner R. Table recognition and understanding from PDF files // *Proc. 9th Int. Conf. on Document Analysis and Recognition*. 2007. V. 2. P. 1143–1147.

204. He K., Zhang X., Ren S., Sun J. Deep residual learning for image recognition // *Proc. 2016 IEEE Conf. on Computer Vision and Pattern Recognition*. 2016. P. 770–778.

205. He D., Cohen S., Price B. et al. Multi-scale multi-task FCN for semantic page segmentation and table detection // *Proc. 14th IAPR Int. Conf. on Document Analysis and Recognition*. 2017. V. 01. P. 254–261.

206. Mask R-CNN / K. He, G. Gkioxari, P. Dollár, R. Girshick // *CoRR*. 2017. abs/1703.06870. URL: `http://arxiv.org/abs/1703.06870`.

207. Hermans F., Murphy-Hill E., Enron's spreadsheets and related emails: A dataset and analysis // *Proc. IEEE/ACM 37th IEEE Int. Conf. on Software Engineering*. 2015. V. 2. P. 7–16.

208. Hermans F., Pinzger M., van Deursen A. Detecting and refactoring code smells in spreadsheet formulas // *Empir. Softw. Eng.* 2015. V. 20, No. 2. P. 549–575.

209. Herzig J., Nowak P. K., Müller T., Piccinno F., Eisenschlos J. TaPas: Weakly supervised table parsing via pre-training // *Proc. 58th Annual M. of the Association for Computational Linguistics*. 2020. P. 4320–4333.

210. Hinterberger H. Table // *Encyclopedia of Database Systems*. 2018. P. 3873–3874.

211. Hoffswell J., Liu Z. Interactive repair of tables extracted from PDF documents on mobile devices // *Proc. CHI Conference on Human Factors in Computing Systems*. 2019. P. 293:1–293:13.

212. Hogan A., Blomqvist E., Cochez M. et al. Knowledge graphs // *ACM Comput. Surv.* 2021. V. 54, No. 4.

213. Holeček M., Hoskovec A., Baudiš P., Klinger P. Table understanding in structured documents // *2019 Int. Conf. on Document Analysis and Recognition Workshops*. 2019. V. 5. P. 158–164.

214. Hu J., Kashi R., Lopresti D., Wilfong G. Medium-independent table detection // *Proc. Document Recognition and Retrieval VII*. 1999. SPIE 3967. P. 291–302.

215. Hu J., Kashi R.S., Lopresti D., Wilfong G. Table structure recognition and its evaluation // *Proc. Document Recognition and Retrieval VIII*. 2000. SPIE 4307. P. 44–55.

216. Hu J., Kashi R.S., Lopresti D., Wilfong G.T. Evaluating the performance of table processing algorithms // *Int. J. Doc. Anal. Recog.* 2002. V. 4, No. 3. P. 140–153.

217. Huang Y., Yan Q., Li Y. et al. A YOLO-based table detection method // *Proc. 15th Int. Conf. on Document Analysis and Recognition*. 2019. P. 813–818.

218. Hulsebos M., Demiralp Ç., Groth P. GitTables: A large-scale corpus of relational tables // *CoRR*. 2021. abs/2106.07258. URL: `https://arxiv.org/abs/2106.07258`.

219. Hung V., Benatallah B., Saint-Paul R. Spreadsheet-based complex data transformation // *Proc. 20th ACM Int. Conf. on Information and Knowledge Management*. 2011. P. 1749–1754.

220. Hung V. Spreadsheet-based complex data transformation: Ph.D. thesis / University of New South Wales. 2011.

221. Hurst M. The interpretation of tables in texts: Ph.D. thesis / University of Edinburgh. 2000.

222. Hurst M. Layout and language: Challenges for table understanding on the web // *Proc. Int. W. on Web Document Analysis*. 2001. P. 27–30.

223. Hurst M. Towards a theory of tables // *Int. J. Doc. Anal. Recog.* 2006. V. 8, No. 2–3. P. 123–131.

224. Itonori K. Table structure recognition based on textblock arrangement and ruled line position // *Proc. 2nd Int. Conf. on Document Analysis and Recognition*. 1993. P. 765–768.

225. Jaekyu H., Haralick R.M., Phillips I.T. Recursive X-Y cut using bounding boxes of connected components // *Proc. 3rd Int. Conf. on Document Analysis and Recognition*. 1995. V. 2. P. 952–955.

226. Jauhar S.K., Turney P., Hovy E. Tables as semi-structured knowledge for question answering // *Proc. 54th Annual Meeting of the Association for Computational Linguistics*. 2016. V. 1. P. 474–483.

227. Jiménez-Ruiz E., Cuenca Grau B. LogMap: Logic-based and scalable ontology matching // *Proc. 10th Int. Semantic Web Conf.: Part II*. 2011. LNCS 7031. P. 273–288.

228. Jiménez-Ruiz E., Hassanzadeh O., Efthymiou V. et al. SemTab 2019: Resources to benchmark tabular data to knowledge graph matching systems // *The Semantic Web*. 2020. LNCS 12123. P. 514–530.

229. Jin Z., Anderson M., Cafarella M., Jagadish H.V. Foofah: Transforming data by example // Proc. ACM SIGMOD Int. Conf. on Management of Data. 2017. P. 683–698.

230. Jin N., Siebert J., Li D., Chen Q. A survey on table question answering: recent advances // Knowledge Graph and Semantic Computing: Knowledge Graph Empowers the Digital Economy. 2022. CCIS 1669. P. 174–186.

231. Jung S.-W., Kwon H.-C. A scalable hybrid approach for extracting head components from Web tables // *IEEE Transactions on Knowledge and Data Engineering*. 2006. V. 18, No. 2. P. 174–187.

232. Kandel S., Paepcke A., Hellerstein J., Heer J. Wrangler: Interactive visual specification of data transformation scripts // *Proc. SIGCHI Conf. on Human Factors in Computing Systems*. 2011. P. 3363–3372.

233. Kasar T., Barlas P., Adam S. et al. Learning to detect tables in scanned document images using line information // *Proc. 12th Int. Conf. on Document Analysis and Recognition*. 2013. P. 1185–1189.

234. Kavasidis I., Pino C., Palazzo S. et al. A saliency-based convolutional neural network for table and chart detection in digitized documents // *Image Analysis and Processing*. 2019. LNIP 11752. P. 292–302.

235. Khan S.A., Khalid S.M.D., Shahzad M.A., Shafait F. Table structure extraction with bi-directional gated recurrent unit networks // *Proc. 15th Int. Conf. on Document Analysis and Recognition*. 2019. P. 1366–1371.

236. Khusro S., Latif A., Ullah I. On methods and tools of table detection, extraction and annotation in PDF documents // *J. Inf. Sci.* 2015. V. 41, No. 1. P. 41–57.

237. Kieninger T. Table structure recognition based on robust block segmentation // *Document Recognition V*. 1998. P. 22–32.

238. Kieninger T., Dengel A. The T-Recs table recognition and analysis system // *Document Analysis Systems: Theory and Practice*. 1999. LNCS 1655. P. 255–270.

239. Kieninger T., Dengel A. Applying the T-Recs table recognition system to the business letter domain // *Proc. 6th Int. Conf. on Document Analysis and Recognition*. 2001. P. 518–522.

240. Kim Y.-S., Lee K.-H. Detecting tables in web documents // *Engineering Applications of Artificial Intelligence*. 2005. V. 18, No. 6. P. 745–757.

241. Kim Y.-S., Lee K.-H. Extracting logical structures from HTML tables // *Computer Standards & Interfaces*. 2008. V. 30, No. 5. P. 296–308.

242. Kim D.H., Hoque E., Kim J., Agrawala M. Facilitating document reading by linking text and tables // *Proc. 31st Annual ACM Symposium on User Interface Software and Technology*. 2018. P. 423–434.

243. Kim J., Hwang H. A rule-based method for table detection in website images // *IEEE Access*. 2020. V. 8. P. 81022–81033.

244. Klampfl S., Jack K., Kern R. A comparison of two unsupervised table recognition methods from digital scientific articles // *D-Lib Magazine*. 2014. V. 20, No. 11/12.

245. Klein B., Agne S., Bagdanov A. Understanding document analysis and understanding (through modeling) // *Proc. 7th Int. Conf. on Document Analysis and Recognition*. 2003. P. 1218–1222.

246. Koch P., Hofer B., Wotawa F. On the refinement of spreadsheet smells by means of structure information // *Journal of Systems and Software*. 2019. V. 147. P. 64–85.

247. Koci E., Thiele M., Romero O., Lehner W. A machine learning approach for layout inference in spreadsheets // *Proc. Int. Joint Conf. on Knowledge Discovery, Knowledge Engineering and Knowledge Management*. 2016. P. 77–88.

248. Koci E., Thiele M., Romero O., Lehner W. Table identification and reconstruction in spreadsheets // *Advanced Information Systems Engineering*. 2017. LNCS 10253. P. 527–541.

249. Koci E., Thiele M., Lehner W., Romero O. Table recognition in spreadsheets via a graph representation // Proc. 13th IAPR Int. W. on Document Analysis Systems. 2018. P. 139–144.

250. Koci E., Thiele M., Rehak J., Lehner W. DECO: A dataset of annotated spreadsheets for layout and table recognition // *Proc. 15th Int. Conf. Document Analysis and Recognition*. 2019. P. 1280–1285.

251. Koci E., Thiele M., Romero O., Lehner W. Cell classification for layout recognition in spreadsheets // *Knowledge Discovery, Knowledge Engineering and Knowledge Management*. 2019. CCIS 914. P. 78–100.

252. Koci E., Thiele M., Romero O., Lehner W. A genetic-based search for adaptive table recognition in spreadsheets // *Proc. 15th Int. Conf. on Document Analysis and Recognition*. 2019. P. 1274–1279.

253. Koci E., Kuban D., Luettig N. et al. XLIndy: Interactive recognition and information extraction in spreadsheets // *Proc. ACM Symposium on Document Engineering*. 2019. P. 1–4.

254. Kolb S., Paramonov S., Guns T., et al. Learning constraints in spreadsheets and tabular data // *Machine Learning*. 2017. V. 106, No. 9. P. 1441–1468.

255. Kolb S., Teso S., Dries A., de Raedt L. Predictive spreadsheet autocompletion with constraints // *Machine Learning*. 2020. V. 109, No. 2. P. 307–325.

256. Kononova O., He T., Huo H. et al. Opportunities and challenges of text mining in materials research // *iScience*. 2021. V. 24, No. 3. P. 102155.

257. Kovalenko O., Serral E., Biffl S. Towards evaluation and comparison of tools for ontology population from spreadsheet data // *Proc. 9th Int. Conf. on Semantic Systems*. 2013. P. 57–64.

258. Krizhevsky A., Sutskever I., Hinton G. ImageNet classification with deep convolutional neural networks // *Commun. ACM*. 2017. V. 60, No. 6. P. 84–90.

259. Kruit B., Boncz P., Urbani J. Extracting novel facts from tables for knowledge graph completion // *Proc. 18th Int. Semantic Web Conf.: Part II*. 2019. LNCS 11778. P. 364–381.

260. Krüpl B., Herzog M. Visually guided bottom-up table detection and segmentation in web documents // *Proc. 15th Int. Conf. on World Wide Web*. 2006. P. 933–934.

261. Kweon S., Kwon Y., Cho S., Jo Y., Choi E. Open-WikiTable: Dataset for open domain question answering with complex reasoning over table // *Findings of the Association for Computational Linguistics: ACL 2023*. 2023. P. 8285–8297.

262. Langegger A., Wöß W. XLWrap – querying and integrating arbitrary spreadsheets with SPARQL // *Proc. 8th Int. Semantic Web Conf.* 2009. LNCS 5823. P. 359–374.

263. Lautert L., Scheidt M., Dorneles C. Web table taxonomy and formalization // *ACM SIGMOD Record*. 2013. V. 42, No. 3. P. 28–33.

264. Lehmberg O., Ritze D., Meusel R., Bizer C. A large public corpus of web tables containing time and context metadata // *Proc. 25th Int. Conf. on World Wide Web*. 2016. P. 75–76.

265. Lehmberg O., Bizer C. Web table column categorisation and profiling // *Proc. 19th Int. W. on Web and Databases*. 2016.

266. Lerman K, Knoblock C, Minton S. Automatic data extraction from lists and tables in web sources // *Proc. W. on Adaptive Text Extraction and Mining*. 2001.

267. Lerman K., Getoor L., Minton S., Knoblock C. Using the structure of web sites for automatic segmentation of tables // *Proc. 2004 ACM SIGMOD Int. Conf. on Management of Data*. 2004. P. 119–130.

268. Li J., Tang J., Song Q., Xu P. Table detection from plain text using machine learning and document structure // *Frontiers of WWW Research and Development*. 2006. LNCS 3841. P. 818–823.

269. Li Y., Gao L., Tang Z. et al. A GAN-based feature generator for table detection // *Proc. 15th Int. Conf. Document Analysis and Recognition*. 2019. P. 763–768.

270. Li M., Cui L., Huang S. et al. TableBank: Table benchmark for image-based table detection and recognition // *Proc. 12th Language Resources and Evaluation Conf.* 2020. P. 1918–1925.

271. Li P., He Y., Yashar D. et al. Table-GPT: Table fine-tuned GPT for diverse table tasks // *Proc. ACM Manag. Data*. 2024. V. 2, No. 3. P. 176

272. Li M., Cui L., Huang S. et al. TableBank: Table benchmark for image-based table detection and recognition // *Proc. 12th Language Resources and Evaluation Conference*. 2020. P. 1918–1925.

273. Limaye G., Sarawagi S., Chakrabarti S. Annotating and searching web tables using entities, types and relationships // *Proc. VLDB Endowment*. 2010. V. 3, No. 1–2. P. 1338–1347.

274. Lin T.-Y., Maire M., Belongie S. et al. Microsoft COCO: Common objects in context // *Proc. 13th European Conf. on Computer Vision*. 2014. LNCS 8693. P. 740–755.

275. Lin T.-Y., Goyal P., Girshick R. et al. Focal Loss for Dense Object Detection // *IEEE Transactions on Pattern Analysis and Machine Intelligence*. 2020. V. 42, No. 2. P. 318–327.

276. Ling X., Halevy A., Wu F., Yu C. Synthesizing union tables from the Web // *Proc. 23rd Int. Joint Conf. on Artificial Intelligence*. 2013. P. 2677–2683.

277. Liu Y., Bai K., Mitra P., Giles C.L. TableSeer: Automatic table metadata extraction and searching in digital libraries // *Proc. 7th ACM/IEEE-CS Joint Conf. on Digital Libraries*. 2007. P. 91–100.

278. Liu Y., Bai K., Mitra P., Giles C.L. Automatic searching of tables in digital libraries // *Proc. 16th Int. Conf. on World Wide Web*. 2007. P. 1135–1136.

279. Liu Y., Mitra P., Giles C.L. Identifying table boundaries in digital documents via sparse line detection // *Proc. 17th ACM Conf. on Information and Knowledge Mining*. 2008.

280. Liu Y., Bai K., Mitra P., Giles C.L. Improving the table boundary detection in PDFs by fixing the sequence error of the sparse lines // *Proc. 10th Int. Conf. on Document Analysis and Recognition*. 2009. P. 1006–1010.

281. Liu W., Anguelov D., Erhan D. et al. SSD: Single Shot MultiBox Detector // *Proc. 14th European Conf. on Computer Vision*. 2016. P. 21–37.

282. Liu L., Ouyang W., Wang X. et al. Deep learning for generic object detection: a survey // *International Journal of Computer Vision*. 2020. V. 128, No. 2. P. 261–318.

283. Liu J., Chabot Y., Troncy R. et al. From tabular data to knowledge graphs: A survey of semantic table interpretation tasks and methods // *Journal of Web Semantics*. 2023. V. 76. P. 100761.

284. Liu T., Wang F., Chen M. Rethinking tabular data understanding with large language models // *Proc. Conf. of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*. 2024. V. 1. P. 450–482.

285. Long V. An agent-based approach to table recognition and interpretation: Ph.D. thesis / Macquarie University. 2010.

286. Lopresti D., Nagy G. A tabular survey of automated table processing // *Graphics Recognition Recent Advances*. 2000. LNCS 1941. P. 93–120.

287. Lu W., Zhang J., Fan J. et al. Large language model for table processing: a survey. *Front. Comput. Sci.* 2025. V. 19. P. 192350.

288. Luo X., Luo K., Chen X., Zhu K. Cross-lingual entity linking for web tables // *Proc. 32nd AAAI Conf. on Artificial Intelligence*. 2018. P. 362–369.

289. Mandal S., Chowdhury S.P., Das A.K., Chanda B. A complete system for detection and identification of tabular structures from document images // Image Analysis and Recognition. 2004. LNCS 3212. P. 217–225.

290. Mandal S., Chowdhury S.P., Das A.K., Chanda B. A simple and effective table detection system from document images // *Int. J. Doc. Anal. Recog.* 2006. V. 8, No. 2-3. P. 172–182.

291. Martinez-Rodriguez J., Hogan A., Lopez-Arevalo I. Information extraction meets the Semantic Web: a survey // *Semantic Web*. 2020. V. 11, No. 2. P. 255–335.

292. Mauro N.D., Esposito F., Ferilli S. Finding critical cells in web tables with SRL: trying to uncover the Devil's tease // *Proc. 12th Int. Conf. on Document Analysis and Recognition*. 2013. P. 882–886.

293. Melinda L., Bhagvati C. Parameter-free table detection method // *Proc. 15th Int. Conf. Document Analysis and Recognition*. 2019. P. 454–460.

294. Milosevic N., Gregson C., Hernandez R., Nenadic G. Disentangling the structure of tables inascientific literature // *Natural Language Processing and Information Systems*. 2016. LNCS 9612. P. 162–174.

295. Milosevic N., Gregson C., Hernandez R., Nenadic G. A framework for information extraction from tables in biomedical literature // *Int. J. Doc. Anal. Recog.* 2019. V. 22, No. 1. P. 55–78.

296. Mitlöhner J., Neumaier S., Umbrich J., Polleres A. Characteristics of open data CSV files // *Proc. 2nd Int. Conf. on Open and Big Data*. 2016. P. 72–79.

297. Mulwad V., Finin T., Joshi A. A domain independent framework for extracting linked semantic data from tables // *Search Computing*. 2012. LNCS 7538. P. 16–33.

298. Muñoz E., Hogan A., Mileo A. Using linked data to mine RDF from Wikipedia's tables // *Proc. 7th ACM Int. Conf. on Web Search and Data Mining*. 2014. P. 533–542.

299. Nagy G., Seth S., Jin D. et al. Data extraction from web tables: the devil is in the details // Proc. Int. Conf. on Document Analysis and Recognition. 2011. P. 242–246.

300. Nagy G. Learning the characteristics of critical cells from web tables // *Proc. 21st Int. Conf. on Pattern Recognition*. 2012. P. 1554–1557.

301. Nagy G., Seth S., Embley D.W. End-to-end conversion of HTML tables for populating a relational database // *Proc. 11th IAPR Int. W. on Document Analysis Systems*. 2014. P. 222–226.

302. Nagy G., Embley D.W., Krishnamoorthy M., Seth S. Clustering header categories extracted from web tables // *Proc. Document Recognition and Retrieval XXII*. 2015. SPIE 9402. P. 94020M.

303. Nagy G., Seth S. Table headers: an entrance to the data mine // *Proc. 23rd Int. Conf. on Pattern Recognition*. 2016. P. 4065–4070.

304. Nagy G. TANGO-DocLab web tables from international statistical sites (Troy200). 2016. URL: `http://tc11.cvc.uab.es/datasets/Troy_200_1`.

305. Namysl M., Esser A., Behnke S., Köhler J. Flexible table recognition and semantic interpretation system. 2021.

306. Nargesian F., Zhu E., Pu K.Q., Miller R.J. Table union search on Open Data // *Proc. VLDB Endow.* 2018. V. 11, No. 7. P. 813–825.

307. Ng H.T., Lim C.Y., Koo J.L.T. Learning to recognize tables in free text // *Proc. 37th Meeting of the ACL on Computational Linguistics*. 1999. P. 443–450.

308. Nganji J.T. The Portable Document Format (PDF) accessibility practice of four journal publishers // *Library & Information Science Research*. 2015. V. 37, No. 3. P. 254–262.

309. Nguyen T.T., Hung Nguyen Q.V., Weidlich M., Aberer K. Result selection and summarization for Web Table search // *2015 IEEE 31st Int. Conf. on Data Engineering*. 2015. P. 231–242.

310. Nielson H., Barrett W. Consensus-based table form recognition of low-quality historical documents // *Int. J. Doc. Anal. Recog.* 2006. V. 8, No. 2-3. P. 183–200.

311. Nishida K., Sadamitsu K., Higashinaka R., Matsuo Y. Understanding the semantic structures of tables with a hybrid deep neural network architecture // *Proc. 31st AAAI Conf. on Artificial Intelligence*. 2017. P. 168–174.

312. Nurminen A. Algorithmic extraction of data in tables in PDF documents: Ms. thesis. 2013.

313. Oro E., Ruffolo M. PDF-TREX: An approach for recognizing and extracting tables from PDF documents // *Proc. 10th Int. Conf. on Document Analysis and Recognition*. 2009. P. 906–910.

314. Paliwal S.S., D V., Rahul R. et al. TableNet: Deep learning model for end-to-end table detection and tabular data extraction from scanned document images // *2019 Int. Conf. on Document Analysis and Recognition*. 2019. P. 128–133.

315. Parikh A., Wang X., Gehrmann S. et al. ToTTo: A controlled table-to-text generation dataset // *Proc. 2020 Conf. on Empirical Methods in Natural Language Processing*. 2020. P. 1173–1186.

316. Pavlidis T., Zhou J. Page segmentation and classification // *CVGIP: Graphical Models and Image Processing*. 1992. V. 54, No. 6. P. 484–496.

317. Pemberton J.D., Robson A.J. Spreadsheets in business // Industrial Management & Data Systems. 2000. V. 100, No. 8. P. 379–388.

318. Penn G., Hu J., Luo H., McDonald R. Flexible web document analysis for delivery to narrow-bandwidth devices // *Proc. 6th Int. Conf. on Document Analysis and Recognition*. 2001. P. 1074–1078.

319. Perez-Arriaga M.O., Estrada T., Abad-Mota S. TAO: system for table detection and extraction from PDF documents // *Proc. 29th Int. Florida Artificial Intelligence Research Society Conference*. 2016. P. 591–596.

320. Peterman C., Chang C.-H., Alam H. A system for table understanding // *Proc. S. on Document Image Understanding Technology*. 1997. P. 55–62.

321. Pinto D., McCallum A., Wei X., Bruce Croft W. Table extraction using conditional random fields // *SIGIR Forum*. 2003. P. 235–242.

322. Pivk A. Automatic ontology generation from web tabular structures // *AI Commun.* 2006. V. 19, No. 1. P. 83–85.

323. Ponza M., Ferragina P., Chakrabarti S. On computing entity relatedness in Wikipedia, with applications // *Knowledge-Based Systems*. 2020. V. 188. P. 105051.

324. Powell S., Baker K., Lawson B. Errors in operational spreadsheets // *J. Organ. End User Comput.* 2009. V. 21, No. 3.

325. Prasad D., Gadpal A., Kapadni K. et al. CascadeTabNet: An approach for end to end table detection and structure recognition from image-based documents // *2020 IEEE/CVF Conf. on Computer Vision and Pattern Recognition Workshops*. 2020. P. 2439–2447.

326. Proust C. Early-dynastic tables from southern Mesopotamia, or the multiple facets of the quantification of surfaces // *Mathematics, Administrative and Economic Activities in Ancient Worlds*. 2020. P. 345–395.

327. Pujara J., Rajendran A., Ghasemi-Gol M., Szekely P. A common framework for developing table understanding models // *Proc. ISWC 2019 Satellite Tracks*. 2019.

328. Pujara J., Szekely P., Sun H., Chen M. From tables to knowledge: Recent advances in table understanding // *Proc. 27th ACM SIGKDD Conf. on Knowledge Discovery and Data Mining*. 2021. P. 4060–4061.

329. Pyreddy P., Croft W.B. TINTIN: a system for retrieval in text tables // *Proc. 2nd ACM Int. Conf. on Digital Libraries*. 1997. P. 193–200.

330. Qasim S., Mahmood H., Shafait F. Rethinking table recognition using graph neural networks // *Int. Conf. on Document Analysis and Recognition*. 2019. P. 142–147.

331. Rahm E. The case for holistic data integration // *Advances in Databases and Information Systems*. 2016. LNCS 9809. P. 11–27.

332. Raja S., Mondal A., Jawahar C.V. Table structure recognition using top-down and bottom-up cues // *Proc. 16th European Conf. on Computer Vision*. 2020. LNIP 12373. P. 70–86.

333. Raman V., Hellerstein J. Potter's wheel: an interactive data cleaning system. *Proc. 27th Int. Conf. on Very Large Data Bases*. 2001. P. 381–390.

334. Ramel J.-Y., Crucianu M., Vincent N., Faure C. Detection, extraction and representation of tables // *Proc. 7th Int. Conf. on Document Analysis and Recognition*. 2003. P. 374–378.

335. Rastan R., Paik H.-Y., Shepherd J. TEXUS: A task-based approach for table extraction and understanding // *Proc. ACM S. on Document Engineering*. 2015. P. 25–34.

336. Rastan R., Paik H.-Y., Shepherd J., Haller A. Automated table understanding using stub patterns // *Database Systems for Advanced Applications*. 2016. LNCS 9642. P. 533–548.

337. Rastan R., Paik H.-Y., Shepherd J. A PDF wrapper for table processing // *Proc. ACM S. on Document Engineering*. 2016. P. 115–118.

338. Rastan R., Paik H.-Y., Shepherd J. et al. TEXUS: Table extraction system for PDF documents // *Databases Theory and Applications*. 2018. LNCS 10837. P. 345–349.

339. Rastan R., Paik H.-Y., Shepherd J. TEXUS: A unified framework for extracting and understanding tables in PDF documents // *Inf. Process. Manag.* 2019. V. 56, No. 3. P. 895–918.

340. Ré C., Sadeghian A.A., Shan Z. et al. Feature engineering for knowledge base construction // *IEEE Data Eng. Bull.* 2014. V. 37, No. 3. P. 26–40.

341. Redmon J., Farhadi A. YOLOv3: An incremental improvement // *CoRR*. 2018. abs/1804.02767. URL: `http://arxiv.org/abs/1804.02767`.

342. Ren S., He K., Girshick R., Sun J. Faster R-CNN: Towards real-time object detection with region proposal networks // *CoRR*. 2015. abs/1506.01497. URL: `http://arxiv.org/abs/1506.01497`.

343. Riba P., Dutta A., Goldmann L. et al. Table detection in invoice documents by graph neural networks // *Proc. 15th Int. Conf. on Document Analysis and Recognition*. 2019. P. 122–127.

344. Rim C.S., Nakajima K. On rectangle intersection and overlap graphs // *IEEE Transactions on Circuits and Systems I: Fundamental Theory and Applications*. 1995. V. 42, No. 9. P. 549–553.

345. Ristoski P., Paulheim H. RDF2Vec: RDF graph embeddings for data mining // *Proc. 15th Int. Semantic Web Conf.: Part I*. 2016. P. 498–514.

346. Ritze D., Lehmberg O., Bizer C. Matching HTML tables to DBpedia // *Proc. 5th Int. Conf. on Web Intelligence, Mining and Semantics*. 2015. P. 10:1–10:6.

347. Ritze D., Bizer C. Matching web tables to DBpedia – a feature utility study // *Proc. 20th International Conf Advances in Database Technology*. 2017. P. 210–221.

348. Roldán J., Jiménez P., Corchuelo R. On extracting data from tables that are encoded using HTML // *Knowledge-Based Systems*. 2020. V. 190. P. 105157.

349. Roldán J. Enterprise data integration: on extracting data from HTML tables: Ph.D. thesis. Universidad de Sevilla. 2020.

350. Roldán J., Jiménez P., Szekely P., Corchuelo R. TOMATE: A heuristic-based approach to extract data from HTML tables // *Information Sciences*. 2021. V. 577. P. 49–68.

351. Santoso H.A., Haw S.-C., Abdul-Mehdi Z.T. Ontology extraction from relational database: concept hierarchy as background knowledge // *Knowledge-Based Systems*. 2011. V. 24, No. 3. P. 457–464.

352. Sarawagi S. Information extraction // *Foundations and Trends in Databases*. 2007. V. 1, No. 3. P. 261–377.

353. Scaffidi C., Shaw M., Myers B. Estimating the numbers of end users and end user programmers // *Proc. IEEE S. on Visual Languages and Human-Centric Computing*. 2005. P. 207–214.

354. Schreiber S., Agne S., Wolf I. et al. DeepDeSRT: Deep learning for detection and structure recognition of tables in document images // *Proc. 14th IAPR Int. Conf. on Document Analysis and Recognition*. 2017. V. 1. P. 1162–1167.

355. Seth S., Jandhyala R., Krishnamoorthy M., Nagy G. Analysis and taxonomy of column header categories for web tables // *Proc. 9th IAPR Int. W. on Document Analysis Systems*. 2010. P. 81–88.

356. Seth S., Nagy G. Segmenting tables via indexing of value cells by table headers // *Proc. 12th Int. Conf. on Document Analysis and Recognition*. 2013. P. 887–891.

357. Shafait F., Keysers D., Breuel T. Performance comparison of six algorithms for page segmentation // *Document Analysis Systems VII*. 2006. LNCS 3872. P. 368–379.

358. Shafait F., Smith R. Table detection in heterogeneous documents // *Proc. 9th IAPR Int. W. on Document Analysis Systems*. 2010. P. 65–72.

359. Shahab A., Shafait F., Kieninger T., Dengel A. An open approach towards the benchmarking of table structure recognition systems // *Proc. 9th IAPR Int. W. on Document Analysis Systems*. 2010. P. 113–120.

360. Siddiqui S.A., Malik M.I., Agne S. et al. DeCNT: Deep deformable CNN for table detection // *IEEE Access*. 2018. V. 6. P. 74151–74161.

361. Siddiqui S.A., Fateh I.A., Rizvi S.T.R. et al. DeepTabStR: Deep learning based table structure recognition // *Proc. 15th Int. Conf. Document Analysis and Recognition*. 2019. P. 1403–1409.

362. Siddiqui S.A., Khan P.I., Dengel A., Ahmed S. Rethinking semantic segmentation for table structure recognition in documents // *Proc. 15th Int. Conf. Document Analysis and Recognition*. 2019. P. 1397–1402.

363. e Silva A.C., Jorge A.M., Torgo L. Design of an end-to-end method to extract information from tables // *Int. J. Doc. Anal. Recog.* 2006. V. 8, no. 2–3. P. 144–171.

364. e Silva A.C. Parts that add up to a whole: a framework for the analysis of tables: Ph.D. thesis / University of Edinburgh. 2010.

365. e Silva A.C. Metrics for evaluating performance in document analysis: application to tables // *Int. J. Doc. Anal. Recog.* 2011. V. 14, No. 1. P. 101–109.

366. Simonyan K., Zisserman A. Very deep convolutional networks for large-scale image recognition // *Proc. 3rd Int. Conf. on Learning Representations*. 2015. P. 1–14.

367. Singh R., Gulwani S. Transforming spreadsheet data types using examples // *SIGPLAN Not.* 2016. V. 51, No. 1. P. 343–356.

368. Son J.-W., Park S.-B. Web table discrimination with composition of rich structural and content information // Applied Soft Computing. 2013. V. 13, No. 1. P. 47–57.

369. Srihari S., Lam S., Govindaraju V. et al. Document Image Understanding. Tech. Rep. CEDAR-TR-92-1. 1992.

370. Stoffel A., Spretke D., Kinnemann H., Keim D., Enhancing document structure analysis using visual analytics // *Proc. 2010 ACM Symp. on Applied Computing*. 2010. P. 8–12.

371. Suchanek F., Abiteboul S., Senellart P. PARIS: Probabilistic alignment of relations, instances, and schema // *Proc. VLDB Endow.* 2011. V. 5, No. 3. P. 157–168.

372. Y. Sui, M. Zhou, M. Zhou, et al. Table meets LLM: Can large language models understand structured table data? A benchmark and empirical study // Proc. 17th ACM Int. Conf. on Web Search and Data Mining. 2024. P. 645–654.

373. Sun H., Ma H., He X. et al. Table cell search for question answering // *Proc. 25th Int. Conf. on World Wide Web*. 2016. P. 771–782.

374. Sun N., Zhu Y., Hu X. Faster R-CNN based table detection combining corner locating // *Proc. 15th Int. Conf. Document Analysis and Recognition*. 2019. P. 1314–1319.

375. Sun K., Rayudu H., Pujara J. A hybrid probabilistic approach for table understanding // *Proc. AAAI Conf. on Artificial Intelligence*. 2021. V. 35, No. 5. P. 4366–4374.

376. Swidan A., Hermans F. Semi-automatic extraction of cross-table data from a set of spreadsheets // *Proc. Int. Symp. on End User Development*. 2017. LNCS 10303. P. 84–99.

377. Syed Z., Finin T., Mulwad V., Joshi A. Exploiting a web of semantic data for interpreting tables // *Proc. WebSci10: Extending the Frontiers of Society On-Line*. 2010. P. 26–27.

378. Takeoka K., Oyamada M., Nakadai S., Okadome T. Meimei: An efficient probabilistic approach for semantically annotating tables // *Proc. 33rd AAAI Conf. on Artificial Intelligence*. 2019. V. 33, No. 01. P. 281–288.

379. Tallet P., Marouard G. The harbor of Khufu on the Red Sea Coast at Wadi al-Jarf, Egypt // *Near Eastern Archaeology*. 2014. V. 77, No. 1. P. 4–14.

380. Tan M., Le Q.V. EfficientNet: Rethinking model scaling for convolutional neural networks // *CoRR*. 2019. abs/1905.11946. URL: `http://arxiv.org/abs/1905.11946`.

381. Tao C., Embley D.W. Automatic hidden-web table interpretation, conceptualization, and semantic annotation // *Data Knowl. Eng.* 2009. V. 68, No. 7. P. 683–703.

382. Tensmeyer C., Morariu V., Price B. et al. Deep splitting and merging for table structure decomposition // *Proc. 15th Int. Conf. on Document Analysis and Recognition*. 2019. P. 114–121.

383. Tersteegen W., Wenzel C. SCANTAB: table recognition by reference tables // *Proc. IAPR W. on Document Analysis Systems*. 1998.

384. Tijerino Y.A., Embley D.W., Lonsdale D.W. et al. Towards ontology generation from tables // *World Wide Web*. 2005. V. 8, No. 3. P. 261–285.

385. Tran D.N., Tran T.A., Oh A. et al. Table detection from document image using vertical arrangement of text blocks // *International Journal of Contents*. 2015. V. 11, No. 4. P. 77–85.

386. Tupaj S., Shi Z., Chang C.H., Alam H. Extracting tabular information from text files / EECS, Tufts University, Medford, USA. 1996.

387. Turró M.R. Are PDF documents accessible? // *Inform. Technol. Libr.* 2008. V. 27, No. 3. P. 25.

388. Venetis P., Halevy A., Madhavan J. et al. Recovering semantics of tables on the web // *Proc. VLDB Endowment*. 2011. V. 4, No. 9. P. 528–538.

389. Verborgh R., de Wilde M. Using OpenRefine / Packt Publishing Ltd. 2013.

390. Wang Y., Hu J. Detecting tables in HTML documents // *Proc. 5th Int. W. on Document Analysis Systems V*. 2002. P. 249–260.

391. Wang Y., Hu J. A machine learning based approach for table detection on the Web // *Proc. 11th Int. Conf. on World Wide Web*. 2002. P. 242–250.

392. Wang Y. Document analysis: table structure understanding and zone content classification: Ph.D. thesis / University of Washington. 2002.

393. Wang Y., Phillips I.T., Haralick R.M. Table structure understanding and its performance evaluation // *Pattern Recognition*. 2004. V. 37, No. 7. P. 1479–1497.

394. Wang Z., Dong H., Jia R. et al. TUTA: Tree-based transformers for generally structured table pre-training // *Proc. 27th ACM SIGKDD Conf. on Knowledge Discovery & Data Mining*. 2021. P. 1780–1790.

395. Wang D.Z., Dong X.L., Sarma A.D. et al. Functional dependency generation and applications in pay-as-you-go data integration systems // *Proc. 12th Int. W. on the Web and Databases*. 2009.

396. Wang J., Wang H., Wang Z., Zhu K.Q. Understanding tables on the Web // *Conceptual Modeling*. 2012. LNCS 7532. P. 141–155.

397. Wang X. Tabular abstraction, editing, and formatting: Ph.D. thesis / University of Waterloo. 1996.

398. Wang Y., Phillips I., Haralick R. Table structure understanding and its performance evaluation // *Pattern Recognition*. 2004. V. 37, No. 7. P. 1479–1497.

399. Wu X., Cao C., Wang Y. et al. Extracting knowledge from web tables based on DOM tree similarity // *Knowledge Science, Engineering and Management*. 2016. P. 302–313.

400. Wu X., Chen H., Bu C. et al. HUSS: A heuristic method for understanding the semantic structure of spreadsheets // *Data Intelligence*. 2023. V. 5, No. 3. P. 537–559.

401. Xie S., Girshick R., Dollar P. et al. Aggregated residual transformations for deep neural networks // *Proc. 2017 IEEE Conf. on Computer Vision and Pattern Recognition*. 2017. P. 5987–5995.

402. Xiong K., Huang C. A., Wybrow M., Wu Y. TableCanoniser: Interactive grammar-powered transformation of messy, non-relational tables to canonical tables // *Proc. CHI Conference on Human Factors in Computing Systems*. 2025. Article 849. P. 1–20.

403. Xu L., Fan W., Sun J. et al. A knowledge-based table recognition method for Chinese bank statement images // *Proc. IEEE Int. Conf. on Image Processing*. 2016. P. 3279–3283.

404. Yamada I., Asai A., Sakuma J. et al. Wikipedia2Vec: An efficient toolkit for learning and visualizing the embeddings of words and entities from Wikipedia // *Proc. 2020 Conf. on Empirical Methods in Natural Language Processing: System Demonstrations*. 2020. P. 23–30.

405. Yan L., Sheng J., Tu Y. et al. Automatic construction of RDF with web tables // *Expert Systems with Applications*. 2021. V. 182. P. 115200.

406. Yang Y., Luk W.-S. A framework for web table mining // *Proc. 4th Int. W. on Web Information and Data Management*. 2002. P. 36–42.

407. Yepes A. J., Verspoor K. Literature mining of genetic variants for curation: quantifying the importance of supplementary material // *Database*. 2014. bau003.

408. Yepes A. J., Zhong P., Burdick D. ICDAR 2021 competition on scientific literature parsing // *Proc. 16th Int. Conf. on Document Analysis and Recognition* 2021. LNIP 12824. P. 605–617.

409. Yildiz B., Kaiser K., Miksch S. pdf2table: A method to extract table information from PDF files // *Proc. 2nd Indian Int. Conf. on Artificial Intelligence*. 2005. P. 1773–1785.

410. Yin P., Neubig G., Yih W., Riedel S. TaBERT: Pretraining for joint understanding of textual and tabular data // *Proc. the 58th Annual M. of the Association for Computational Linguistics*. 2020. P. 8413–8426.

411. Yoshida M., Torisawa K., Tsujii J. A method to integrate tables of the world wide web // *Proc. Int. W. on Web Document Analysis*. 2001. P. 31–34.

412. Yoshida M., Matsumoto K., Kita K. Table topic models for hidden unit estimation // *Information Retrieval Technology*. 2016. LNCS 9994. P. 302–307.

413. Yu W., Li Z., Zeng Q., Jiang M. Tablepedia: Automating PDF table reading in an experimental evidence exploration and analytic system // *World Wide Web Conf.* 2019. P. 3615–3619.

414. Zaidi S.S.A., Ansari M.S., Aslam A. et al. A survey of modern deep learning based object detection models // *Digital Signal Processing*. 2022. V. 126. P. 103514.

415. Zanibbi R., Blostein D., Cordy J. A survey of table recognition // *Int. J. Doc. Anal. Recognit.* 2004. V. 7, No. 1. P. 1–16.

416. Zavitsanos E., Mavroeidis D., Spyropoulou E. et al. ENTRANT: A large financial dataset for table understanding // *Sci Data*. 2024. 11, 876.

417. Zhang X.-W., Lyu M.R., Dai G.-Z. Extraction and segmentation of tables from Chinese ink documents based on a matrix model // *Pattern Recognition*. 2007. V. 40, No. 7. P. 1855–1867.

418. Zhang Z. Towards efficient and effective semantic table interpretation // *Proc. 3th Int. Semantic Web Conf.* 2014. LNCS 8796. P. 487–502.

419. Zhang Z. Effective and efficient semantic table interpretation using TableMiner+ // *Semantic Web*. 2017. V. 8, No. 6. P. 921–957.

420. Zhang S., Balog K. Auto-completion for data cells in relational tables // *Proc. 28th ACM Int. Conf. on Information and Knowledge Management*. 2019. P. 761–770.

421. Zhang S., Balog K. Web table extraction, retrieval, and augmentation: a survey // *ACM Trans. Intell. Syst. Technol.* 2020. V. 11, No. 2.

422. Zhang S., Dai Z., Balog K., Callan J. Summarizing and exploring tabular data in conversational search // *Proc. 43rd Int. ACM SIGIR Conf. on Research and Development in Information Retrieval*. 2020. P. 1537–1540.

423. Zhang S., Meij E., Balog K., Reinanda R. Novel entity discovery from web tables // *Proc. The Web Conf.* 2020. P. 1298–1308.

424. Zhang M., Perelman D., Le V., Gulwani S. An integrated approach of deep learning and symbolic analysis for digital PDF table extraction // *Proc. 25th Int. Conf. on Pattern Recognition*. 2021. P. 4062–4069.

425. Zhang Z., Zhang J., Du J., Wang F. Split, embed and merge: an accurate table structure recognizer // *Pattern Recognition*. 2022. V. 126. P. 108565.

426. Zhang X., Wang D., Dou L., et al. A survey of table reasoning with large language models. *Front. Comput. Sci.* 2025. V. 19, No. 9.

427. Zhang S., Balog K. EntiTables: Smart assistance for entity-focused tables // *Proc. 40th Int. ACM SIGIR Conf. on Research and Development in Information Retrieval*. 2017. P. 255–264.

428. Zhang X., Luo S., Zhang B. et al. TableLLM: Enabling tabular data manipulation by LLMs in real office usage scenarios // *arXiv*. 2025.

429. Zheng X., Burdick D., Popa L. et al. Global Table Extractor (GTE): A framework for joint table identification and cell structure recognition using visual context // *2021 IEEE Winter Conference on Applications of Computer Vision*. 2021. P. 697–706.

430. Zheng M., Feng X., Si Q., et al. Multimodal Table Understanding // *Proc. 62nd Annual M. of the Association for Computational Linguistics*. 2024. V. 1. P. 9102–9124.

431. Zhong X., Tang J., Yepes A. J. PubLayNet: largest dataset ever for document layout analysis // *Proc. 15th Int. Conf. on Document Analysis and Recognition*. 2019. P. 1015–1022.

432. Zhong X., Shafiei Bavani E., Yepes A. J. Image-based table recognition: data, model, and evaluation // *Proc. 16th European Conf. on Computer Vision*. 2020. P. 564–580.

433. Zhu G., Iglesias C. Computing Semantic Similarity of Concepts in Knowledge Graphs // *IEEE Transactions on knowledge and data engineering*. 2017. V. 29, No. 1. P. 72–85.

434. Zhu G., Iglesias C. Exploiting semantic similarity for named entity disambiguation in knowledge graphs // *Expert Systems with Applications*. 2018. V. 101. P. 8–24.

435. Zuyev K. Table image segmentation // *Proc. 4th Int. Conf. on Document Analysis and Recognition*. 1997. V. 2. P. 705–708.

436. Zwicklbauer S., Seifert C., Granitzer M. DoSeR – a knowledge-base-agnostic framework for entity disambiguation using semantic embeddings // *Proc. 13th Int. Conf. on The Semantic Web. Latest Advances and New Domains*. 2016. V. 9678. P. 182–198.

Список иллюстративного материала

Список таблиц

Приложение A

Грамматика языка CRL

Ниже представлена грамматика предметно-ориентированного языка CRL в расширенной форме Бэкуса–Наура:

```
ruleset      ::= import* rule*
import       ::= 'import static' identifier ('.*')? EOL
rule         ::= 'rule #' id EOL 'when' EOL condition*
                 'then' EOL action* 'end' EOL?
condition    ::= query_type identifier?
                 (':' (constraint (',' constraint)*
                 (',' assignment)? | assignment))? EOL
query_type   ::= 'cell' | 'entry' | 'label' | 'category' |
                 'no cells' | 'no entries' | 'no labels' |
                 'no categories'
assignment   ::= identifier ':' expression
action       ::= merge | split | set_text | set_indent |
set_tag | new_entry | new_label |
add_label | set_parent | set_category | group
merge        ::= 'merge' identifier 'with' identifier EOL
split        ::= 'split' identifier EOL
set_text     ::= 'set text' expression 'to' identifier EOL
set_indent   ::= 'set indent' expression 'to' identifier EOL
set_tag      ::= 'set tag' expression 'to' identifier EOL
new_entry    ::= 'new entry' identifier ('as' expression)? EOL
new_label    ::= 'new label' identifier ('as' expression)? EOL
add_label    ::= 'add label' (identifier | expression
                 ('of' (identifier | expression))?)
                 'to' identifier EOL
set_parent   ::= 'set parent' identifier 'to' identifier EOL
set_category ::= 'set category' (identifier | expression)
                 'to' identifier EOL
group        ::= 'group' identifier 'with' identifier EOL
```

где `identifier` — Java-идентификатор (сокращенный или полный), `id` — целочисленный литерал, `constraint` — Java-выражение логического типа данных Boolean, `expression` — Java-выражение.

Приложение Б

Примеры тестовых таблиц

Приводятся примеры тестовых таблиц двух коллекций, которые использовались в тестовых задачах Troy (раздел 5.4.4) и SAUS (раздел 5.4.5) соответственно.

Б.1. Коллекция Troy200

	2004 $ thousands	2005 $ thousands	2006 $ thousands	2007 $ thousands
Canada				
Total revenue	1,612,694	1,663,808	1,770,769	1,901,308
Operating revenue	1,594,230	1,650,174	1,761,737	1,888,917
Non-operating revenue	18,464	13,634	9,033	12,390
Total expenses	1,401,159	1,464,959	1,533,970	1,556,922
Operating expenses	1,366,472	1,426,520	1,491,448	1,520,422
Non-operating expenses	34,687	38,439	42,522	36,500
Operating margin[1]	227,758	223,654	270,288	368,495
Estimated number of carriers	38,659	38,897	40,677	41,283

Рисунок Б.1 — Пример таблицы со структурированными ячейками в шапке

State	Company Name	Plant I.D.	Plant Name	County	Biomass/ Coal Cofiring Capacity	Total Plant Capacity
AL	DTE Energy Services	50407	Mobile Energy Services LLC	Mobile	91	91
AL	Georgia-Pacific Corp	10699	Georgia Pacific Naheola Mill	Choctaw	31	78
AL	International Paper Co	52140	International Paper Prattville Mill	Autauga	49	90
AR	Domtar Industries Inc	54104	Ashdown	Little River	157	157
AZ	Tucson Electric Power Co	126	H Wilson Sundt Generating Station	Pima	173	559
DE	Conectiv Delmarva Gen Inc	593	Edge Moor	New Castle	252	710
FL	International Paper Co-Pensacola	50250	International Paper Pensacola	Escambia	83	83
FL	Jefferson Smurfit Corp	10202	Jefferson Smurfit Fernandina Beach	Nassau	74	128
FL	Stone Container Corp-Panama Ci	50807	Stone Container Panama City Mill	Bay	20	34
GA	Georgia Pacific CSO LLC	54101	Georgia Pacific Cedar Springs	Early	101	101
GA	International Paper Co-Augusta	54358	International Paper Augusta Mill	Richmond	85	85
GA	SP Newsprint Company	54004	SP Newsprint	Laurens	45	82
HI	Hawaiian Com & Sugar Co Ltd	10604	Hawaiian Comm & Sugar Puunene Mill	Maui	46	62
IA	Ames City of	1122	Ames Electric Services Power Plant	Story	109	109
IA	Archer Daniels Midland Co	10860	Archer Daniels Midland Clinton	Clinton	180	211
IA	University of Iowa	54775	University of Iowa Main Power Plant	Johnson	21	23
KY	East Kentucky Power Coop, Inc	6041	H L Spurlock	Mason	659	1 609

Рисунок Б.2 — Пример таблицы с многоколонной компоновкой боковика

Year	Schools	Pupils					Grade 1	Leaving certificates
		Pre-primary education	Grades		Additional education (10th grade)	Total		
			1-6	7-9				
1990	4 869	2 189	389 410	197 719	3 602	592 920	67 427	61 054
1991	4 843	2 240	392 059	197 505	4 193	595 997	66 320	64 175
1992	4 758	2 375	392 537	195 532	3 777	594 221	63 837	65 634
1993	4 610	2 454	390 892	193 591	3 369	590 306	62 060	65 483
1994	4 539	3 126	387 306	193 657	3 434	587 523	61 004	64 297
1995	4 474	3 973	384 369	196 642	3 178	588 162	64 364	63 756
1996	4 391	5 293	380 932	200 349	2 554	589 128	64 337	63 514
1997	4 319	6 520	381 078	202 234	2 543	592 375	66 543	64 247
1998	4 228	7 440	382 746	199 204	2 289	591 679	65 993	66 726
1999	4 099	7 327	388 063	193 646	2 236	591 272	67 212	67 043
2000	4 022	10 881	392 150	188 417	2 003	593 451	65 306	65 937
2001	3 953	12 613	393 267	187 998	1 849	595 727	65 313	63 747
2002	3 873	12 393	392 741	190 617	1 605	597 356	63 574	61 450
2003	3 808	12 434	387 934	195 585	1 461	597 414	61 300	60 831
2004	3 720	12 335	381 785	197 414	1 614	593 148	59 823	63 828

Рисунок Б.3 — Пример таблицы с иерархической компоновкой шапки

	2002		
	Total households (estimated number of households)	**Households with expenditures % reporting**	**Total repairs and renovations average expenditure1($)**
Period of construction			
Total repairs and renovations	**8 054 930**	**75,6**	**2 910**
Before 1946	1 023 500	80,5	4 202
1946–1960	1 096 870	78,5	3 128
1961–1970	1 005 390	77,1	2 718
1971–1980	1 656 110	80,1	2 787
1981–1990	1 463 430	78,4	2 881
1991 and after	1 465 030	67,7	2 504
Not stated	344 620	48,5	1 387
Contract work	**8 054 930**	**48**	**1 850**
Before 1946	1 023 500	53,1	2 853
1946–1960	1 096 870	54,2	1 958
1961–1970	1 005 390	49,5	1 771
1971–1980	1 656 110	49,2	1 708
1981–1990	1 463 430	50,6	1 855
1991 and after	1 465 030	40	1 528
Not stated	344 620	26,6	798

Рисунок Б.4 — Пример таблицы с иерархической компоновкой боковика

Б.2. Коллекция SAUS200

STATE	Post Office Abbreviation	2-DIGIT ANSI	2007			
			Number of mines	Total reserves	Method of mining	
					Underground	Surface
				(Million short tons)		
United States			**1 374**	**489 395**	**333 277**	**156 118**
Alabama	AL	01	49	4 141	963	3 178
Alaska	AK	02	1	6 107	5 423	684
Arizona	AR	04	1	0	0	0
Arkansas	AZ	05	2	417	272	144
California	CA	06	(NA)	(NA)	(NA)	(NA)
Colorado	CO	08	12	16 092	11 331	4 761
Georgia	GA	13	(NA)	4	2	2
Idaho	ID	16	(NA)	4	4	0
Illinois	IL	17	21	104 347	87 811	16 536
Indiana	IN	18	27	9 379	8 699	681
Iowa	IA	19	(NA)	2 189	1 732	457
Kansas	KS	20	2	971	(NA)	971
Kentucky	KY	21	417	29 618	16 770	12 848
Kentucky, Eastern			394	10 219	990	9 229
Kentucky, Western			23	19 399	15 780	3 619

Рисунок Б.5 — Пример таблицы с перерезом в шапке

SOURCE OF INCOME	2007					
	Total persons with income	Under 65 years old	65 years old and over	White \1	Black \2	Hispanic origin \3
Unit indicator	(1,000s)					
Total	210 019	174 534	35 485	172 453	23 654	25 874
Earnings	158 777	151 541	7 236	129 975	17 748	21 887
Wages and salary	149 437	143 251	6 186	121 930	17 087	20 713
Nonfarm self-employment	12 499	11 450	1 049	10 708	938	1 455
Farm self-employment	2 048	1 801	247	1 859	109	111
Unemployment compensation	5 200	4 996	204	4 185	721	622
State or local only	4 942	4 762	180	3 987	666	593
Combinations	258	234	24	198	55	29
Workers compensation	1 604	1 491	113	1 332	187	225
State payments	566	518	48	452	68	94
Employment insurance	682	652	30	570	82	94
Own insurance	56	53	3	44	10	5
Other	370	337	32	324	34	50
Social security	41 897	10 342	31 556	36 061	4 070	2 736
Supplemental security income (SSI)	5 039	3 902	1 137	3 411	1 218	777
Public assistance	1 828	1 748	80	1 078	630	340

Рисунок Б.6 — Пример таблицы с иерархией перерезов

State	2007	2007	2007	State	2007	2007	2007
	Crude petroleum and natural gas extraction (211111) \1				Natural gas liquid extraction (211112) \1		
	Establishments	Number of employees \2	Annual payroll ($1,000)		Establishments	Number of employees \2	Annual payroll ($1,000)
United States	7 221	133 286	8 987 718	United States	321	8 523	616 805
Alabama	27	(\3)	(D)	Alabama	2	(\5)	(D)
Alaska	14	(\4)	(D)	Alaska	1	(\5)	(D)
Arizona	10	(\5)	1 112	Arkansas	2	(\7)	(D)
Arkansas	88	1 255	78 824	California	10	137	10 666
California	192	8 903	532 013	Colorado	18	465	37 591
Colorado	371	7 458	599 286	Deleware	1	(\7)	(D)
Connecticut	1	(\6)	(D)	Florida	4	(\8)	(D)
Deleware	6	(\5)	(D)	Illinois	3	(\6)	(D)
District of Columbia	1	(\7)	(D)	Indiana	1	(\7)	(D)
Florida	39	229	9 724	Kansas	10	(\8)	(D)
Georgia	4	(\5)	(D)	Kentucky	4	(\5)	(D)
Hawaii	1	(\7)	(D)	Louisiana	40	941	65 084
Idaho	3	(\5)	(D)	Michigan	6	85	5 160
Illinois	160	963	34 101	Minnesota	3	(\5)	(D)
Indiana	46	195	7 655	Mississippi	3	70	5 308
Iowa	1	(\7)	(D)	Missouri	1	(\5)	(D)
Kansas	381	3 087	161 035	Montana	7	(\5)	(D)
Kentucky	89	921	44 039	New Mexico	16	670	44 988
Louisiana	363	12 056	792 842	North Dakota	6	(\6)	(D)
Maryland	5	(\7)	1 346	Ohio	4	(\7)	393
Massachusetts	6	(\7)	484	Oklahoma	41	(\3)	(D)
Michigan	95	1 171	72 813	Pennsylvania	12	(\8)	(D)
Minnesota	8	(\5)	3 928	South Dakota	4	(\7)	(D)
Mississippi	76	952	48 950	Tennessee	1	(\7)	(D)
Missouri	10	(\5)	(D)	Texas	88	2 872	240 803
Montana	72	664	44 294	Utah	6	111	5 559
Nebraska	22	82	2 982	Washington	1	(\7)	(D)
Nevada	21	(\6)	5 547	West Virginia	6	(\5)	(D)
New Jersey	2	(\7)	(D)	Wyoming	20	665	46 737

Рисунок Б.7 — Пример таблицы с чередованием столбцов боковика и тела

Приложение B

Прикладные задачи

Представлены сопроводительные материалы части прикладных задач, описанных в главе 6.

B.1. Создание тематических слоев электронной карты

Представлен единственный набор CRL-правил, который использовался в прикладной задаче создания тематических слоев электронной карты на основе пространственно-временных данных, извлекаемых из статистических отчетов Иркутскстата (рис. B.1), — раздел 6.4.1.

1. Генерация и категоризация столбцевых меток из ячеек, расположенных в шапке:

```
when
  cell corner: cl == 1, rt == 1
  cell c: cl > 1, rb <= corner.rb, !blank
then
  set tag "HEAD" to c
  new label c
  set category "HEAD" to c.label
```

2. Восстановление родительско-дочерних связей между столбцевыми метками:

```
when
  cell c1: tag == "HEAD"
  cell c2: tag == "HEAD", rt == c1.rb + 1,
  (c1.cl <= cl && cr < c1.cr) || (c1.cl < cl  && cr <= c1.cr)
then
  set parent c1.label to c2.label
```

3. Генерация и категоризация строчных меток из ячеек, расположенных в боковике:

	A	B	C	D	E	F	G	H	I	J	K
9	**Выбросы основных загрязняющих веществ от отдельных групп источников загрязнения тонн**										
10		Загрязняющие вещества									
11		твердые		диоксид серы		оксид углерода		оксид азота		углеводороды с учетом ЛОС (исключая метан)	
12		от сжигания топлива (для выработки электро- и тепло-энергии)	от технологических и других процессов	от сжигания топлива (для выработки электро- и тепло-энергии)	от технологических и других процессов	от сжигания топлива (для выработки электро- и тепло-энергии)	от технологических и других процессов	от сжигания топлива (для выработки электро- и тепло-энергии)	от технологических и других процессов	от сжигания топлива (для выработки электро- и тепло-энергии)	от технологических и других процессов
13	**ВСЕГО ПО ОБЛАСТИ**	71329	29539	211227	16029	31118	120448	53230	51665	516	39496
14	**Городские округа:**										
15	МО города Братска	6665	11803	9773	3900	4429	65932	6590	1843	84	968
16	Зиминское городское МО	211	18	226	1	506	8	91	4		22
17	г. Иркутск	11265	322	44426	359	3916	682	11654	166	11	671
18	г. Саянск	4444	112	18066	11	42	67	2507	42		6821
19	г. Свирск	1189	22	290	8	222	22	49	5	1	1
20	г. Тулун	777	32	1126	11	916	12	166	6	1	22
21	МО города Усолье - Сибирское	4931	49	16970	9	22	21	2976	52		218
22	г. Усть-Илимск	11118	2685	6290	932	3031	297	3201	299		439
23	г. Черемхово	1185	162	2167	11	916	713	491	229		43
24	**Муниципальные районы:**										
25	Ангарское МО	29301	2185	111011	7090	132	7122	16652	35267	11	16668
26	Балаганский район										
27	МО города Бодайбо и района	682	1608	511	185	2689	2901	318	908	29	326
28	Братский район	493	118	290	31	1165	129	166	184	7	122
29	Жигаловский район	39	65	44	12	300	1135	22	162	2	11
30	Заларинский район	316	12	267	2	626	6	32	11		12
31	Зиминский район	112	290	22	8	118	6	16	3		22
32	Иркутское районное МО	318	52	211	31	682	112	21	29		51
33	Казачинско-Ленский район	705	22	85	9	529	184	29	29		11
34	Катангский район	1621	294	62	71	4939	29743	440	2931	297	5080
35	Качугский район		29				82		21		7
36	Киренский район	11	26	29	29	71	11602	65	1660	26	216
37	Куйтунский район	4	2		1	1	4		3		11

Рисунок В.1 — Пример таблицы из отчета Иркутскстата со статистическими сведениями для формирования тематических слоев электронной карты административно-территориального деления Иркутской области

```
when
  cell c : rt > 1, cl == 1, !blank
then
  set tag "STUB" to c
  new label c
  set category "STUB" to c.label
```

4. Восстановление родительско-дочерних связей между строчными метками:

```
when
  cell c1: tag == "STUB"
  cell c2: tag == "STUB", rt > c1.rt, indent == c1.indent + 2
  no cells: tag == "STUB", indent == c1.indent, rt > c1.rt,
  rt < c2.rt
then
  set parent c1.label to c2.label
```

5. Генерация вхождений из ячеек, расположенных в теле:

```
when
  cell c: cl > 1, rt > 1, tag == null, !blank
then
  set tag "DATA" to c
  new entry c
```

6. Ассоциирование вхождений и столбцевых меток:

```
when
  cell c1: tag == "HEAD", label.terminal
  cell c2: tag == "DATA", cl == c1.cl
then
  add label c1.label to c2.entry
```

7. Ассоциирование вхождений и строчных меток:

```
when
  cell c1: tag == "STUB"
  cell c2: tag == "DATA", rt == c1.rt
then
  add label c1.label to c2.entry
```

B.2. Наполнение информационно-аналитической системы

Представлен набор CRL-правил, использованных в прикладной задаче наполнения базы данных ИАС Института национального развития Монголии статистическими сведениями по социально-экономическому положению территорий Монголии (раздел 6.4.2). На рис. B.2 демонстрируется пример одной из исходных таблиц, предоставленных Институтом национального развития Монголии.

1. Генерация строчных меток с исключением случаев агрегаций данных:

```
when
  cell c: cl == 1, rt > 1, cl == cr, text not matches
  "(?i).*(total)|.*(region)|.*(average)|(\\s*)"
then
  new lable c
  set category "ROW_LABEL" to c.label
```

2. Генерация столбцевых меток верхнего уровня:

```
when
  cell c: rt == 1, cl > 1, !blank
then
  new lable c
  set category "COL_LABEL" to c.label
```

3. Генерация столбцевых меток вложенных уровней:

```
when
  cell c1: cl > 1, rt < 3
  cell c2: cl > 1, rt < 4, rt == c1.rb + 1,
  c1.cl <= cl && cr <= c1.cr, text not matches "[\\s\\d.*]*"
then
  new lable c1
  new lable c2
  set category "COL_LABEL" to c.label
  set category "COL_LABEL" to c.label
  set parent c1 to c2
```

	A	B	C	D	E	F	G	H	I	J	K	L
1	$NAME	*SALES OF INDUSTRIAL PRODUCTS, by regions, aimags and the Capital, at current prices*										
2	$MEASURE	mln.tog										
3	$BEGIN							сая төг *mln.tog*				
4	**Аймаг, нийслэл**	**Aimag, capital**	**1990**	**1991**	**1992**	**1993**	**1994**	**1995**	**1996**	**1997**	**1998**	**1999**
5												
6	**БҮГД**	**TOTAL**	**8425,40**	**14180,20**	**27717,00**	**156358,20**	**233288,80**	**296756,40**	**314422,80**	**458722,40**	**439954,30**	**480511,20**
7	**Баруун бүс**	***Western region***	**449,43**	**730,19**	**1016,60**	**3513,38**	**5531,80**	**5750,70**	**5898,60**	**5758,70**	**5248,20**	**6816,80**
8	Баян-Өлгий	*Bayan-Olgii*	108,25	179,53	352,69	1109,17	1414,70	1468,70	1381,20	1299,40	1208,50	1068,90
9	Говь-Алтай	*Govi-Altai*	51,32	91,21	88,43	603,81	439,50	750,60	1036,30	1104,70	930,30	1557,70
10	Завхан	*Zavkhan*	118,88	186,81	271,14	294,07	1181,90	1318,20	1393,50	1341,60	803,20	855,30
11	Увс	*Uvs*	90,38	153,31	185,10	1269,52	1592,60	1681,00	1380,40	1111,30	1097,10	2338,00
12	Ховд	*Khovd*	80,60	119,34	119,23	236,81	903,10	532,20	707,20	901,70	1209,10	996,90
13	**Хангайн бүс**	**Khangai region**	**1481,62**	**1716,62**	**6306,18**	**71361,33**	**101510,90**	**135222,90**	**120605,60**	**177907,30**	**143101,70**	**161429,30**
14	Архангай	*Arkhangai*	46,97	82,19	160,84	647,28	823,80	633,90	1078,30	723,10	816,30	808,30
15	Баянхонгор	*Bayankhongor*	65,27	99,88	138,72	1466,88	1696,10	1610,80	1812,00	2255,60	2144,00	1678,80
16	Булган	*Bulgan*	44,25	119,38	112,36	505,12	1269,10	1980,60	1847,90	1825,50	2435,10	1994,20
17	Орхон	*Orkhon*	1127,20	1092,44	5227,39	66568,94	93150,70	124425,50	108085,10	164777,30	124935,20	145819,40
18	Өвөрхангай	*Ovorkhangai*	99,60	162,55	387,28	1195,08	2410,60	3798,00	5342,10	5554,50	10621,70	9060,80
19	Хөвсгөл	*Khovsgol*	98,32	160,18	279,60	978,03	2160,60	2774,10	2440,20	2771,30	2149,40	2067,80
20	**Төвийн бүс**	**Central region**	**1409,94**	**2915,02**	**3108,25**	**15321,91**	**25914,10**	**37354,10**	**45270,20**	**79058,40**	**82363,90**	**86117,10**
21	Говьсүмбэр	*Govisumber*				390,82	775,20	576,20	520,50	783,90	935,20	2225,20
22	Дархан-Уул	*Darkhan-Uul*	816,86	966,04	1643,31	8365,32	13385,40	15679,40	19121,10	17772,30	17289,90	16324,80
23	Дорноговь	*Dornogovi*	51,53	80,78	135,15	966,75	368,60	431,00	628,30	1141,80	1056,00	1427,20
24	Дундговь	*Dundgovi*	43,03	78,73	130,35	562,72	869,20	428,70	411,40	531,50	560,90	879,70
25	Өмнөговь	*Omnogovi*	36,60	75,25	130,38	457,76	879,60	827,90	876,40	1634,90	879,20	880,90
26	Сэлэнгэ	*Selenge*	391,85	1496,52	637,15	2987,02	5305,20	10460,10	9497,20	28726,40	24434,10	25037,00
27	Төв	*Tov*	70,07	217,70	431,92	1591,52	4330,90	8950,80	14215,30	28467,60	37208,60	39342,30
28	**Зүүн бүс**	**East region**	**383,26**	**719,61**	**1266,35**	**3889,82**	**4881,10**	**5585,40**	**5986,30**	**4561,90**	**2923,80**	**3811,30**
29	Дорнод	*Dornod*	256,49	474,95	856,30	2646,79	3041,30	2337,90	3179,50	2139,60	1224,10	1902,70
30	Сүхбаатар	*Sukhbaatar*	38,43	82,98	102,74	482,66	812,30	1655,00	1265,50	1135,60	1003,80	932,70
31	Хэнтий	*Khentii*	88,33	161,68	307,31	760,37	1027,50	1592,50	1541,30	1286,70	695,90	975,90
32	**Улаанбаатар**	***Ulaanbaatar***	**4700,80**	**8098,90**	**16019,80**	**62271,80**	**95450,90**	**112843,20**	**136662,10**	**191436,10**	**206316,90**	**222336,90**

Рисунок В.2 — Пример таблицы со статистическими сведениями по социально-экономическому положению территорий Монголии

4. Генерация вхождений:

```
when
  cell c: rt > 1, cl > 1, cl == cr, text matches "[\\d\\s\\.]+"
then
  new entry c
```

5. Ассоциирование вхождений со строчными метками:

```
when
  label l: category.name == "ROW_LABEL"
  entry e: cell.rt == l.cell.rt, cell.rb == l.cell.rb
then
  add label l to e
```

6. Ассоциирование вхождений со столбцевыми метками:

```
when
  label l: category.name == "COL_LABEL"
  entry e: cell.cl == l.cell.cl, cell.cr == l.cell.cr
then
  add label l to e
```

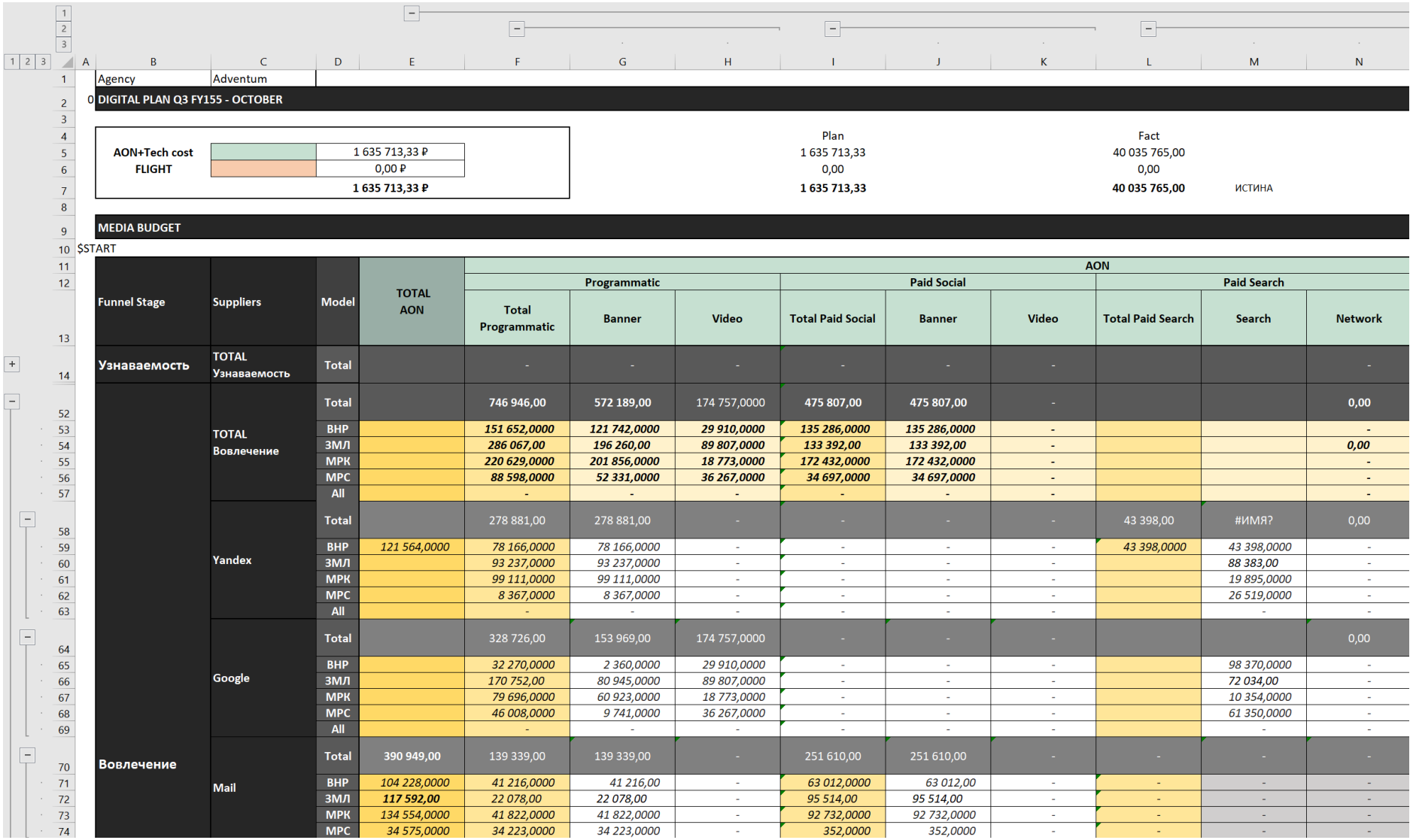
B.3. Интеграция данных медиапланирования

Представлен один из наборов CRL-правил для извлечения данных из медиапланов, подготовленных по табличным шаблонам (рис. B.3). Подобные наборы CRL-правил использовались в прикладной задаче автоматизации сбора данных медиапланирования рекламных кампаний — раздел 6.5.

1. Генерация и категоризация меток:

```
when
  cell c: cl > 4, rt == 1, !blank
then
  new label c
  set category "AON/FLIGHT" to c.label
```

```
when
  cell c: cl > 4, rt == 2, !blank
```



Agency	Adventum	
0 DIGITAL PLAN Q3 FY155 - OCTOBER		
AON+Tech cost	1 635 713,33 ₽	
FLIGHT	0,00 ₽	
	1 635 713,33 ₽	

	Plan	Fact	
	1 635 713,33	40 035 765,00	
	0,00	0,00	
	1 635 713,33	**40 035 765,00**	ИСТИНА

MEDIA BUDGET

$START

Funnel Stage	Suppliers	Model	TOTAL AON	AON								
				Programmatic			Paid Social			Paid Search		
				Total Programmatic	Banner	Video	Total Paid Social	Banner	Video	Total Paid Search	Search	Network
Узнаваемость	**TOTAL Узнаваемость**	**Total**		-	-	-	-	-	-			-
Вовлечение	**TOTAL Вовлечение**	**Total**		**746 946,00**	**572 189,00**	174 757,0000	**475 807,00**	**475 807,00**	-			**0,00**
		ВНР		***151 652,0000***	***121 742,0000***	***29 910,0000***	***135 286,0000***	***135 286,0000***	**-**			**-**
		ЗМЛ		***286 067,00***	***196 260,00***	***89 807,0000***	***133 392,00***	***133 392,00***	**-**			***0,00***
		МРК		***220 629,0000***	***201 856,0000***	***18 773,0000***	***172 432,0000***	***172 432,0000***	**-**			**-**
		МРС		***88 598,0000***	***52 331,0000***	***36 267,0000***	***34 697,0000***	***34 697,0000***	**-**			**-**
		All		-	-	-	-	-	-			-
	Yandex	**Total**		278 881,00	278 881,00	-	-	-	-	43 398,00	#ИМЯ?	0,00
		ВНР	*121 564,0000*	*78 166,0000*	*78 166,0000*	-	-	-	-	*43 398,0000*	*43 398,0000*	-
		ЗМЛ		*93 237,0000*	*93 237,0000*	-	-	-	-		*88 383,00*	-
		МРК		*99 111,0000*	*99 111,0000*	-	-	-	-		*19 895,0000*	-
		МРС		*8 367,0000*	*8 367,0000*	-	-	-	-		*26 519,0000*	-
		All		-	-	-	-	-	-		-	-
	Google	**Total**		328 726,00	153 969,00	174 757,0000	-	-	-			0,00
		ВНР		*32 270,0000*	*2 360,0000*	*29 910,0000*	-	-	-		*98 370,0000*	-
		ЗМЛ		*170 752,00*	*80 945,0000*	*89 807,0000*	-	-	-		*72 034,00*	-
		МРК		*79 696,0000*	*60 923,0000*	*18 773,0000*	-	-	-		*10 354,0000*	-
		МРС		*46 008,0000*	*9 741,0000*	*36 267,0000*	-	-	-		*61 350,0000*	-
		All		-	-	-	-	-	-		-	-
	Mail	**Total**	**390 949,00**	139 339,00	139 339,00	-	251 610,00	251 610,00	-	-	-	-
		ВНР	*104 228,0000*	*41 216,0000*	*41 216,00*	-	*63 012,0000*	*63 012,00*	-	-	-	-
		ЗМЛ	***117 592,00***	*22 078,00*	*22 078,00*	-	*95 514,00*	*95 514,00*	-	-	-	-
		МРК	*134 554,0000*	*41 822,0000*	*41 822,0000*	-	*92 732,0000*	*92 732,0000*	-	-	-	-
		МРС	*34 575,0000*	*34 223,0000*	*34 223,0000*	-	*352,0000*	*352,0000*	-	-	-	-

Рисунок В.3 — Пример таблицы медиаплана

```
then
  new label c
  set category "Source Group" to c.label

when
  cell c: cl > 4, rt == 3, !blank
then
  new label c
  set category "AdType" to c.label

when
  cell c: cl == 1, rt > 4, !blank
then
  new label c
  set category "Funnel Step" to c.label

when
  cell c: cl == 2, rt > 4, !blank
then
  new label c
  set category "Source" to c.label

when
  cell c: cl == 3, rt > 4, !blank
then
new label c
  set category "Model" to c.label
```

2. Генерация вхождений:

```
when
  cell c: cl > 4, rt > 4, !blank, !text.matches("-.*"),
  !text.matches(".*%"), style.fgColor == null
then
  new entry c
```

3. Ассоциирование вхождений со столбцевыми метками:

```
when
  label l: cell.rt < 4
  entry e: l.cell.cl <= cell.cl && cell.cr <= l.cell.cr
then
  add label l to e
```

4. Ассоциирование вхождений со строчными метками:

```
when
  label l: cell.cl < 4
  entry e: l.cell.rt <= cell.rt && cell.rb <= l.cell.rb
then
  add label l to e
```

5. Ассоциирование вхождений с контекстными метками:

```
when
  entry e
then
  add label "01.11.2020" of "StartDate" to e
  add label "30.11.2020" of "EndDate" to e
  add label "Adventum" of "Agency" to e
```

B.4. Конструирование онтологии предметной области

Представлено два набор CRL-правил, использованных в прикладной задаче конструирования онтологии предметной области экспертизы промышленной безопасности на основе фактов, извлекаемых из таблиц двух компоновочных типов соответственно (рис. B.4) — раздел 6.6.

Первый набор включает семь следующих правил.

1. Генерация и категоризация столбцевых меток из ячеек, расположенных в шапке:

```
when
  cell corner: rt == 1, cl == 1
  cell c: cl > corner.cr, rb <= corner.rb
then
  new label c
  set category "HEAD" to c.label
```

2. Восстановление родительско-дочерних связей между столбцевыми метками:

Наименование элемента	Кол., шт.	Размеры			Материал	
		Диаметр внутренний или наружний	Толщина стенки	Длина (высота)	Марка	Гост или ТУ
Обечайка корпуса	1	2600	26	6400	ВСт3	380
Обечайка штуцера №1 ввода трубного пучка	1	700	36	600	09Г2С	5520
Обечайка штуцера №2 ввода трубного пучка	1	700	36	600	09Г2С	5520
Обечайка штуцера №3 ввода трубного пучка	1	700	36	600	09Г2С	5520
Обечайка распределительной камеры №1	1	700	10	400	09Г2С	5520
Обечайка распределительной камеры №2	1	700	10	400	09Г2С	5520
Обечайка распределительной камеры №3	1	700	10	400	09Г2С	5520
Днище корпуса	2	2600	26	730	09Г2С	5520
Днище распределительной камеры №1	1	700	10	200	09Г2С	5520
Днище распределительной камеры №2	1	700	10	200	09Г2С	5520
Днище распределительной камеры №3	1	700	10	200	09Г2С	5520

Конструктивный элемент	Si, мм	Sp, мм	Sотб, мм	Sф, мм	t1, лет	Tнр, лет	a, мм/год	m1	m2
Обечайка корпуса межтурбного пространства	26,0	12,75	12,75	23,6	14	20	0,24	0	43,8
Обечайка штуцера №1 ввода трубного пучка межтрубного	36,0	3,3	4,0	31,6			0,43		63,0
Обечайка штуцера №2 ввода трубного пучка межтрубного	36,0	3,3	4,0	31,6			0,43		63,0
Обечайка штуцера №3 ввода трубного пучка межтрубного	36,0	3,3	4,0	31,6			0,43		63,0
Обечайка распределительной камеры №1 трубного пространства	10,0	3,3	4,0	9,9			0,1		59,0
Обечайка распределительной камеры №2 трубного пространства	10,0	3,3	4,0	9,9			0,1		59,0
Обечайка распределительной камеры №3 трубного пространства	10,0	3,3	4,0	9,9			0,1		59,0
Днище корпуса межтурбного пространства	26,0	12,2	12,2	24,9	14	20	0,12	0	102,6
Днище распределительной камеры №1 трубного пространства	10,0	3,3	4,0	11,2			0,1		72,0
Днище распределительной камеры №2 трубного пространства	10,0	3,3	4,0	11,2			0,1		72,0
Днище распределительной камеры №3 трубного пространства	10,0	3,3	4,0	11,2			0,1		72,0

а

Дата проведения	18.04.2013
Заказчик	НПЗ, ОАО "РогКо", цех 17/19б установка 21-10/3М
Объект контроля	Испаритель (теплообменник) Т-3
Заводской номер	323
Регистрационный номер	1819-Т
Метод контроля	ультразвуковой
Исходная толщина	S об. = 25,0 мм

б

Рисунок В.4 — Примеры таблиц из отчетов экспертизы промышленной безопасности: первого ($а$) и второго компоновочного типа ($б$)

```
when
  label l1
  label l2: cell.rt == l1.cell.rb + 1,
  cell.cl >= l1.cell.cl, cell.cr <= l1.cell.cr
then
  set parent l1 to l2
```

3. Генерация и категоризация строчных меток из ячеек, расположенных в боковике:

```
when
  cell corner: rt == 1, cl == 1
  cell c: rt > corner.rb, cr <= corner.cr
then
  new label c
  set category corner.text to c.label
```

4. Восстановление родительско-дочерних связей между строчными метками:

```
when
  label l1: cell.cl == 1
  label l2: cell.cl == 2,
  l1.cell.rt <= cell.rt, cell.rb <= l1.cell.rb
then
  set parent l1 to l2
```

5. Разделение объединенных ячеек, расположенных в теле:

```
when
  cell corner: rt == 1, cl == 1
  cell c: rt > corner.rb, cl > corner.cl, rt != rb
then
  split c
```

6. Генерация вхождений из ячеек, расположенных в теле:

```
when
  cell corner: rt == 1, cl == 1
  cell c: rt > corner.rb, cl > corner.cr
then
  new entry c
```

7. Ассоциирование вхождений с метками:

```
when
  entry e
  label l1: cell.cl == e.cell.cl, terminal
  label l2: cell.rt <= e.cell.rt, e.cell.rb <= cell.rb, terminal
then
  add label l1 to e
  add label l2 to e
```

Второй набор включает всего одно правило, которое выполняет следующие действия: генерирует метки из ячеек, расположенных в левом столбце; генерирует вхождения, расположенные в правом столбце; ассоциирует вхождение и метку, происходящие из ячеек одной строки.

```
when
  cell left: cl == 1
  cell right: cl == 2, rt == left.rt
then
  new label left
  new entry right
  add label left.label to right.entry
```

Приложение Г

Опубликованные Интернет-ресурсы

Следующие материалы опубликованы в виде Интернет-ресурсов:

- Репозитории исходного кода ПО извлечения таблиц из PDF-документов TabbyPDF: функциональное ядро[1]; веб-клиент[2], веб-сервер[3]; модули обнаружения таблиц на основе ИНС и фильтрации кандидатных случаев[4]; скрипты автоматизации процесса обучения ИНС обнаружения таблиц в изображениях документов[5].

- Репозитории исходного кода ПО анализа и интерпретации таблиц рабочих книг TabbyXL: функциональное ядро[6], грамматика CRL в формате ANTLR3[7], DSL-спецификация диалекта CRL для системы исполнения правил Drools[8], вики-документация[9]; виртуальные машины с развернутым ПО TabbyXL: Code Ocean[10], Docker[11].

- Инструкции и данные для воспроизведения результатов оценки производительности решений анализа и интерпретации таблиц на тестовых задачах: Troy200[12] и SAUS200[13].

[1] https://github.com/tabbydoc/tabbypdf

[2] https://github.com/tabbydoc/tabbypdf-front

[3] https://github.com/tabbydoc/tabbypdf-web

[4] https://github.com/tabbydoc/tabbypdf2

[5] https://github.com/tabbydoc/dl4td

[6] https://github.com/tabbydoc/tabbyxl

[7] https://github.com/tabbydoc/tabbyxl/blob/master/src/main/resources/CRL.g

[8] https://github.com/tabbydoc/tabbyxl/blob/master/src/main/resources/crl.dsl

[9] https://github.com/tabbydoc/tabbyxl/wiki/crl-rules

[10] https://doi.org/10.24433/CO.8210587.v1

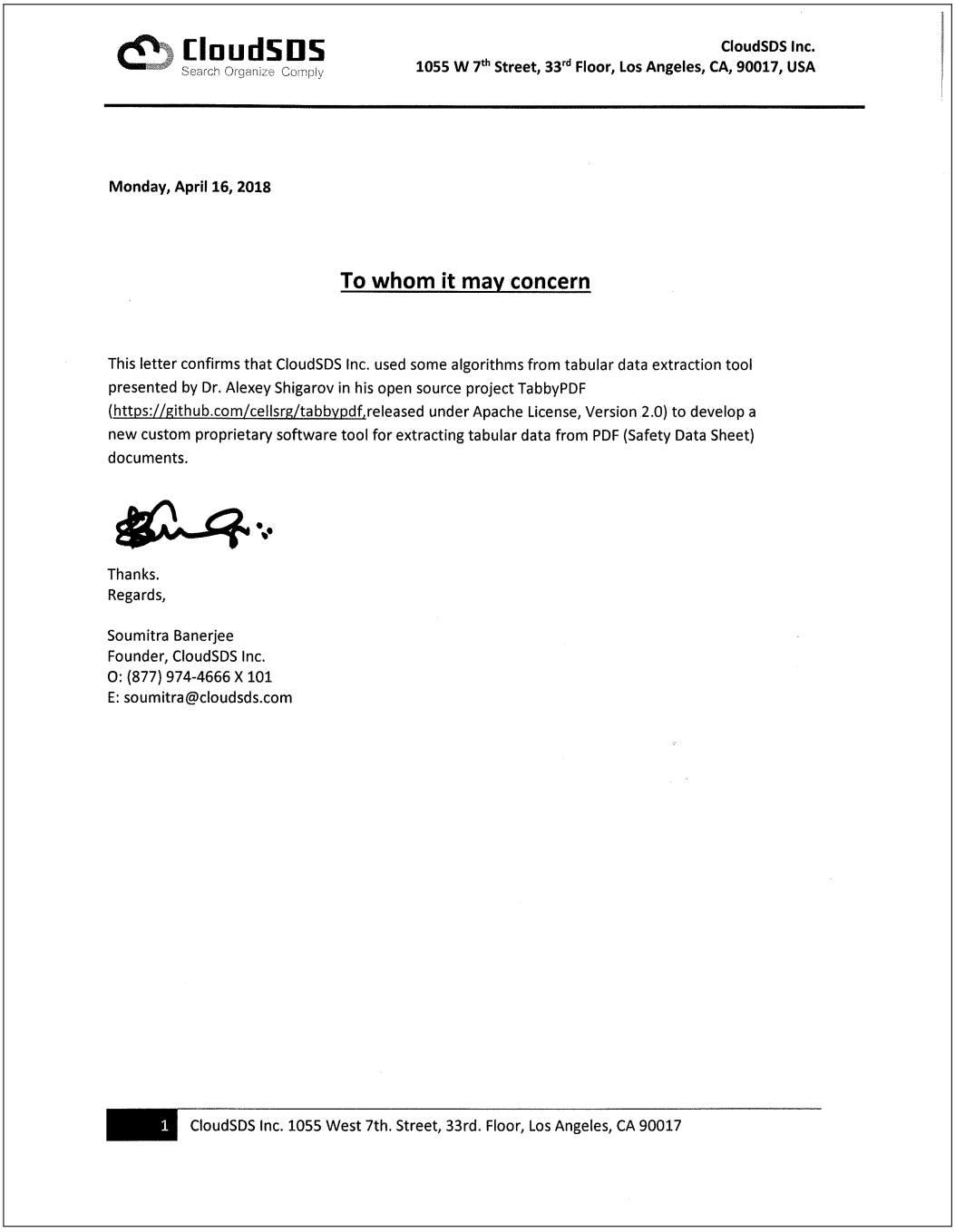
[11] https://hub.docker.com/r/tabbydoc/tabbyxl

[12] https://doi.org/10.17632/ydcr7mcrtp.6

[13] https://doi.org/10.6084/m9.figshare.14371055.v2

Приложение Д

Подтверждения внедрения результатов



CloudSDS
Search Organize Comply

CloudSDS Inc.
1055 W 7th Street, 33rd Floor, Los Angeles, CA, 90017, USA

Monday, April 16, 2018

To whom it may concern

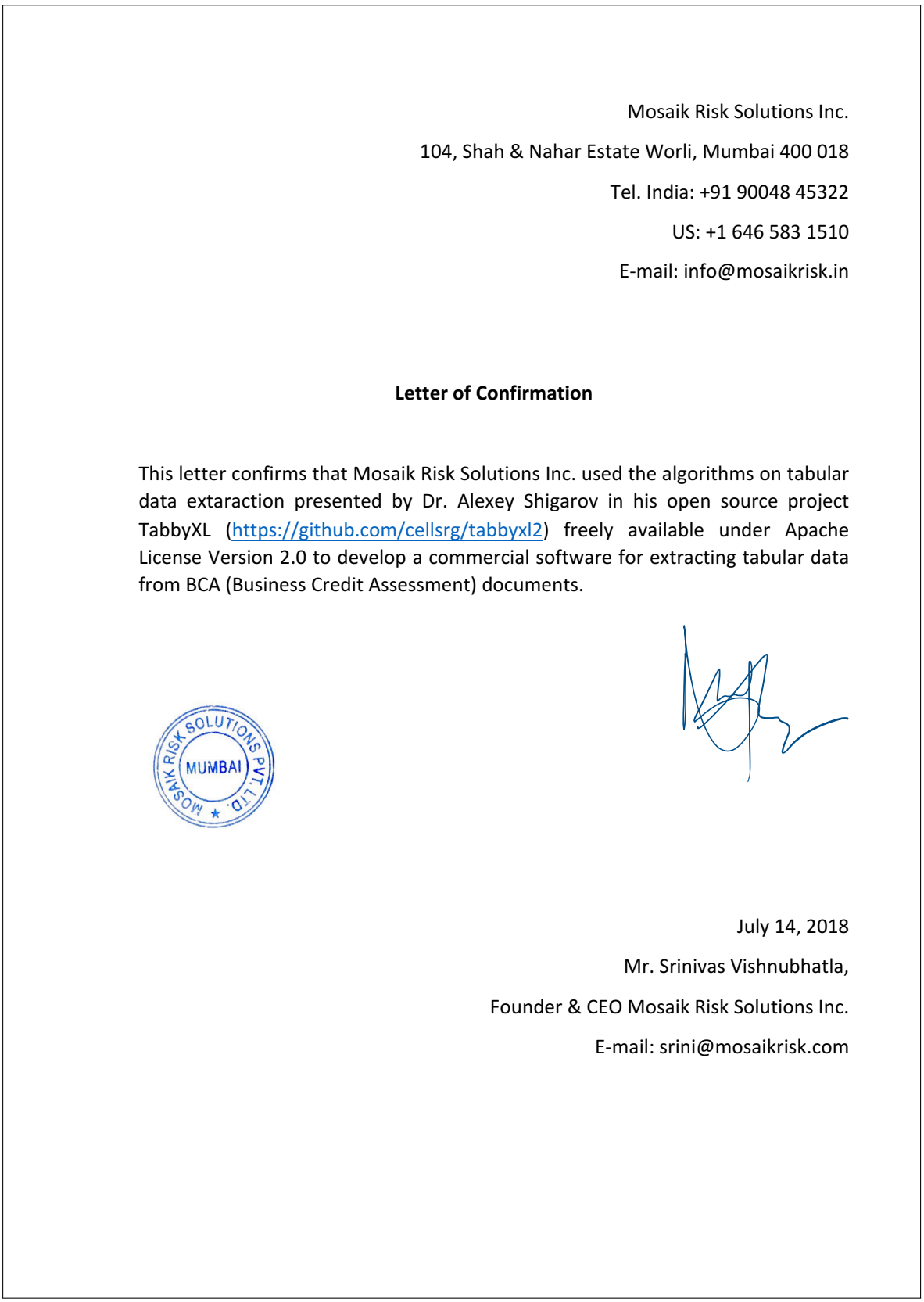
This letter confirms that CloudSDS Inc. used some algorithms from tabular data extraction tool presented by Dr. Alexey Shigarov in his open source project TabbyPDF (https://github.com/cellsrg/tabbypdf,released under Apache License, Version 2.0) to develop a new custom proprietary software tool for extracting tabular data from PDF (Safety Data Sheet) documents.

Thanks.
Regards,

Soumitra Banerjee
Founder, CloudSDS Inc.
O: (877) 974-4666 X 101
E: soumitra@cloudsds.com

1 CloudSDS Inc. 1055 West 7th. Street, 33rd. Floor, Los Angeles, CA 90017

Рисунок Д.1 — Письмо с подтверждением использования ПО TabbyPDF для извлечения табличных данных из паспортов безопасности химических продуктов в компании «CloudSDS Inc.» (Лос-Анджелес, США)



Mosaik Risk Solutions Inc.

104, Shah & Nahar Estate Worli, Mumbai 400 018

Tel. India: +91 90048 45322

US: +1 646 583 1510

E-mail: info@mosaikrisk.in

Letter of Confirmation

This letter confirms that Mosaik Risk Solutions Inc. used the algorithms on tabular data extaraction presented by Dr. Alexey Shigarov in his open source project TabbyXL (https://github.com/cellsrg/tabbyxl2) freely available under Apache License Version 2.0 to develop a commercial software for extracting tabular data from BCA (Business Credit Assessment) documents.

July 14, 2018

Mr. Srinivas Vishnubhatla,

Founder & CEO Mosaik Risk Solutions Inc.

E-mail: srini@mosaikrisk.com

Рисунок Д.2 — Письмо с подтверждением использования ПО TabbyXL для извлечения табличных данных из отчетов оценки кредитного риска в компании «Mosaik Risk Solutions Inc.» (Мумбаи, Индия)

ООО Адвентум Консалтинг

+7 495 99 88 661 | www.adventum.ru

127015, Москва, Новодмитровская ул. 2 к. 2, 18 этаж

АКТ
о внедрении программного обеспечения

Подтверждаем, что программное обеспечение извлечения реляционных данных из документных таблиц рабочих книг форматов Excel/Sheets — «TABBYXL», разработанное А. О. Шигаровым, исходный код которого опубликован в открытом доступе под свободной лицензией «Apache License Version 2.0» в репозитории GitHub по адресу: https://github.com/tabbydoc/tabbyxl, *внедрено* в ООО Адвентум Консалтинг с целью автоматизации процесса интеграции данных по медиапланированию рекламных кампаний.

Использование программного обеспечения «TABBYXL» позволило:

- Автоматизировать процесс извлечения реляционных данных по медиапланированию рекламных кампаний из документных таблиц рабочих книг форматов Excel/Sheets с помощью пользовательского программирования на основе правил.

- Упростить ввод клиентских данных по медиапланированию рекламных кампаний в единую базу данных за счет разработки пользовательских программ извлечения реляционных данных из документных таблиц рабочих книг форматов Excel/Sheets, выраженных на предметно-ориентированном языке правил анализа и интерпретации таблиц CRL.

Технический директор

кандидат технических наук Н. А. Сущенко

23 ноября 2023 м.п.

Общество с ограниченной ответственностью «Адвентум Консалтинг» «Adventum Consulting» ИНН 7717720559 * МОСКВА * 1127746018

adventum.

Рисунок Д.3 — Акт о внедрении ПО TabbyXL для автоматизации сбора данных из клиентских медиапланов в маркетинговом агентстве «Адвентум Консалтинг» (Москва, Россия)