РОССИЙСКАЯ АКАДЕМИЯ НАУК

ВСЕРОССИЙСКИЙ ИНСТИТУТ НАУЧНОЙ И ТЕХНИЧЕСКОЙ ИНФОРМАЦИИ
(ВИНИТИ)

НАУЧНО • ТЕХНИЧЕСКАЯ
ИНФОРМАЦИЯ

Серия 2. ИНФОРМАЦИОННЫЕ ПРОЦЕССЫ И СИСТЕМЫ

ЕЖЕМЕСЯЧНЫЙ НАУЧНО-ТЕХНИЧЕСКИЙ СБОРНИК

Издается с 1961 г. № 10 Москва 2013

ИНФОРМАЦИОННЫЕ СИСТЕМЫ

УДК 004.5

А. А. Соловьёв, О. В. Пескова

Об архитектурах программных систем вопросно-ответного поиска

Представлен анализ принципов построения известных систем вопросно-ответного поиска, позволяющего пользователю получать лаконичный ответ на вопрос, сформулированный на естественном языке. Предложена классификация архитектур вопросно-ответных систем, рассмотрены их основные характеристики, выделены достоинства и недостатки.

Ключевые слова: *вопросно-ответный поиск, классификация архитектур вопросно-ответных систем, информационный поиск, экспертные системы, программное обеспечение*

ВВЕДЕНИЕ

Программные системы вопросно-ответного поиска, или просто вопросно-ответные системы (от англ: *Question Answering Systems*), – это вид информационно-поисковых систем, способных обрабатывать введенный пользователем вопрос на естественном языке и выдавать осмысленный ответ. В отличие от задачи классического поиска по ключевым словам, в которой результатом является перечень документов, содержащих ответ на вопрос, в задаче вопросно-ответного поиска результат – это краткий и лаконичный ответ, сформированный системой в результате анализа разнообразных источников данных. Примером такого источника может служить некоторая коллекция полнотекстовых документов (множество страниц глобальной сети Интернет), а ответ составляется из фрагмента наиболее релевантного документа коллекции.

Традиционно в работах по вопросно-ответному поиску приводят классификацию методов или систем по используемому математическому аппарату:

- логические формы и логический вывод,
- графы зависимостей слов в предложениях,
- статистический подход и машинное обучение классификаторов,
- лексические онтологии для анализа отдельных слов текста и др.

Этой классификации следует в своём обзоре 2006 г. J. Prager [1]. Более поздний обзор других авторов [2] также следует этой классификации. Боль-

шинство исследователей следуют некоторой типовой архитектуре вопросно-ответной системы, которая, по сути, заключается в разбиении задачи вопросно-ответного поиска на четыре подзадачи: анализ вопроса, поиск, извлечение потенциальных ответов и валидация ответов. С точки зрения проектирования программного комплекса архитектурой называют способ разбиения системы на модули и определение связей между ними. В настоящей работе предлагается классификация архитектур вопросно-ответных систем, т.е. способов разбиения систем на модули.

Существует ряд различных подходов и принципов построения вопросно-ответных систем (ВОС), основными из которых являются следующие:

1) метапоисковая система;
2) система поиска по аннотированному тексту;
3) экспертная система;
4) система поиска в коллекциях вопросов и ответов.

Далее приводится обзор перечисленных архитектур (как способов декомпозиции ВОС на составляющие компоненты и подзадачи) и примеры известных программных реализаций рассмотренных подходов.

МЕТАПОИСКОВАЯ СИСТЕМА

Архитектура метапоисковой системы предусматривает использование существующей классической поисковой системы в качестве источника данных (рис. 1). ВОС преобразует введённый пользователем вопрос на естественном языке в запрос в виде ключевых слов и анализирует выдачу поисковой системы – несколько наиболее релевантных документов или их фрагментов (сниппетов).

Система анализирует вопрос пользователя с целью выделить следующие данные [3]:

- предположение о *семантическом классе ответа*;
- *фокус вопроса*, т. е. вопросительные слова, обозначающие искомую информацию, например, «в каком городе», «кто», «где», «в каком году», «когда», «сколько», «сколько метров», «какой высоты», «какого цвета» и др.; в случае простого фактографического вопроса искомая информация обычно является каким-то атрибутом некоторого объекта (имя, вес, цвет);
- *опора вопроса*, т. е. остальные члены вопросительного предложения, которые описывают уникальные свойства искомого объекта.

Метапоисковые ВОС обычно формулируют запрос по ключевым словам на основе слов, входящих в *опору вопроса*. В англоязычных системах распространён приём расширения поискового запроса синонимами и гипонимами на основе лексической онтологии WordNet [4].

Результаты поиска по ключевым словам – сниппеты – обрабатываются существующими компонентами систем автоматической обработки текста (компьютерной лингвистики). Например, выделяются все именованные сущности, соответствующие искомому семантическому классу: персоны, топонимы (географические названия), названия организаций, линейные размеры и т.п. Более глубокий лингвистический анализ (например, синтаксический или семантический разбор) позволяет выбрать из всех найденных сущностей нам более подходящие [5].

Преимущества метапоисковой архитектуры заключаются в следующем:

- отсутствие собственного индекса документов и, как следствие, отсутствие необходимости хранить огромный массив информации (для поиска в Интернете);

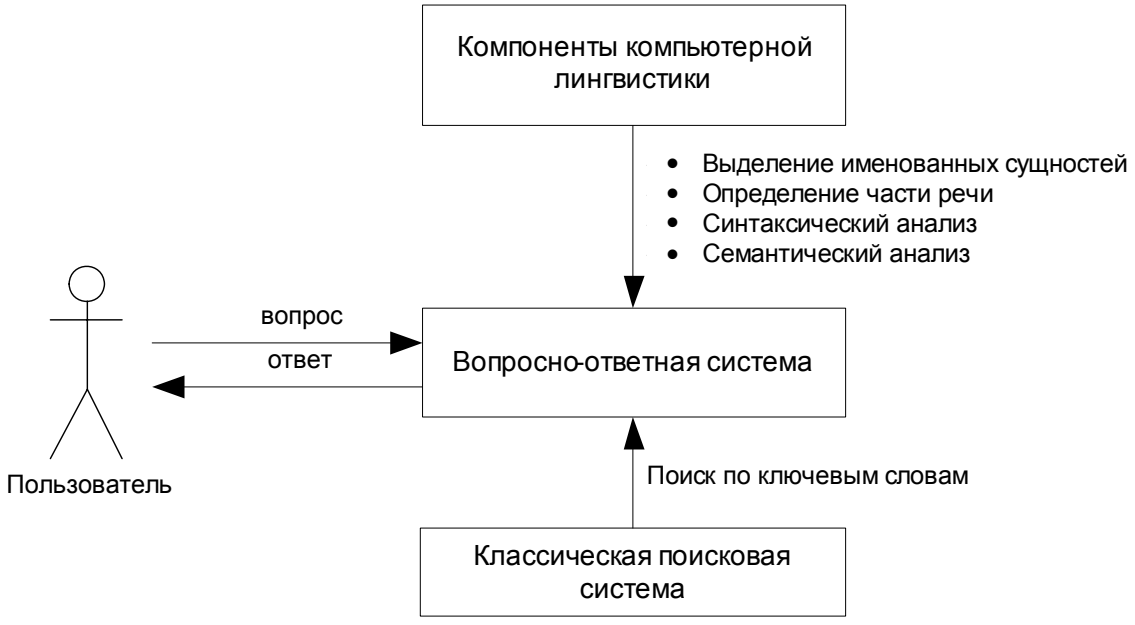


Рис. 1. Обобщённая схема архитектуры метапоисковой вопросно-ответной системы

- гибкость, т.е. возможность абстрагироваться от задач поиска и компьютерной лингвистики. ВОС использует поисковую машину и лингвистические компоненты как чёрные ящики, и обычно не возникает проблем с заменой этих компонентов. Метапоисковая ВОС может использовать любые доступные инструменты для анализа сниппетов, независимо от математического аппарата, используемого в поисковой машине или лингвистических компонентах. Например, поисковая машина может работать на деревьях решений, построенных методами машинного обучения, синтаксический анализ может выполняться вероятностными методами, а ВОС в то же время будет анализировать вопрос с помощью регулярных выражений и представлять сниппеты в виде графов синтаксических зависимостей.

Недостатками метапоисковой архитектуры являются:

- высокая вычислительная нагрузка в момент обработки вопроса, введённого пользователем, связанная с высокими вычислительными затратами на выполнение лингвистических задач;
- ограничения по управлению поиском (длина и «целостность» сниппетов, авторитетность источников и др.).

ПОИСК ПО АННОТИРОВАННОМУ ТЕКСТУ

Вопросно-ответные системы, построенные по принципу поисковых систем с коллекциями аннотированных документов, имеют в своём составе поисковый индекс документов (рис. 2). Данный индекс, в отличие от классических поисковых систем, дополняется специфическими для ВОС атрибутами. Элементами индекса являются не отдельные слова текста, а объекты детального лингвистического анализа, например:

- именованные сущности [5];
- элементарные синтаксические связки (пары грамматически связанных слов и др.);
- предикативно-аргументные структуры [6].

Построение индекса происходит с привлечением компьютерной лингвистики: каждый новый документ проходит автоматическую обработку текста на естественном языке, размечаются требуемые ВОС объекты, затем они добавляются в индекс.

Использование своего специального индекса позволяет преодолеть некоторые недостатки метапоисковой архитектуры.

Преимуществами поиска по аннотированному тексту являются:

- меньшая (по сравнению с метапоисковыми ВОС) вычислительная нагрузка в момент обработки вопроса пользователя в реальном времени благодаря специализированному индексу;
- возможность, благодаря специализированному индексу, организовать наиболее удобный для ВОС поисковый аппарат.

Недостатки поиска по аннотированному тексту состоят в следующем:

- невысокая гибкость по сравнению с метапоисковой системой: на этапе построения индекса выбирается какая-то определённая модель представления текста. Любые изменения, скорее всего, потребуют перестройки индекса – например, изменение онтологии семантических классов и имён объектов или замена грамматики зависимостей (*dependency grammar)* грамматикой связей*,(link grammar*) [7] в модели представления текста;

Рис. 2. Обобщённая схема архитектуры вопросно-ответной системы, основанной на поиске по аннотированным текстам

- потребность в значительно бóльших вычислительных ресурсах, связанная с необходимостью индексации всей коллекции (по сравнению с классическим поиском). При этом данные ресурсы расходуются недостаточно эффективно: документы анализируются целиком, хотя ответы на вопросы пользователей содержатся в очень редких предложениях.

ЭКСПЕРТНАЯ СИСТЕМА

Вопросно-ответные системы, построенные по принципу работы со структурированными базами данных, можно отнести к классу экспертных систем (рис. 3). Вопрос на естественном языке преобразуется в поисковый запрос к базе данных, содержащей структурированные факты, например, фреймы. В отличие от рассмотренных выше архитектур, такие системы могут выполнять логический вывод новой информации на основе множества разрозненных фактов. Метапоисковые же системы и системы со специальным индексом – это полнотекстовые поисковые системы. Они ищут ответ на вопрос, очевидным образом содержащийся в полном тексте документа, не пытаясь делать логический вывод новой информации, не содержащейся в тексте в явном виде.

База данных фактов может быть построена автоматически в результате анализа коллекции документов. Этот процесс аналогичен построению аннотированного индекса. Однако он происходит на более детальном уровне обработки естественного текста: извлекаются не синтаксические или поверхностно-семантические конструкции, а факты. Такие системы могут не хранить текст исходного документа, из которого был извлечён той или иной факт (фрейм).

Преимуществами ВОС, спроектированных на основе экспертной системы, являются:

- высокая скорость работы (по сравнению, например, с метапоисковой архитектурой);
- точность и достоверность результатов.

Недостатки заключаются в следующем:

- сильная зависимость от структуры фактов (фреймовой модели). Одна структура может быть адекватна одной предметной области, но не другой. Обычно это требует постоянной работы аналитиков-редакторов, чёткого понимания потребностей пользователей и очень внимательной проработки модели данных, постоянного расширения и модификации это модели для новых предметных областей;
- необходимость выбирать только авторитетные исходные тексты для извлечения информации об окружающем мире. При этом извлечённые факты могут противоречить друг другу и система должна это учитывать;
- трудоёмкость построения базы фактов, причём как вычислительная, так и организационная. Лингвистическая обработка на глубоком семантическом уровне (например, на уровне извлечения фактов) подвержена большому количеству ошибок. Она аккумулирует все ошибки лингвистической обработки на предшествующих уровнях: графематическом, морфологическом, синтаксическом и поверхностно-семантическом.

Рис. 3. Обобщённая схема архитектуры вопросно-ответной системы, применяющей принципы построения экспертных систем

ПОИСК В КОЛЛЕКЦИИ ВОПРОСОВ И ОТВЕТОВ

Другим подходом к автоматизации поиска ответов на вопросы пользователей является социальный вопросно-ответный поиск (*collaborative question answering*). В таких системах одни пользователи отвечают на вопросы других. Пользователь, имеющий информационную потребность, открывает страницу веб-сайта системы и формулирует вопрос. Система ищет похожие вопросы в коллекции вопросов и ответов и выдаёт найденный раздел, где обсуждается вопрос. Если подобного вопроса не существует, создаётся новый раздел для обсуждения вопроса. На этот вопрос отвечают другие пользователи, автору вопроса приходят уведомления по мере появления ответов. Данные в такой системе представлены в виде коллекции вопросов с ответами, которая может пополняться другими пользователями или даже автоматически.

Усложнением системы является модуль извлечения вопросов и ответов из коллекции документов. ВОС непрерывно сканирует все страницы Интернета, анализирует тексты на естественном языке и формулирует возможные вопросы по этому тексту (см. *LCC Predictive questioning* в [5]). Аналогично социальным ВОС, где пользователи оценивают ответы друг друга, данная модификация позволяет поднимать или понижать рейтинг автоматически сгенерированной пары «вопрос–ответ».

Другой подход предлагают разработчики немецкой системы LogAnswer [8]: система работает как программный робот, который обходит известные вопросно-ответные сайты с пользовательским содержимым и пытается отвечать на вопросы автоматически как обычный участник обсуждений.

Преимуществами использования коллекций вопросов и ответов являются:

- возможность развёрнутых, необязательно фактографических ответов;
- проверка достоверности ответов другими пользователями;
- низкие вычислительные затраты на поиск ответа в коллекции.

Недостатками являются:

- необходимость мотивации пользователей как для пополнения коллекции, так и для простановки оценок, особенно для ответов и вопросов, порождённых автоматически;
- трудоёмкость автоматического порождения коллекции, необходимость объёмного хранилища.

ОБЗОР ИЗВЕСТНЫХ ВОПРОСНО-ОТВЕТНЫХ СИСТЕМ

В таблице приведены примеры исследовательских и коммерческих систем, классифицированных по рассмотренным архитектурам.

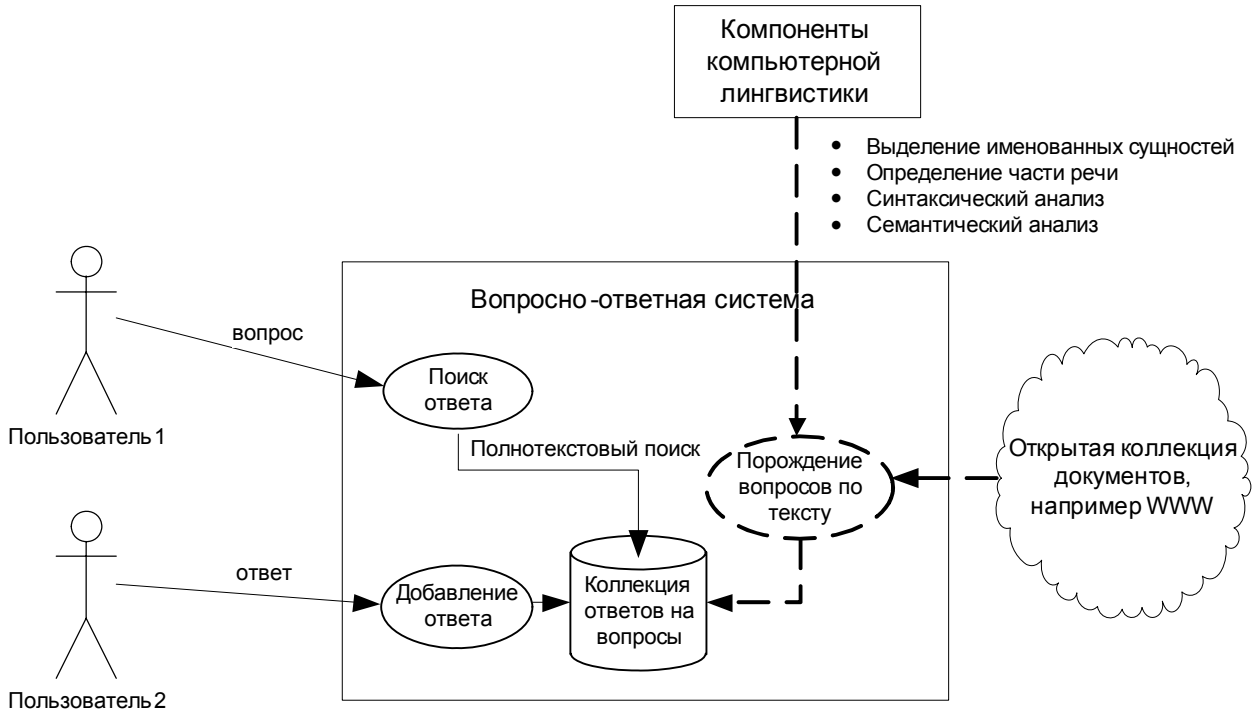


Рис. 4. Обобщённая схема архитектуры системы социального вопросно-ответного поиска

Таблица

Известные примеры исследовательских и коммерческих ВОС

		Исследовательские	Коммерческие
Метапоиск		LCC Power Answerer [9], SMU Falcon, OpenEphyra [6], AskMsr, AnswerBus, Умба [10]	–
Аннотированный индекс		LCC Chaucer-2 [11], IBM Watson [12], Javelin, START [13]	Powerset, Asknet [14]
Структурированная БД		BASEBALL, LUNAR	WolframAlfa [15], TrueKnowledge
Коллекция вопросов и ответов	**Пополняется пользователями**	–	Chacha, Ask.com [16],Yahoo!Answers [17], otvety.google.com, ответы@mail.ru
	Пополняется автоматически	LCC Ferret [5]	Swingly

Отметим отсутствие коммерческих реализаций метапоисковых систем. Это объясняется высокими вычислительными затратами на обработку каждого запроса. Например, системам OpenEphyra и Умба может потребоваться до нескольких минут процессорного времени на обработку одного вопроса. Отметим также, что система IBM Watson, успешно выступающая в телевикторине Jeopardy, работает на кластере из 3000 узлов. Создатели отмечают, что на одном процессоре системе требуется несколько часов на ответ. Однако AskNet, использующая собственный индекс, выдаёт ответ за доли секунды.

Особый интерес представляет семейство систем от Language Computer Corporation (LCC). Компания экспериментировала со всеми архитектурами и недавно запустила коммерческий стартап Swingly, основанный на исследовательских прототипах компании.

ЗАКЛЮЧЕНИЕ

В результате проделанной работы выполнен анализ подходов к решению задачи автоматического поиска ответа на вопрос, сформулированный пользователем на естественном языке. Выделены четыре типа архитектуры ВОС, у каждой из которых есть своё соотношение между трудоёмкостью предобработки информации и вычислениями «на лету», между более низким, но надёжным уровнем лингвистической обработки языка и более высоким уровнем абстракции. Некоторые архитектуры ориентированы на поиск ответа, который присутствует в явном виде, другие же позволяют порождать новую информацию на основе логического вывода из доступных фактов. Поиск ответа в полном тексте остаётся вычислительно сложной задачей: некоторым исследовательским системам требуется от нескольких минут до нескольких часов работы одного процессора на поиск ответа на вопрос. Например, IBM Watson потребовалось 3000 процессоров, чтобы успешно конкурировать с людьми в телевикторине. В то же время – это пример наиболее гибкой вопросно-ответной системы, которая использует полный арсенал современных методов для решения задачи.

СПИСОК ЛИТЕРАТУРЫ

1. Prager John. Open-Domain Question–Answering // Foundation and Trends in Information Retrieval. – 2006. – Vol. 1, № 2. – P. 91–231.
2. Kolomiyets Oleksandr, Moens Marie-Francine. A survey on question answering technology from an information retrieval perspective // Information Sciences. – 2011. – Vol. 181, Is. 24. – P. 5412–5434.
3. Соловьёв А. А., Пескова О. В. Построение вопросно-ответной системы для русского языка: модуль анализа вопросов // Новые информационные технологии в автоматизированных системах: материалы 13-го научно–практического семинара. – Моск. гос. ин-т электроники и математики. – 2010. – С. 41–49. – URL: http://nps.itas.miem.edu.ru/2010/sbornik13.pdf, свободный.
4. Проект WordNet. – URL: http://wordnet.princeton.edu/, свободный.
5. Harabagiu S., Hickl A., Lehmann J., Moldovan D. Experiments with interactive question–answering // Proceedings of the 43rd Annual Meeting on Association for Computational Linguistics (ACL '05). Association for Computational Linguistics. Stroudsburg, PA. – 2005. – P. 205–214.
6. Schlaefer N. A Semantic Approach to Question Answering. – Saarbrücken, 2007. – P. 27.
7. Протасов С. В. Обучение с нуля грамматике связей русского языка // X Национальная конференция по искусственному интеллекту с международным участием «КИИ-06». – М., 2006. – С. 515–524.
8. Dong T., Furbach U., Glöckner I., Pelzer B. A natural language question answering system as a participant in human Q&A portals // Proceedings of the Twenty-Second International Joint Conference on Artificial Intelligence (IJCAI–2011). Barcelona, Spain. July 2011. – P. 2430–2435.
9. Moldovan D., Pasca M., Surdeanu M. Some Advanced Features of LCC's Poweranswer // Advances in Open Domain Question Answering.

Text, Speech and Language Technology. – 2006. – Vol. 32, Part 1. – P. 3–34.

10. Соловьёв А. А. Кто виноват и где собака зарыта? Метод валидации ответов на основе неточного сравнения семантических графов в вопросно-ответной системе // Российский семинар по оценке методов информационного поиска. Труды РОМИП. – Казань, 2010. – С. 125–141.
11. Hickl A., Roberts K., Rink B., Bensley J., Jungen T., Shi Y., Williams J. Question Answering with LCC's Chaucer-2 at TREC 2007 // Proceedings of TREC 2007. Gaithersburg, MD, 2007.
12. Ferrucci D., Brown E., Chu–Carroll J., Fan J., Gondek D., Kalyanpur A. A., Lally A., Murdock J. W., Nyberg E., Prager J., Schlaefer N.,Welty Ch. Building Watson: An overview of the DeepQA project // AI Magazine. – 2010, 31(3). – P. 59–79.
13. Katz B., Borchardt G., Felshin S. Natural Language Annotations for Question Answering // Proceedings of the 19th International FLAIRS Conference (FLAIRS 2006). May 2006. Melbourne Beach, FL, 2006. – P. 303–306.
14. Проект AskNet. – URL: http://asknet.ru/, свободный.
15. Проект WolframAlpha. – URL: http://www.wolframalpha.com/, свободный.
16. Проект Ask.Com. – URL: http://www.ask.com/, свободный.
17. Проект Yahoo!Answers. – URL: http://answers.yahoo.com/, свободный.

Материал поступил в редакцию 20.06.13.

Сведения об авторах

СОЛОВЬЁВ Александр Александрович – аспирант МГТУ им. Н.Э. Баумана, Научно-техническая библиотека, программист
e-mail: a-soloviev@mail.ru

ПЕСКОВА Ольга Вадимовна – кандидат технических наук, доцент МГТУ им. Н.Э.Баумана,
e-mail: opeskova@mail.ru

ИНФОРМАЦИОННЫЙ ПОИСК

УДК 025.4.036 : 81’37 : 004.6

И.В. Ковалев, П.В. Зеленков, Н.А. Распопин, А.В.Демиш, К.К. Бахмарева

Дискретная декомпозиция поискового запроса с учетом семантической связи структур данных*

Раскрывается структура обработки поискового запроса и сущность понятия агент – системы при работе с поисковыми задачами. Определены начальный этап работы поисковой транзакции и конечный этап обработки отклика системы. Показано, что между этими двумя состояниями происходит собственно процесс обработки информации, в котором подробно рассматриваются основные этапы поиска информации.

Ключевые слова: *обработка информации, поисковые технологии, научно-техническая информация, система поддержки принятия решений*

Работа информационно-поисковой системы начинается с формирования поискового запроса. Проблемы поиска и обработки информации можно отнести к процессам формирования новой информационной инфраструктуры, ориентированной на информационное обеспечение инноваций, бизнеса и науки [1]. Проведено множество исследований с целью классифицирования информационных ресурсов, которые применяются в России для решения задач в разных сферах науки, образования и т.д. [2]. В связи с этим необходимо строго описать, формализовать процессы, которые рассматриваются в настоящей статье.

Поисковый запрос – исходная информация для осуществления поиска с помощью поисковой системы. Формат поискового запроса зависит как от устройства поисковой системы, так и от типа информации для поиска. Чаще всего поисковый запрос задаётся в виде набора слов или фразы, иногда – используя расширенные возможности языка запросов поисковой системы. Поскольку исходная информация, которой оперирует система, практически всегда разбивается на составляющие – переменные – дадим более краткое определение поисковому запросу, которое и будем использовать в дальнейшем (рис.1).

Поисковый запрос – это *набор переменных*, передаваемый системе для осуществления операции поиска.

Необходимо понимать, что в общем случае пользователем может являться не только человек, но и другая *программная или аппаратная система*. Регламентирование набора переменных должно производиться в каждом конкретном случае отдельно, под «задачу». Под регламентом понимается количество, качество, допустимые ограничения, возможные состояния набора переменных. В простейшем случае, когда мы получаем на вход некоторое текстовое поле, набор переменных вырождается в одну, которая несет в себе текстовую информацию и ограничена по размеру заранее строго определенным количеством символов. Поскольку цепочка обработки информации в большинстве случаев начинается с человеко-машинного взаимодействия, задачу по формированию набора переменных целесообразно передать самой поисковой системе, либо ее агенту. В случае, если подготовкой запроса пользователя будет заниматься агент (рис. 2), которым фактически может выступать персональный компьютер пользователя с установленным на нем специализированным программным обеспечением, можно в значительной степени сэкономить вычислительные мощности серверов поисковой системы.

Начальным этапом работы поисковой транзакции можно считать получение системой поискового запроса, конечным – возвращение результатов работы решающих алгоритмов. Между этими двумя состояниями происходит собственно процесс обработки информации, в котором можно выделить несколько основных этапов.

1. Этап получения поискового запроса и подготовки его к обработке системой. На этом этапе осуществляется получение набора переменных. Задача системы на этом этапе – подготовить набор переменных к дальнейшему использованию, приведение его к соответствию внутренним алгоритмам и правилам.

В общем случае система может получить на вход одну переменную, содержащую в себе текстовое поле произвольной длины. В этом случае необходимо разбить текстовое поле на отдельные составляющие, например, слова или морфы. После приведения набора переменных к внутренним стандартам необходимо провести лингвистический анализ поискового запроса для определения его грамматической, морфологической, синтаксической и семантической составляющей.

* Работа выполнена в рамках ФЦП Научные и Научно-педагогические кадры России, ГК № 16.74011.0750

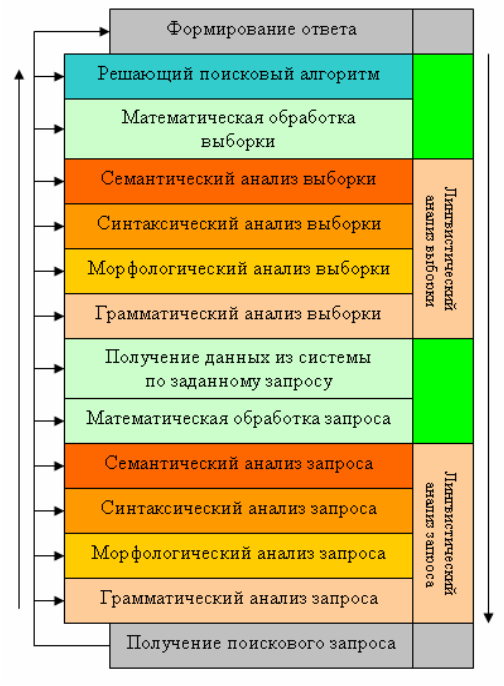


Рис. 1. Структура обработки поискового запроса

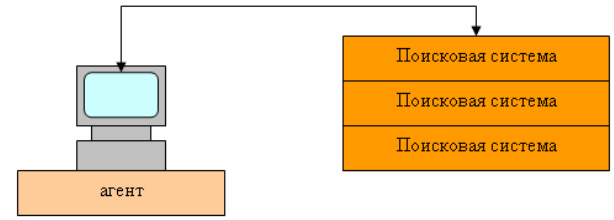


Рис. 2. Агент – система

2. На втором этапе производится сопоставление переменных поискового запроса с базами данных системы для определения соответствия. В случае, когда соответствия найдены, система переходит к третьему этапу, если соответствий найдено не было – поиск заканчивается.

3. На третьем этапе производится лингвистический анализ полученной на втором этапе выборки для определения более точного соответствия полученной выборки исходному запросу. По окончании третьего этапа система передает полученные данные на четвертый, заключительный этап.

4. На четвертом этапе решающий алгоритм системы производит отсев нерелевантных результатов и сортировку оставшихся по степени значимости. После получения результата система формирует ответ и предоставляет его пользователю.

Достоверность данного укрупненного алгоритма доказывается рядом исследований [3-6], выполненных авторами. Однако необходимо отметить, что реализация того или иного этапа в алгоритме может существенно отличаться и влиять на качество работы поисковой процедуры.

Для наглядности определим поисковый запрос. Представим себе, что необходимо получить информацию об отдыхе в Тайланде. Для определенности положим, что нас интересует информация о существующих в Тайланде курортах и расценках на аренду гостиниц. Поисковая строка будет выглядеть как: «Отдых в Тайланде».

Модуль анализа поискового запроса, получив на вход переменную, произведет ее декомпозицию. Под декомпозицией переменных понимается разделение переменных на возможно меньшие элементы до уровня морфемы. Наш поисковый запрос будет выглядеть как представлено на рис. 3: «(((отдых) (в)) ((Тайланд)(е))).

Декомпозиция поискового запроса производится лингвистическим процессором (рис. 4), который в зависимости от потребностей может определять: состав предложения, выделение основ и корней слов, выделение собственных имен и имен нарицательных, определение подлежащего, сказуемого (действия) и так далее.

В нашем случае нам может понадобиться определение подлежащего, т. е. объекта поиска и дополнительных членов предложения (дополнения, обстоятельства и т.д.), – уточнение характеристик этого объекта. Подлежащим в нашем поисковом запросе будет слово «отдых», оно несет на себе основной смысл, инициатора запроса в первую очередь интересует именно «отдых». Обстоятельством места будет слово «Тайланд», имя собственное, обозначающее страну.

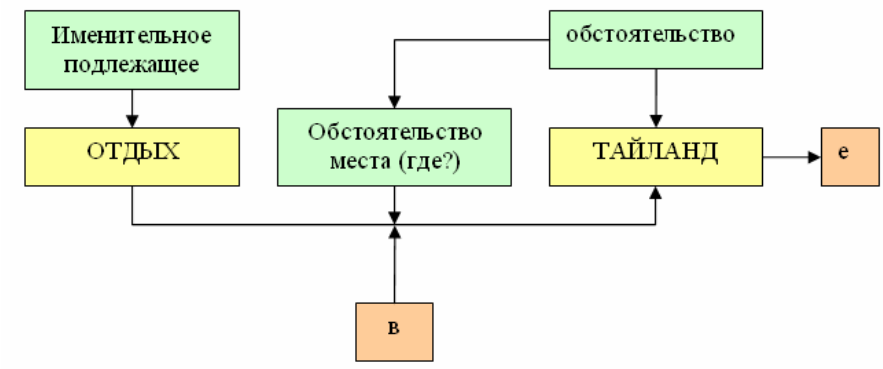


Рис. 3. Декомпозиция поискового запроса

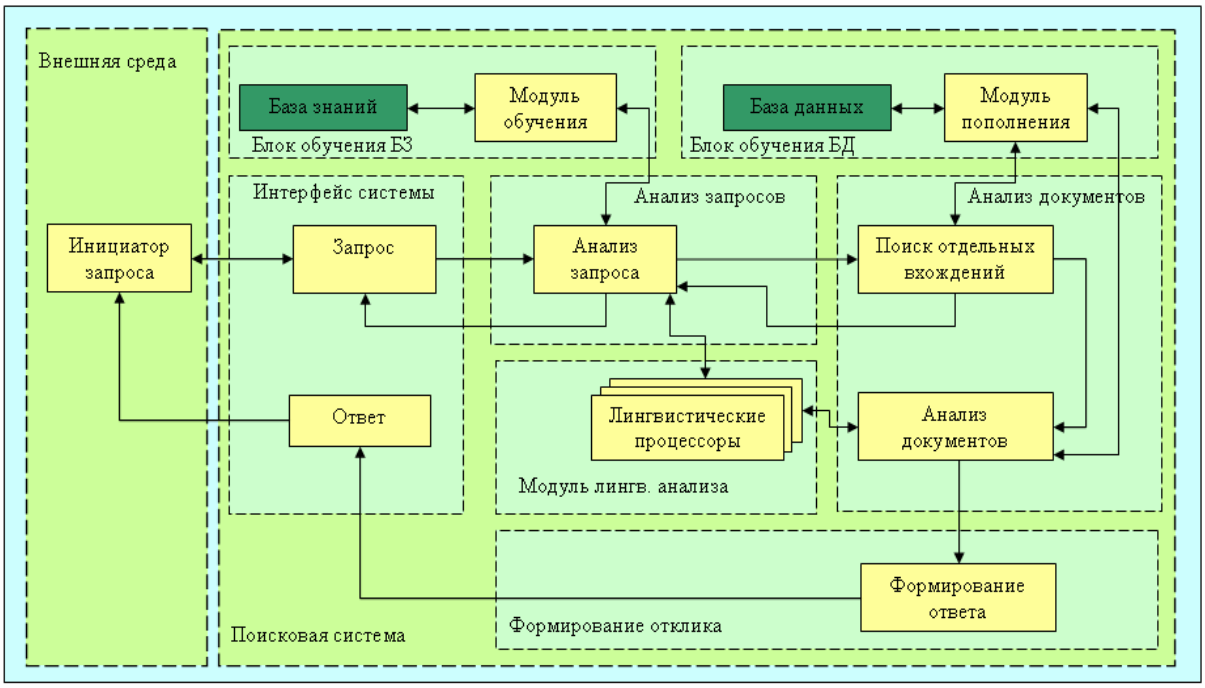


Рис. 4. Общая структура поисковой системы

Итак, в нашем случае поисковый запрос генерирует две *переменные структуры* и одну *зависимость*: подлежащее, обстоятельство и связь «действие-обстоятельство места». *Под переменной структурой* будем понимать любой объект, образованный на основе морфа - слово. Под *зависимостью* будем понимать отношения между *переменными структурами* - словами. Так как явного действия в нашем предложении нет (предложение по правилам должно быть следующего вида: «отдыхать отдых в Тайланде»), связь является односторонней, т. е. в предложении присутствует только зависимая часть. Каждая *переменная структура* может обладать рядом характеристик, например: род, число, падеж, язык, дата и так далее. Наличие характеристик требуется для последующего лингвистического анализа.

Полученные на этапе анализа запроса данные передаются далее – в анализатор документов. Анализатор документов представляет собой модуль, в котором происходит выборка данных из базы системы и сравнение с поступившими на вход данными поискового запроса. На основании анализа совокупности данных система формирует отклик – ответ на поисковый запрос, который после соответствующей обработки возвращается пользователю.

Квантом поискового запроса будем считать отдельную морфему (или морф), выражающую собой объект или объективное явление. В нашем случае квантами можно считать «отдых» и «Тайланд». Если бы запрос выглядел как: «Великолепный и недорогой отдых в летнем Тайланде», то квантами по-прежнему являлись бы морфы «отдых» и «Тайланд», поскольку «недорогой» и «летний» являются лишь объективными признаками. При последовательной реализации процедуры поиска на этапе поиска отдельных вхождений обрабатываются только кванты, признаки должны быть инкапсулированы в структуру кванта и их обработка целесообразна только в случае присутствия кванта в базе данных системы. Если на этапе поиска отдельных вхождений выясняется, что все или какой-либо один из квантов отсутствует, то поведение системы должно предусматривать семантическую замену отсутствующего кванта по имеющимся квантам и объективным признакам. Например, если на этапе поиска вхождений выясняется, что в базе присутствует только «отдых», но нет «Тайланд», то система должна подобрать вместо отсутствующего кванта другой, например «Турция» или оповестить пользователя о возникшей ситуации и предложить ему ввести недостающий квант самостоятельно, выбрать его из списка предложенных или переформулировать запрос.

На рис.5 показан срез семантической зависимости синхронизированного линейного дерева и расширенной базы. Допустим, мы хотим найти документы, в которых есть выражение «Мастер быстро проверил новую машину». Квантами в данном случае будут являться «Мастер» и «Машин (у)». Для того чтобы ответить на вопрос о присутствии в документах данного выражения, необходимо проверить наличие обоих квантов. Поскольку оба кванта в базе присутствуют, опуская саму методологию поиска, можно заключить, что, если искомое выражение содержится в документах базы, то должна существовать семантическая связь между элементами расширенной базы, при которой различные пути сходятся в одном элементе на конечном цикле итерации, т. е. должны существовать две сходящиеся семантические связи от «Мастер» и «Машин(у)». В нашем случае такие пути имеются (пути 7-8 и 3-6-8), которые сходятся в элементе номер 8. Поскольку установлено наличие в базе квантов и требуемой семантической связи, можно заняться анализом объективных признаков, если данный анализ требуется.

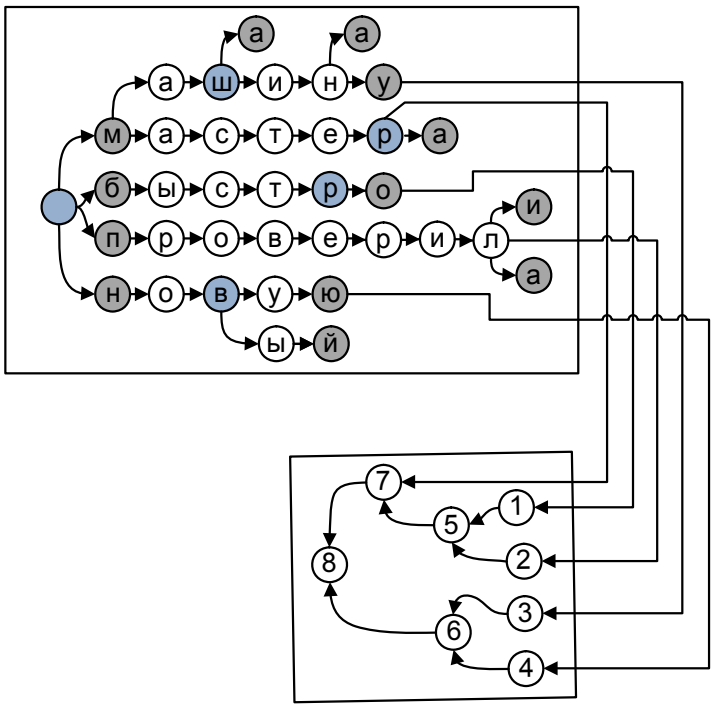


Рис. 5. Семантическая связь структур данных

Обработка поискового запроса пользователя является одной из важнейших составляющих современной поисковой системы, которая позволяет организовать интеллектуальный интерактивный обмен информацией с пользователем. Организация интерактивного обмена информацией с пользователем, основанного на технологии семантических сетей с применением лингвистического анализа, позволит значительно расширить возможности поисковой системы, учесть специфику поисковых запросов конкретного пользователя, его предпочтения и интересы.

СПИСОК ЛИТЕРАТУРЫ

1. Арский Ю.М., Цветкова В.А., Яшукова С.П. О развитии информационной инфраструктуры инновационной сферы // Научно-техническая информация. Сер. 1. – 2006. - №1. – С. 12-18.
2. Антопольский А.Б. Классификация информационных ресурсов // Научно-техническая информация. Сер. 1. – 2013. – № 4. – С. 25-27.
3. Зеленков П.В., Селиванова М.А., Брезицкая В.В., Хохлов А.П. Модуль обработки информационных запасов пользователей в сеть Интернет для корпоративных информационно – управляющих систем // Вестник Сибирского государственного аэрокосмического университета им. академика М.Ф. Решетнева. – 2009. – № 3. – С. 69-74.
4. Зеленков П.В., Каюков Е.В., Царев Р.Ю., Штарик Е.Н., Штарик А.В. К проблеме синтеза распределенных информационно – аналитических систем поддержки принятия решений // Фундаментальные исследования. – 2013. – № 4-2. – С. 286-289.
5. Распопин Н.А., Карасева М.В., Зеленков П.В., Каюков Е.В., Ковалев И.В. Модели и методы оптимизации сбора и обработки информации // Вестник Сибирского государственного аэрокосмического университета им. академика М.Ф. Решетнева. – 2012. – № 2. – С. 69-72.
6. Энгель Е.А., Ковалев И.В. Использование интеллектуальных методов для обработки информации на примере решения задач WCCI 2010 // Вестник Сибирского государственного аэрокосмического университета им. академика М.Ф. Решетнева. – 2011. – № 3. – С. 4-8.

Материал поступил в редакцию 01.07.13.

Сведения об авторах

КОВАЛЕВ Игорь Владимирович – доктор технических наук, профессор, ректор Сибирского государственного аэрокосмического университета им. академика М.Ф. Решетнева (СибГАУ им. М.Ф. Решетнева), г. Красноярск
e-mail: kleniks@yandex.ru

ЗЕЛЕНКОВ Павел Викторович – кандидат технических наук, доцент, начальник НИУ СибГАУ им. М.Ф. Решетнева.
e-mail: kleniks@yandex.ru

РАСПОПИН Николай Александрович – аспирант СибГАУ им. М.Ф. Решетнева.
e-mail: kleniks@yandex.ru

ДЕМИШ Александра Владимировна – младший научный сотрудник СибГАУ им. М.Ф. Решетнева.
e-mail: kleniks@yandex.ru

БАХМАРЕВА Кристина Константиновна – студент 5 курса СибГАУ им. М.Ф. Решетнева.
e-mail: kleniks@yandex.ru

УДК 004.65 – 028.22

Ю.Д. Калафати, П.А. Козловский, С.О. Старков, А.А. Тельных

Поиск «похожего» в графических базах*

Обсуждается проблема поиска «похожего» в базах данных графических объектов. Приведен подход к описанию процесса индексации изображений, обсуждаются различные схемы использования алгоритмов поиска дубликатов. Представлен алгоритм поиска «похожего» на основе бинарных шаблонов.

Ключевые слова: *интернет-контент, поиск дубликатов, сигнатура изображения, графический плагиат, индексация изображений, бинарный шаблон*

ВВЕДЕНИЕ

Технологии анализа сходства графических объектов, поиска дубликатов активно развиваются на протяжении последних 10-15 лет [1-7]. В связи с появлением облачных сервисов вопросы анализа «схожести» графической и видео- информации приобретают дополнительную актуальность [8-13]. Перечень возникающих при этом применений достаточно широк:

- поиск похожих графических изображений, фотографий в различных распределенных графических библиотеках;
- анализ интернет-контента на предмет наличия графической информации определенного содержания (символы, знаки, рисунки);
- поиск графического «плагиата» для мониторинга нарушений авторского права в Интернете [2];
- фильтрация графического спама для исключения дубликатов изображений при сборе графической информации в Интернете (например, поиск новостей и отслеживание новостных сюжетов [3, 4]) и т.д.

Наиболее сложной представляется проблема обнаружения сходства при неполном и частичном совпадении графических изображений, когда «меру сходства»- «похожесть» не удается однозначно формализовать.

Действительно, различные изображения разные люди могут посчитать как сходными, так и несходными.

Мы будем придерживаться такого определения нечетких дубликатов: если одно изображение можно получить из другого применением любой комбинации следующих преобразований: изменение контраста, малое масштабирование изображения, поворот на угол, кратный 90 градусам, добавление малого шума, применение сжатия с потерей качества, то такие изображения считаются нечеткими дубликатами. На фоне одинаковых изображений могут встречаться небольшие фрагменты с дополнительными символами, «баннерами», текстовыми вкраплениями. Эти изображения также следует считать «похожими». Таким образом, алгоритм поиска «похожего» должен быть устойчив к добавлению рамок, водяных знаков и другим локальным изменениям в изображении, незначительно влияющим на объем его площади.

Наглядным примером актуальности рассматриваемого подхода может служить проблема регистрации товарных знаков. Для защиты интеллектуальной собственности товарный знак необходимо зарегистрировать в реестре товарных знаков. В процессе регистрации требуется проверить, не является ли регистрируемый торговый знак похожим на зарегистрированный ранее. Если для текстовых описаний решение этой задачи известно, то в случае, когда товарный знак представляет собой изображение, данная проверка становится не такой простой. Например, должно исключаться появление на фоне уже известного логотипа дополнительных надписей или фрагментов.

В настоящей статье приведен подход к описанию процесса индексации изображений, обсуждаются различные схемы использования алгоритмов поиска дубликатов, представлен алгоритм на основе бинарных шаблонов, который использует приведенное понятие нечетких дубликатов.

ИНДЕКСАЦИЯ ИЗОБРАЖЕНИЙ

В общем случае цель любой индексации состоит в том, чтобы при поиске релевантных документов для определенных поисковых запросов оптимизировать скорость и требования к ресурсам системы. Рассмотрим случай, где документ представляет собой изображение; таким образом, конечная задача системы - поиск изображений по некоторым поисковым запросам.

Вид таких поисковых запросов может быть весьма разнообразным. Например, поиск изображений может осуществляться на основе текстового поискового запроса, запроса в виде изображения с целью найти близкие изображения, запроса в виде указания цвета, который должен преобладать на изображении, или запроса в виде указания желаемого результата некоторого детектора объектов. Все эти виды поиска и разработанные для них структуры индексов могут иметь существенные различия в своей внутренней

* Работа выполнена при финансовой поддержке Минобрнауки (НИР по Государственному контракту № 14.514.11.4022 от 10 августа 2012 г.).

реализации, однако, как правило, справедливы следующие утверждения:

- процесс индексации одного изображения сравнительно продолжителен по времени;
- наличие дубликатов (или же нечетких дубликатов) изображений в базе нежелательно, так как они с большой вероятностью попадут в результаты поиска одновременно.

Чтобы решить проблему наличия дубликатов, можно использовать алгоритмы обнаружения нечетких дубликатов. При этом, если алгоритм определения нечетких дубликатов сравнительно прост, то скорость индексации может увеличиться за счет быстрого отброса дубликатов. Насколько это будет эффективно, зависит от того, какой процент дубликатов попадает на вход системы.

Общая схема алгоритма индексации с отбрасыванием дубликатов приведена на рис. 1. Основной индекс строится исходя из конкретных задач системы, а индекс дубликатов является вспомогательным для учета и отбрасывания дубликатов изображений. Вся суть подхода заключается в предварительном поиске дубликатов в индексе дубликатов и добавлении новых уникальных изображений в этот индекс, чтобы впоследствии отбрасывать их дубликаты. В дальнейшем, под индексом мы будем понимать индекс дубликатов.

Рассмотрим структуру индекса дубликатов. В самом общем случае, индекс может представлять собой просто список *сигнатур* входящих в него изображений. Под сигнатурой будем понимать некоторый набор данных, полученных из изображения для сравнения с другими (например, данные о текстуре, цвете, размере или их комбинации, в зависимости от конкретного алгоритма обнаружения дубликатов). В частности, сигнатурой изображения может служить само изображение в виде битовой карты, однако соотношение размера и эффективности такого способа индексации изображений с целью поиска дубликатов оставляет желать лучшего. Сигнатура всегда может быть представлена в виде набора байтов.

Индексация в этом случае – это добавление новой сигнатуры изображения в список. Поиск может осуществляться перебором всех сигнатур изображений, имеющихся в базе. Очевидно, что в этом случае нам необходима некоторая функция сравнения двух сигнатур $R(S_1, S_2)$, которая возвращает значение 1, если изображения являются дубликатами, и значения 0 - в противном случае. В некоторых алгоритмах нечеткого сравнения изображений результатом сравнения является вещественное число, как правило, из отрезка [0..1] или [0..∞). В этом случае привести такое вещественное значение к бинарному виду можно с использованием порога.

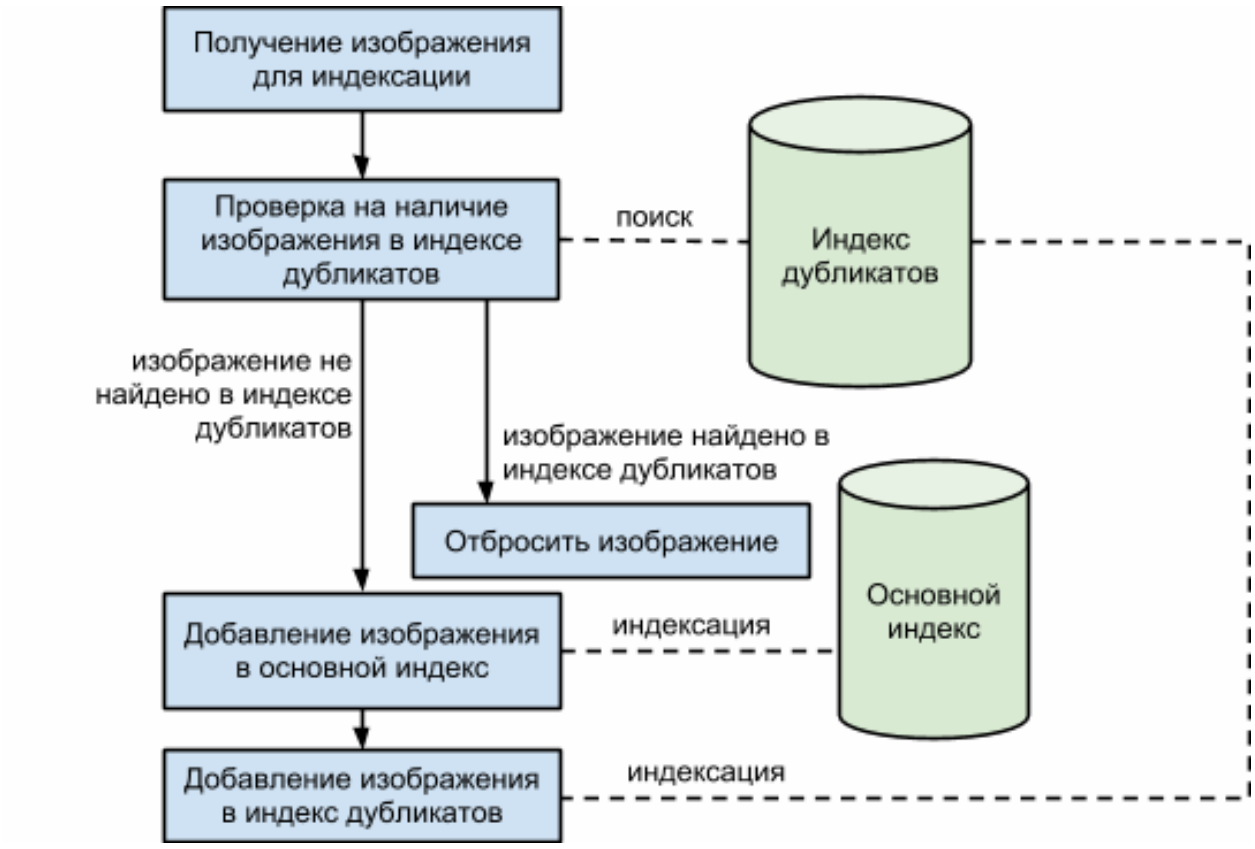


Рис. 1. Индексация с отбрасыванием дубликатов.

Описанный способ организации индекса не всегда эффективен, так как время поиска изображения в индексе растет линейно с количеством изображений, содержащихся в этом индексе. Тем не менее, при малом размере сигнатуры и быстрой процедуре сравнения он применим для многих баз данных изображений.

Чтобы оптимизировать процесс поиска, необходимо наложить дополнительные ограничения на функцию сравнения сигнатур и, соответственно, на саму сигнатуру. Если мы примем, что функция сравнения сигнатур равна единице тогда и только тогда, когда сами сигнатуры равны между собой, то можно хранить список сигнатур изображений в индексе в упорядоченном виде. В этом случае, для поиска сигнатуры в индексе может использоваться бинарный поиск, который имеет логарифмическую сложность относительно количества элементов в списке. Так как операция добавления сигнатуры в индекс должна сохранять упорядоченность сигнатур, то вместо списка предпочтительнее использовать бинарное дерево; в этом случае добавление нового изображения в индекс будет существенно более быстрым, хотя сам индекс будет занимать несколько больше памяти. Альтернативой бинарному дереву может служить хеш-таблица. В качестве такой сигнатуры может использоваться само изображение. В этом случае побайтное равенство изображений означает, что они являются дубликатами. Такой подход к формированию сигнатуры не подходит для поиска нечетких дубликатов, однако в некоторых случаях такого подхода вполне достаточно. Естественно, что при побайтном сравнении с таким же успехом этот индекс можно использовать не только для изображений, но и для документов иного вида. Для уменьшения размера индекса вместо самого бинарного представления изображения можно хранить результат применения к нему хеш-функции.

Другим вариантом организации индекса, который позволяет отбрасывать, в том числе нечеткие дубликаты, может быть введение хеш-функции от сигнатуры. Такая хеш-функция должна с высокой вероятностью возвращать одинаковый результат для близких между собой сигнатур. В этом случае результат вычисления хеш-функции может быть одинаковым и для некоторых неблизких сигнатур. При этом подходе индекс хранит для каждого изображения пару значений (результат хеш-функции от сигнатуры; сигнатура), упорядоченную по результату хеш-функции. Благодаря такой организации индекса мы можем быстро найти ту сигнатуру или сигнатуры, хеш которых совпадает с искомым, а потом сравнить эти сигнатуры с искомой. Такой подход позволяет существенно сократить количество сравнений сигнатур, однако найти подходящую хеш-функцию может быть затруднительно. Чтобы повысить вероятность нахождения нужного изображения, можно использовать несколько различных хеш-функций и проверять сигнатуры всех изображений, у которых совпала хотя бы одна из них.

Отметим, что индекс дубликатов и основной индекс могут быть как объединены, так и представлять собой отдельные структуры данных. Их объединение, как правило, производится тогда, когда основной индекс в качестве поискового запроса принимает изображение и способен найти его (возможно, нечеткие) дубликаты, т. е., служит для тех же целей, что и индекс дубликатов.

ФИЛЬТРАЦИЯ ЗАПРЕЩЕННОГО КОНТЕНТА И СПАМА

Рассмотрим возможную схему защиты от запрещенного контента и графического спама, где под контентом мы будем понимать в первую очередь изображения. Пусть имеется фильтр, который по цветовым, текстурным или другим характеристикам может обнаруживать запрещенные изображения. Внутренняя структура фильтра может быть разнообразной, однако, зачастую, для снижения вероятности ошибки, время работы такого фильтра может быть недостаточно малым для фильтрации запрещенного контента от большого числа пользователей.

Повысить производительность системы можно за счет того, что пользователи часто запрашивают одинаковые или почти одинаковые изображения. В этом случае, можно использовать алгоритмы поиска нечетких дубликатов изображений, чтобы проверить, не выполнялась ли фильтрация такого изображения ранее. Если фильтрация такого изображения уже производилась, то нет необходимости анализировать его повторно.

Общая последовательность действий показана на рис. 2. Вначале выполняется поиск фильтруемого изображения (или его нечеткого дубликата) в индексе дубликатов. Если оно обнаружено, то возвращается сохраненный ранее результат фильтрации такого изображения. В противном случае изображение идет непосредственно на вход фильтра, результат работы которого записывается в индекс дубликатов.

Соображения по организации индекса, высказанные в разделе, посвященном индексации изображений, справедливы и для системы фильтрации запрещенного контента. Единственным изменением является добавление в индекс дубликатов информации о результате фильтрации.

Чтобы ограничить рост индекса дубликатов, изображения, по которым давно не было поисковых запросов, могут удаляться из индекса. В случае повторного появления такого изображения или его нечеткого дубликата, оно просто снова попадет на вход фильтра. По сути, индекс дубликатов в этой схеме играет роль кеша с целью экономии ресурсов на повторных или схожих запросах. Индекс дубликатов может быть полностью очищен и это приведет только к временному замедлению системы до тех пор, пока частые запросы снова не попадут в индекс дубликатов.

Еще одним примером использования алгоритмов поиска нечетких дубликатов изображений могут служить системы помощи экспертам. Такие системы могут представлять собой базу данных товарных знаков, по которой можно осуществлять поиск с целью обнаружения товарных знаков, похожих на искомый. Другим вариантом может быть поиск конкретных типов графических объектов, например банкнот, среди большой базы изображений.

Общий алгоритм работы такой системы представлен на рис. 3. Индекс базы изображений для эксперта составлен заранее, и его организация аналогична организации индекса дубликатов, рассмотренного выше. Суть работы системы заключается в поиске входящего изображения в имеющемся индексе и возврате результата запроса; обновление индекса не производится.

Для таких систем тоже справедливы высказанные в разделе индексации изображений соображения о структуре индекса дубликатов. Стоит иметь в виду, что если индекс не требуется обновлять, то в некоторых случаях его можно представить более компактно. Например, если используемая функция сравнения требует побайтного совпадения сигнатур, то вместо бинарного дерева можно хранить просто упорядоченный список сигнатур и использовать бинарный поиск по этому списку.

В качестве одного из алгоритмов поиска нечетких дубликатов может использоваться **алгоритм на основе бинарных шаблонов**, суть которого состоит в том, что изображение иерархически разбивается на участки, для каждого из которых подсчитывается код участка изображения. Множество таких кодов для нескольких уровней иерархии и составляет сигнатуру изображения. Чем больше рассматриваемый уровень иерархии, тем более мелкие детали изображения принимаются в рассмотрение, но тем сильнее влияние шума или искажений, которыми и отличаются нечеткие дубликаты.

Код участка изображения представляет собой 9-битовый код, описывающий прямоугольный участок этого изображения, причем стороны этого прямоугольного участка параллельны сторонам самого изображения (рис.4). Соответствующий участок разбивается на девять равных ячеек (сеткой 3x3), и каждая из ячеек формирует один бит кода этого участка. Бит равен единице, если яркость соответствующей ячейки больше средней яркости участка изображения и нулю - в противном случае.

Обозначим среднюю яркость фрагмента F за $<I_F>$. Разобьем фрагмент изображения F на девять равных частей $f_0, \ldots, f_8$ (см. рис. 1). Подсчитаем среднюю яркость $<I_{fn}>$ каждой из частей фрагмента F.

Имея в распоряжении яркость всего фрагмента $<F_N>$ и яркость каждого из девяти участков $<f_n>$, входящих в данный фрагмент, мы будем кодировать каждый участок одним битом $c(f_n)$, получаемым следующим образом:

$$c(f_n) = \begin{cases} 1, если < I_{f_n} > меньше < I_F > \\ 0, в противном случае \end{cases} \quad (1)$$

Таким образом, бинарный код фрагмента F состоит из девяти бит $c(f_n)$, соответствующих девяти частям этого фрагмента изображения, и может быть получен по формуле:

$$BCode(F) = \sum_{n=0}^{8} c(f_n) \cdot 2^{8-n} \quad (2)$$

где BCode(F) и есть бинарный код фрагмента F, и он может принимать значения от 1 до 511. Данный бинарный код основан на Census-трансформации, описанной в [1]. Однако в нем имеются некоторые отличия:

- сравнение ведется со средней яркостью всей области и в преобразование входит центральный пиксель; таким образом, для кодирования области используется 9 бит, а не 8, как в Census-трансформации;
- используемое преобразование принципиально не локально, в него вовлечено большое количество пикселей и применяется оно сразу целиком к области изображения произвольного размера.

Преимущество нелокального подхода в том, что с использованием интегрального изображения (более подробно описано в [2]) можно вычислять яркость участков изображения всего за четыре арифметических действия. Это позволяет выделять сигнатуру изображения чрезвычайно быстро.

Сигнатура всего изображения строится иерархически. Иерархическое кодовое представление любого изображения получается посредством рекурсивного разбиения изображения на прямоугольные фрагменты (по четыре фрагмента на каждом шаге рекурсии) и кодирования каждого полученного фрагмента посредством описанного выше кода.

Для сравнения двух кодов участков изображений используется расстояние Хемминга - количество бит, которые нужно изменить, чтобы из одного кода получить другой.

За это расстояние возьмем расстояние Хэмминга – число позиций, в которых соответствующие цифры двух двоичных слов одинаковой длины различны.

На каждом уровне L дерева сигнатуры содержится $N_L=4^L$ бинарных кодов. Функция похожести между двумя сигнатурами на уровне L равна:

$$S(C_1, C_2, L) = \frac{9 \cdot N_L - \sum_{k=0}^{N_L-1} d(C_1(L,k), C_2(L,k))}{9 \cdot N_L} \quad (3)$$

где: C_1 и C_2 – сравниваемые сигнатуры; L – уровень сравнения; C(L, k) – k-й бинарный код уровня L сигнатуры C; d – функция расстояния между двумя бинарными кодами, т. е. расстояние Хемминга.

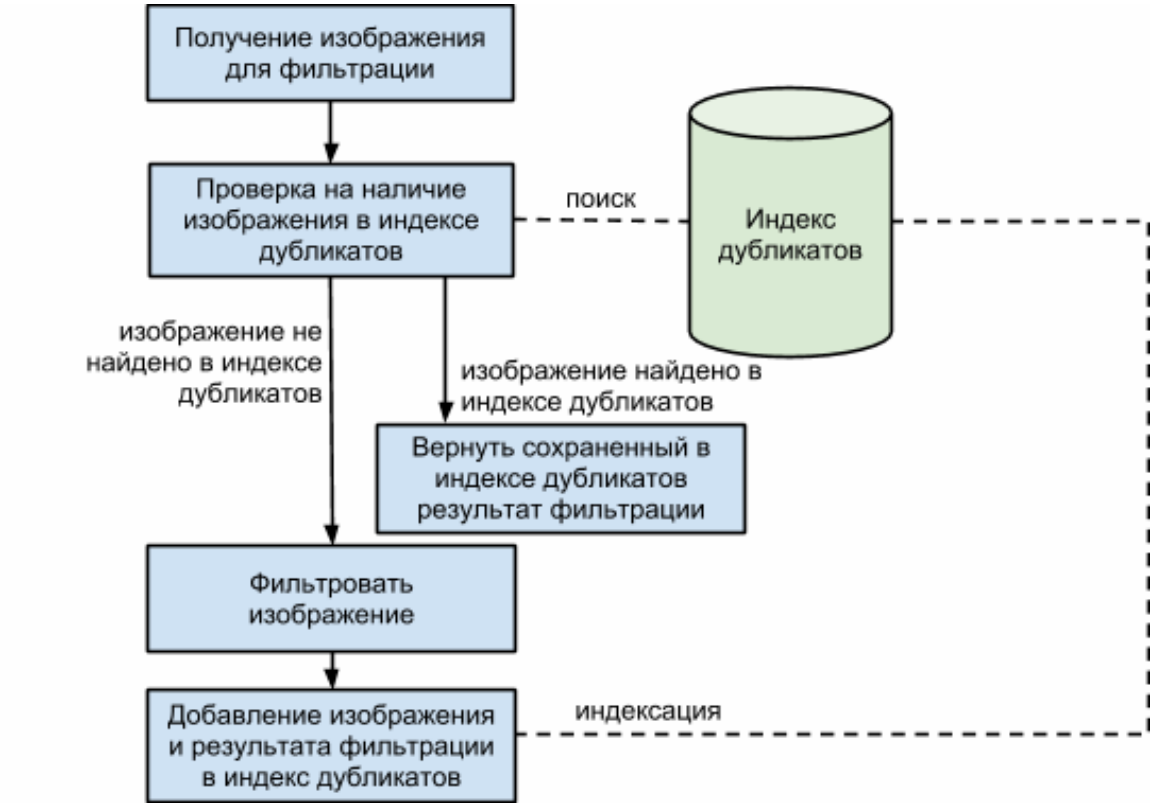


Рис. 2. Фильтрация запрещенного контента с проверкой дубликатов.

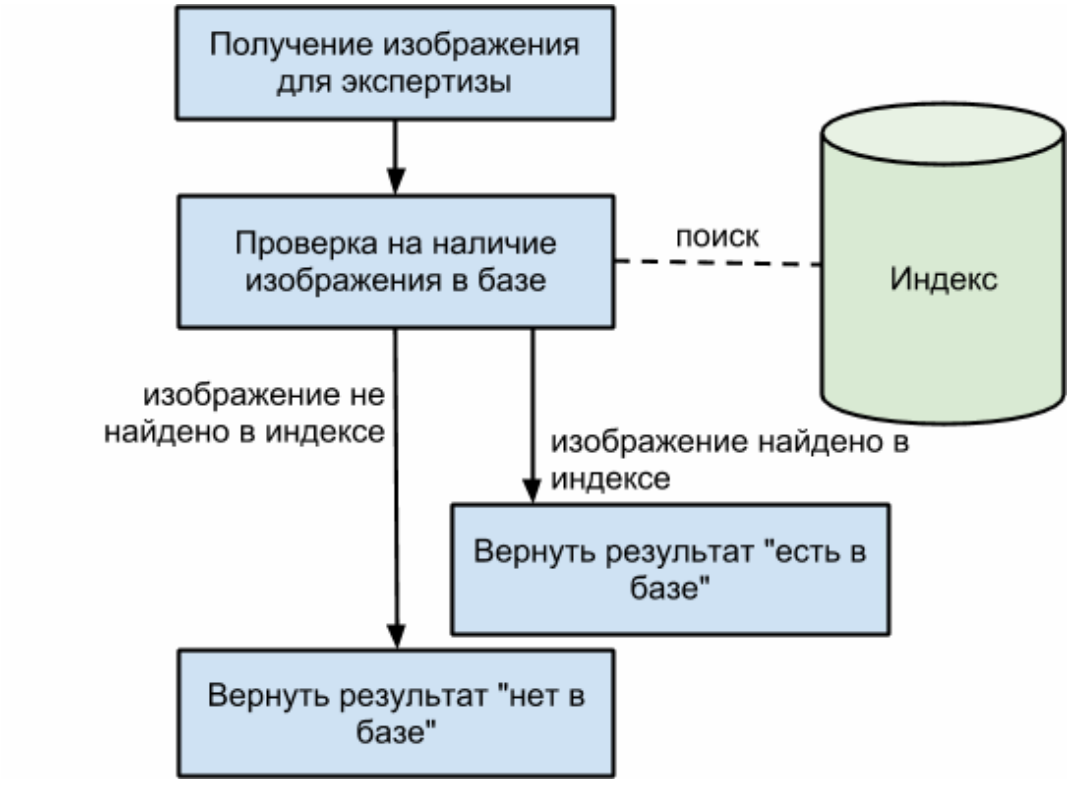


Рис. 3. Система помощи экспертам.

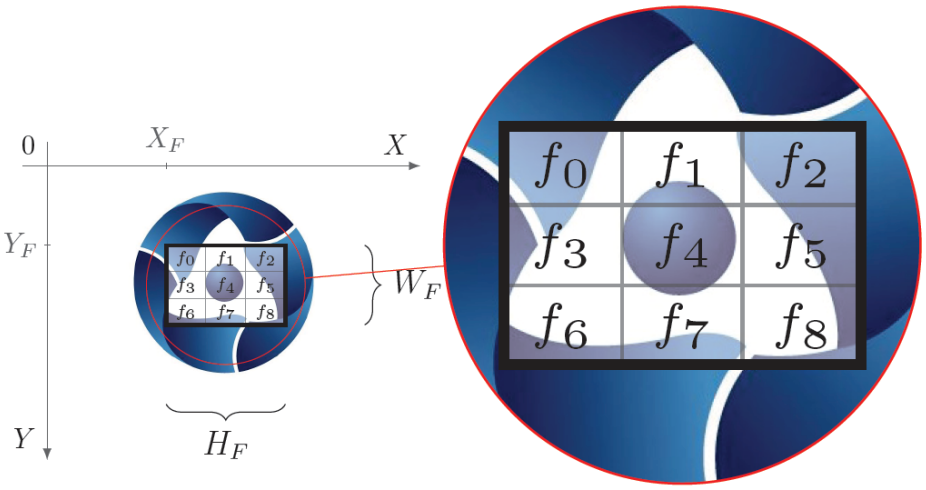


Рис. 4. Пример разбиения фрагмента на участки.

Функция похожести между сигнатурами на уровне L равна отношению совпавших бит во всех бинарных кодах уровня L к общему количеству бит в бинарных кодах этого уровня.

В качестве функции похожести между двумя сигнатурами возьмем наименьшее из всех значений функции похожести по каждому из доступных уровней:

$$S(C_1, C_2) = \min_{L_0 \ldots L_{\max}} (S(C_1, C_2, L)), \quad (4)$$

где L_{max} – максимальный из уровней сигнатур, который имеется как в сигнатуре C_1, так и в сигнатуре C_2.

Для сравнения сигнатур самих изображений коды участков изображений сравниваются по уровням и берется наименьший результат из всех уровней.

Отметим, что можно изменить функцию похожести так, чтобы получить инвариантность к зеркальным отображениям по горизонтали, по вертикали и по поворотам на углы, кратные 90 градусам.

Реализованный на основе описываемого алгоритма программный продукт позволил провести ряд экспериментов по оценке эффективности поиска дубликатов. Установлено, что время сравнения пары сигнатур составляет порядка 0,001 секунды. Время вычисления сигнатуры несколько больше - до 0,5 секунды на изображение для больших изображений. Алгоритм продемонстрировал устойчивость обнаружения дубликатов при появлении небольших (по относительной площади) добавлений в исходный объект.

ЗАКЛЮЧЕНИЕ

Рассмотренный в настоящей статье метод поиска нечетких дубликатов может применяться при организации различных баз данных изображений. Время извлечения сигнатуры изображения сокращается за счет использования интегрального изображения, тогда как время сравнения сигнатур двух изображений невелико вследствие линейной сложности алгоритма сравнения относительно длины сигнатуры и сравнительно малого размера этих изображений.

Отличительной особенностью предлагаемого алгоритма является способность успешно идентифицировать похожие изображения, различающиеся небольшими фрагментами, «баннерами».

СПИСОК ЛИТЕРАТУРЫ

1. Zabih R., Woodfill J. Non-Parametric Local Transforms for Computing Visual Correspondence // Proceedings Third European Conference Computer Vision, 1994. – P. 150-158.
2. Viola P., Jones M. Robust Real-time Object Detection // International Journal of Computer Vision. – 2004. – Vol. 57. – P. 137-154.
3. Jaimes A., Chang S.-F., Loui, A. C. Duplicate detection in consumer photography and news video // Proceedings International Conference on Multimedia, 2002. – P. 423–424.
4. Ke Y., Sukthankar R., Huston L. An Efficient Parts-based Near-Duplicate and Sub-Image Retrieval System // Proceedings of ACM International Conference on Multimedia, New York, USA, October, pp. 869-876, 2004.
5. Zhang D. Q., Chang S. F. Detecting Image Near-Duplicate by Stochastic Attributed Relational Graph Matching with Learning // Proceedings of ACM International Conference on Multimedia, October 2004. – New York, 2004. – P. 877-884,
6. Meng Y., Chang E. Y., Li B. Enhancing DPF for nearreplica image recognition // Proceedings of IEEE International Conference on Computer Vision, June 2003. – Wisconsin, 2003. – P. 416-423.
7. Foo J. J., Sinha R., Zobel J. Discovery of image versions in large collections // Proceedings of ACM International Conference on Multimedia Modeling, January 2007. – Singapore, 2007. – P.433-442.
8. Foo J. J., Zobel J., Sinha R., Tahaghoghi S. M. M. Detection of Near-Duplicate Images for Web Search // Proceedings of the 6th ACM international conference on Image and video retrieval. – New York, 2007. – P. 557-564.

9. Howarth P., Ruger S. M. Evaluation of Texture Features for Content-Based Image Retrieval // Proceedings of the 3-rd ACM international conference on Image and video retrieval, July 2004, Ireland. – Dublin, 2004. – P. 326–334.
10. Thomee B., Huiskes M. J., Bakker E. M., Lew Michael S. Large scale image copy detection evaluation // Proceedings of ACM International Conference on Multimedia Information Retrieval, October 2008, Canada. – Vancouver, 2008.
11. Xin Yang Qiang Zhu Kwang-Ting Cheng. Near-Duplicate Detection for Images and Videos Dept. of Electrical and Computer Engineering // Proceedings of University of California. – Santa Barbara, 2009. – P. 73-80.
12. Кисель Я. Алгоритм поиска нечетких дубликатов в коллекции изображений // Труды Российский семинар по оценке методов информационного поиска (РОМИП) 2007-2008. – М., 2008. – С. 170-173.
13. Bay H., Tuytelaars T., Van-Gool L. SURF: Speeded up robust features // Computer VisionandImage Understanding. – 2008. – № 3. – P. 346-359.

Материал поступил в редакцию 29.05.13.

Сведения об авторах

КАЛАФАТИ Юрий Дмитриевич - кандидат физико-математических наук, ген. директор ООО «ССТ-Технология контролируемого хаоса», Москва
e-mail: kalafati@controlchaostech.com

КОЗЛОВСКИЙ Павел Александрович – аспирант ИАТЭ НИЯУ МИФИ (Институт атомной энергетики Научно-исследовательского ядерного университета МИФИ), г. Обнинск
e-mail: theservy@gmail.com

СТАРКОВ Сергей Олегович – доктор физико-математических наук, зав кафедрой компьютерных систем сетей и технологий ИАТЭ НИЯУ МИФИ, г. Обнинск
e-mail: starkov@iate.obninsk.ru

ТЕЛЬНЫХ Александр Александрович – кандидат физико-математических наук старший научный сотрудник ИПФ РАН, г. Нижний Новгород
e-mail: telnykha@yahoo.com

ИНФОРМАЦИОННЫЙ АНАЛИЗ

УДК 81’37 – 047.44 : [002 : 004.89]

Г. А. Волкова

Принципы семантического анализа предметной области при создании информационно-аналитических систем

Рассмотрено использование онтологии предметной области как информационного ядра информационно-аналитической системы. Предложен подход к построению онтологии, базирующийся на формировании информационного обеспечения автоматизируемых процессов и определения окружения системы, на основе которых из предметной области выделяются основные сущности и отношения. Для систематизации знаний используется метод многоаспектной классификации. Предложен способ формирования системы нормативно-справочной информации информационно-аналитической системы с использованием онтологии предметной области.

Ключевые слова: *предметная онтология, онтология верхнего уровня, многоаспектная классификация, НСИ*

ВВЕДЕНИЕ

Проектирование информационно-аналитической системы, в независимости от платформы реализации, сопряжено с разработкой нормативно-справочной информации (НСИ). Во-первых, НСИ позволяет формализовать знания о предметной области, которые необходимы для хранения, консолидации и анализа данных информационной системы; во-вторых, НСИ решает проблему унификации данных, что позволяет повысить качество производимого системой анализа.

Основой системы нормативно-справочной информации служат взаимосвязанные справочники и классификаторы, описывающие основные сущности, участвующие в процессах, относящихся к предметной области информационной системы. Обычно система НСИ представляет собой реляционную модель и в полной мере не отражает связь между основными сущностями системы.

Для выявления взаимосвязей между сущностями предметной области, для которой разрабатывается информационная система, необходимо провести идентификацию предметной области [1], в ходе которой создается понятийная модель предметной области – онтология.

Онтология представляет собой концептуальную схему предметной области, состоящую из основных понятий предметной области (сущностей), сгруппированных посредством таксономии. Все отношения в таксономии задаются согласно отношениям, существующим в предметной области. Самым распространенным типом связи является обобщение, так называемая классификационная связь, которая обозначает отнесение объекта (сущности онтологии) к более высокому классу, например, «кота» к «животному». Разработчик вправе вводить в онтологию любые связи, специфичные для конкретной предметной области, что позволяет максимально приблизить создаваемую модель к «реальному миру».

Для каждой сущности может быть выделено неограниченное количество характеристических свойств. Если свойство может принимать только фиксированное число значений, как в случае дней недели, такие списки значений вводятся в онтологию в виде *слотов*.

Важным элементом онтологии являются аксиомы, или ограничения предметной области. Аксиомы представляют собой правила существования сущностей и отношений в рамках заданной предметной области, т.е. систему ограничений «того, чего в предметной области быть не может». Благодаря вводу аксиом у онтологии появляется возможность собственного логического вывода, что делает ее универсальным средством хранения знаний о структуре предметной области. Действительно, достаточно разработать интерпретатор конструкций логического вывода онтологии, чтобы использовать знания онтологии в любом программном приложении.

Существует множество подходов к построению онтологий [2], большинство из которых опираются на следующие основные этапы:

1) целеполагание (идентификация задач, которые должна решить создаваемая онтология, и определение критериев ее оценки);

2) построение таксономии понятий предметной области (выделение сущностей и связей);

3) спецификация формальной онтологии на языке представления знаний (формализация отношений между основными сущностями и спецификация аксиом);

4) оценка компетентности онтологии.

В настоящей работе предлагается подход к построению онтологии для создания информационного ядра информационной системы, ориентированный не только на спецификацию знаний о предметной области и основных задач, решаемых информационной системой [3], но и для создания интеграционной платформы, поддерживающей горизонтальную масштабируемость системы.

ОПИСАНИЕ ПОДХОДА

В зависимости от масштаба и степени начальной формализации предметной области, разработчик онтологии может столкнуться с множеством проблем [4], основные из них – сложность в установлении границ предметной области и определение необходимого уровня детализации. Большинство проблем вызваны недостаточным количеством и качеством нормативной документации, регламентирующей подлежащие автоматизации процессы, ее неполнотой и противоречивостью.

На начальных этапах создания необходимо четко представлять себе весь спектр задач, возлагаемых на информационную систему. Сформулируем два ключевых вопроса, на которые должен ответить разработчик онтологии, прежде чем преступать к полномасштабным действиям по ее созданию.

Для кого создается система и каков уровень управления системой? Под уровнем управления мы подразумеваем оперативное управление, корпоративное управление и управление федерального уровня. В зависимости от уровня управления могут различаться регламенты ведения справочников и классификаторов информационной системы, частота обновления данных в системе и требования к актуальности данных, уровень детализации информации. Обычно оперативное управление подразумевает уделение особого внимания техническому состоянию объектов, в связи с чем в онтологии делается акцент на мереологические связи и выделение свойств объектов управления. При этом создается модель эталонных состояний объектов, включающая в себя допустимые значения свойств объектов, например, нормативные сроки службы оборудования или допустимые давления в трубопроводах, если речь идет об автоматизации технологических процессов. Знания об эталонных состояниях предметной области вводятся в онтологию посредством аксиом.

Корпоративный уровень управления характеризуется большим количеством объектов управления, но с более низким уровнем детализации. Здесь особое внимание следует уделять классификационным связям и свойствам объектов управления, поскольку основной задачей корпоративных информационных систем является консолидация данных из различных источников и агрегация данных по определенным признакам, которые заранее могут быть не известны.

На федеральном уровне управления может быть минимальная детализация описания предметной области, но необходимо построение онтологии верхнего уровня, представляющей информационное окружение автоматизируемого процесса [5].

Каковы цели создания системы? Есть несколько обобщенных целей создания информационной системы, в зависимости от которых подход к построению онтологии должен быть различен:

- учет фактического состояния объектов управления подразумевает статическую модель предметной области;
- задача прогнозирования включает в себя создание как статической, так и динамической модели предметной области. Также создается модель внешних воздействий, оказывающих влияние на процессы, автоматизируемые информационной системой. Для задач прогнозирования, помимо классификационных и мереологических связей, вводится новый вид связи – причинно-следственная, выступающая связующим звеном между моделями предметной области и моделью внешних воздействий. Особое внимание уделяется необходимым и достаточным условиям для протекания автоматизируемых процессов и изменения состояния объектов управления;
- анализ с целью генерации рекомендаций об управляющих воздействиях базируется на принципах создания онтологии прогнозирования в сочетании с созданием эталонной модели предметной области. Поскольку для генерации рекомендаций информационная система должна иметь полную информацию об аналогах объектов управления, актуальной задачей становится создание онтологии верхнего уровня, позволяющей масштабировать систему при появлении новых технологий (и других изменений в окружении информационной системы). А также создание мощного аппарата онтологических аксиом, за счет которого будет происходить поиск аналогов объектов управления, обладающих большей эффективностью при тех же условиях.

Для дальнейшей разработки онтологии в рамках нашей работы предлагаются следующие этапы.

1. Идентификация предметной области (спецификация автоматизируемых процессов, смежных процессов и их информационного обеспечения).

2. Определение информационного окружения системы (определение внешних воздействий на автоматизируемые и смежные процессы, и формирование их информационного обеспечения).

3. Сборка онтологии (построение онтологии верхнего уровня, включающей в себя основные сущности информационного обеспечения, выделенные на этапах 1 и 2, построение предметных онтологий).

4. Сопровождение онтологии.

Остановимся на каждом этапе отдельно и опишем основные задачи разработчика на пути создания онтологии информационной системы.

Идентификация предметной области – решение этой задачи должно привести к выявлению основных элементов онтологии: сущностей (понятий), свойств сущностей и слотов (перечней возможных значений свойств), отношений между сущностями и аксиом

(устойчивых правил существования системы сущностей и отношений). Основными типами связей в онтологии являются классификационные и мереологические связи. Классификационная связь отражает отнесение сущности к некоторой категории (вышестоящему классу). Этот тип отношений также имеет ряд других названий, встречающихся в различных исследованиях: таксономическое отношение; отношение IS-A; класс – подкласс; гипоним – гипероним; родовидовое отношение; отношение a-kind-of и др. Данный тип связи используется во всех известных автору проектах, связанных с разработкой онтологий.

Но помимо классификационных связей, для построения адекватной модели нужны также мереологические отношения – т.е. отношения «часть-целое». Наличие такой связи между сущностями означает, что сущность является компонентом вышестоящего комплекса, но не отвечает на вопрос «только ли из этой сущности состоит вышестоящий объект». В некоторых случаях необходимо уточнение данной связи – разделение на мереологические композиционные и мереологические агрегационные связи. Отношение композиции показывает, что все сущности, для которых определена данная связь, являются составной частью вышестоящего объекта, но возможно наличие и других сущностей, которые не отражены в настоящий момент в онтологии. Отношение агрегации постулирует, что вышестоящий объект состоит только из сущностей, для которых определена данная связь.

Основной проблемой на этапе концептуализации предметной области является определение приоритета классификационных связей: построение онтологии может строиться в сторону увеличения количества классификационных связей и уменьшения количества свойств сущностей, или, напротив – в сторону максимального выделения свойств сущностей. Первый подход предполагает выделение дополнительного класса при выявлении некоего отличительного признака: предположим, что в онтологии уже присутствует сущность «стол», тогда «деревянный стол» становится подклассом данного класса. При втором подходе, методе многоаспектной классификации, в этом случае для сущности «стол» выделяется еще одно свойство – «материал», в списке возможных значений которого *(слоте)* появляется значение «дерево». Соответственно, если свойство «материал» у «стола» уже существовало, слот просто пополняется новым значением. Если необходимо выделить отдельный класс, классификационные признаки задают ветвление таксономии.

У метода многоаспектной классификации существует ряд недостатков [6], но этот метод позволяет дальнейшее использование онтологии при разработке информационной системы с использованием технологии OLAP, поскольку слоты, по сути, представляют собой измерения в OLAP-кубах, а сущности – сами кубы. Также плюсом данного метода является простота пополнения онтологии, так как появление нового свойства не отражается на структуре таксономии понятий.

В связи с этим в рамках настоящей работы автор предлагает использовать именно метод многоаспектной классификации, но с рядом ограничений:

1. К одному классу могут быть отнесены только сущности, имеющие одинаковый набор свойств. Если некоторой сущности необходима дополнительная классификация, для него должен быть создан отдельный класс.
2. Требуется анализ необходимости создания нового класса с точки зрения принципиальной его значимости. Если сущность с данным свойством участвует в другом процессе, то необходимо выделить сущность как отдельный класс. К примеру, нельзя объединять в один класс понижающие и повышающие трансформаторы, поскольку первые относятся к процессу «генерация электроэнергии», а вторые – к процессу «передачи».

Автоматизируемые и смежные процессы. Каждый элемент онтологии связан с элементом «сущность» и именно с выделения сущностей начинается идентификация предметной области. Базовый набор сущностей, необходимый для формирования онтологии информационной системы, ограничен участниками автоматизируемых процессов, причем как активными, так и пассивными. Рассмотрим самый простой вариант, когда целью создания информационной системы является учет фактического состояния объектов. В этом случае основным процессом является сам учет, а сущностями онтологии должны стать все объекты учета. Помимо учета, объекты наверняка покупают, модернизируют и где-то хранят, таким образом, мы получаем еще два смежных процесса – «инвестиционная деятельность» и «логистика».

Такой подход к первичному выделению сущностей подобен построению «пирамиды знаний» TOVE [7], которая включает уровень общих знаний: активности, процессы, ресурсы, время, причины. В проекте TOVE базовым этапом при построении онтологии является формулирование вопросов компетенции, т.е. постулирование требований к изобразительной мощности онтологии и возможности с ее помощью решать задачи, стоящие перед онтологией. Ядро онтологии TOVE содержит термины, выявленные на основе вопросов компетенции.

Аналогом вопросов компетенции в нашей работе является спецификация автоматизируемых процессов. В отличие от вопросов компетенции, выделение процессов и связей между процессами дает более полное представление о предметной области, чем спецификация задач, стоящих перед онтологией.

За этапом выделения автоматизируемых и смежных процессов следует этап сбора информации о процессах. Сбор данных необходим для определения информационного обеспечения процессов. Под информационным обеспечением мы будем понимать всю существующую нормативную документацию, которая описывает (полностью или частично) автоматизируемые и смежные процессы.

Информационное обеспечение процессов. Приведем список возможных источников информации при условии, что мы ведем разработку онтологии в России:

– *стандарты ГОСТа*. Практически в каждом документе ГОСТа есть раздел «термины и определения», который крайне полезен при выделении сущностей и связей между ними. Также из ГОСТа можно почерпнуть единицы измерений для свойств сущностей, представляющих собой физические величины, и нормативные значения этих свойств;

– *методики расчета*. Автоматизируемые и смежные процессы часто подвергаются некой внешней оценке, например, со стороны государства. Для оценки соответствия какой-либо системы согласно государственным ограничениям разрабатываются специальные методики, в которых указываются инспектируемые объекты, их признаки и критерии соответствия нормативным показателям. Анализ методик расчета позволяет выделить как основные сущности процессов, так и необходимый перечень свойств сущностей. Например, существует методика расчета нормативных потерь в тепловых сетях [8], из которой можно выделить сущности «тепловая сеть», «участок тепловой сети» и их свойства: для «тепловой сети» - «температура линии, С°» и «температура наружного воздуха, С°», для «участка тепловой сети» - «протяженность трубопроводов, м», «условный диаметр трубопровода, мм», «удельный объем воды в трубопроводе, куб. м/ км»;

– *паспорта объектов учета и оборудования*. На территории Российской Федерации существует стандартизация объектов инфраструктуры и оборудования в виде паспортов. Паспорт – это перечень значимых свойств некоторого объекта (без указания нормативных значений этих свойств). Примером инфраструктуры может служить многоквартирный жилой дом, для которого существуют энергетический паспорт здания, технический паспорт здания, кадастровый паспорт здания, паспорт приемки жилого здания, подготовленного к зимнему периоду, строительный паспорт на капитальный ремонт. Для каждого оборудования существует паспорт оборудования, который составляют заводы-изготовители. Паспорта объектов инфраструктуры и оборудования являются полноценным источником знаний для выделения свойств сущностей онтологии;

– *стандарты предприятий*. Информационная система создается для нужд конкретного предприятия, поэтому при разработке необходимо использовать нормативную документацию этого предприятия. Зачастую полезно использовать и стандарты других предприятий, осуществляющих тот же вид деятельности, поскольку они могут быть более полными и структурированными;

– *формы отчетности*. Обмен информацией внутри предприятия и с внешними агентами происходит обычно за счет формирования некой отчетности (оперативной, корпоративной, государственной). Из отчетности можно почерпнуть данные о ключевых сущностях предметной области, их свойствах и даже отношениях, анализируя структуры отчетных форм. В отчетности часто присутствуют строки «в том числе», представляющие собой мереологические связи.

Итак, материалы ГОСТа, методики расчета, паспорта оборудования и стандарты предприятия в области автоматизируемых процессов являются информационным обеспечением автоматизируемых процессов. Далее из информационного обеспечения согласно выбранному приоритету классификационных связей выделяются сущности. При именовании сущностей необходимо исходить из полноты и конкретности наименований.

Важнейшей задачей при анализе информационного обеспечения процессов является трассировка сущностей и свойств сущностей, т.е. их привязка к исходным документам. Для каждой выделенной сущности необходимо указать:

– источник информации, согласно которому она выделена;

– перечень свойств с указанием источников, на основании которых он сформирован;

– слоты с указанием документов, на основании которых они сформированы;

– для свойств, которые являются физическими величинами - единицы измерения и допустимые значения.

За счет трассировки элементов онтологии, возможно, будет в дальнейшем ее удобное сопровождение. Появление новых сущностей реального мира наверняка будет отражено в соответствующей документации, и при наличии трассировки добавить значение в онтологию не составит труда.

Аксиомы. Основным отличием онтологии от систем нормативно-справочной информации является возможность логического вывода информации за счет формирования аксиом. Целями введения в онтологию аксиом могут быть: определение комплексных ограничений на значения свойств, аргументы отношений и проверка корректности информации, описанной в онтологии. Также аксиомы задают условия соотнесения категорий и отношений. Основной аксиомой отношения классификации на языке предикатов первого порядка является:

$$\forall x.Bx \to Ax ,$$

т.е все объекты класса В входят в класс А. Несоблюдение этого условия приводит к противоречию и не является возможным в рамках онтологии:

$$\neg\exists x.Bx \,\&\, \neg Ax .$$

Аппарат аксиом позволяет устанавливать отношения соответствия между свойствами сущностей, что не может быть реализовано простым построением таксономии. Например, если для участка тепловой сети определено свойство «условный диаметр трубопровода, мм», и участок состоит из нескольких трубопроводов разных диаметров, то значение свойства «условный диаметр трубопровода, мм» для участка сети не может быть меньше самого малого диаметра.

Выявление аксиом частично происходит на этапе анализа информационного обеспечения процессов при работе с нормативными ограничениями свойств сущностей. Но и само выделение свойств сопряжено с не-

которыми ограничениями, например в случае, когда присутствует вложенность в классификации. Рассмотрим в качестве примера водогрейный котел. Он может работать на жидком, твердом и газообразном топливе, и в зависимости от этого параметра составные части котла и способ подготовки топлива могут быть различны. Однако есть список значений топлива: уголь, торф, газомазутное топливо, дизель, природный газ. В этом случае необходимо ограничить аксиомами, что у котла не может быть одновременно «вид топлива = жидкое топливо» и «тип топлива = уголь».

Определение информационного окружения системы. Для формирования ядра информационной системы как интеграционной платформы, необходимо предусмотреть ее расширение. Основной задачей информационного окружения является установление точек связи между онтологией информационной системы и внешними процессами, с которыми связана онтология. На этом этапе создания онтологии следует составить перечень всех возможных внешних воздействий, которые могут влиять на предметную область онтологии.

Зачастую информационное окружение можно определить, анализируя свойства сущностей, присутствующих в онтологии. Вернемся к методике расчета тепловых потерь, о которой говорилось в разделе «информационное обеспечение процессов». Там мы выделили сущность «тепловая сеть» и ее свойство «температура наружного воздуха, С°». В рамках информационной системы такое представление ничему не противоречит, но по сути «температура» есть свойство «внешней среды», которая оказывает воздействие на тепловую сеть. Мы можем поступить с этим примером разными способами: ввести в окружение сущность «температура внешней среды», или ввести сущность «внешняя среда» с ее свойством «температура». По мнению автора более целесообразно вводить концептуальные классы с большим количеством свойств, поэтому в этом случае предлагается пойти по второму пути, поскольку при необходимости внести в онтологию свойство «влажность воздуха» можно будет расширить уже существующий класс.

Для спецификации внешних воздействий необходимо выделить сущности, производящие воздействие и «точки воздействия», т.е. сущности онтологии информационной системы, на которые это воздействие направлено. В рассмотренном примере точкой связи между внешним воздействием среды и предметной областью информационной системы стала сущность «тепловая сеть». Наряду с температурой внешнего воздуха для тепловой сети можно определить также демографические воздействия, так как чем больше людей отапливает система, тем выше должна быть температура в сети и т.п.

Сборка онтологии. Для отражения всех базовых классов онтологии информационной системы создается онтология верхнего уровня, в которую должны войти:

- основные сущности предметной области, выделенные на этапе анализа информационного обеспечения автоматизируемых процессов;
- сущности, определенные как «точки воздействия» внешних воздействий;
- сущности, выявленные путем анализа окружения системы (внешние воздействия).

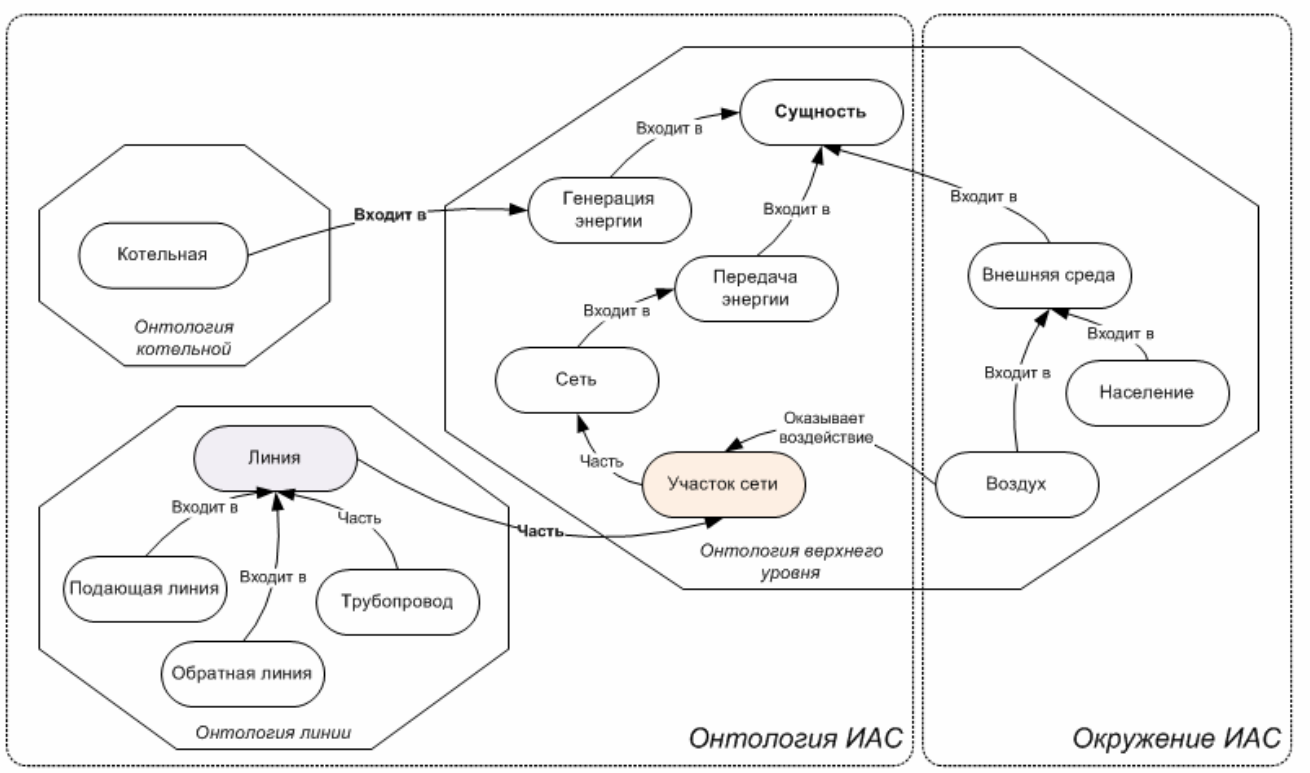


Рис. 1. Взаимосвязь онтологии верхнего уровня и онтологий более низких уровней

Онтология верхнего уровня является основой онтологии информационной системы и служит для взаимосвязи онтологий более низкого уровня. В нее включаются наиболее общие абстрактные классы, некоторые из них подразумевает декомпозицию - предметную онтологию. Обычно в онтологии верхнего уровня присутствуют только классификационные связи, а все мереологические связи разработчики спускают на уровень предметных онтологий для удобства их дальнейшего пополнения, так как онтология верхнего уровня разрабатывается с учетом окружения системы и не подлежит изменению (дополнению). На рис.1 схематично представлена онтология верхнего уровня с ее базовыми классами – «генерацией энергии», «передачей энергии» и «внешней средой» и отмечена «точка воздействия» окружения онтологии (на рис. 1 – «Окружение ИАС») на онтологию информационной системы (на рис. 1 – «Онтология ИАС»).

Следующим этапом сборки онтологии является создание предметных онтологий, количество которых не должно превышать количества сущностей в онтологии верхнего уровня, в противном случае – необходимо доработать онтологию верхнего уровня. Предметные онтологии строятся на основе сущностей и отношений, выявленных на этапе идентификации предметной области.

«Проверка данными» и сопровождение онтологии. Поскольку онтология является хранилищем знаний о предметной области, ее существование бессмысленно без экземпляров, т.е. объектов реального мира, систематизированных с помощью выделенных классов. Основным источником наполнения онтологии экземплярами являются отчетные формы, собранные на этапе формирования информационного обеспечения автоматизируемых процессов.

Первичное наполнение – важнейший этап создания онтологии, так как оно позволяет оценить компетентность построенной модели. Все объекты реального мира и вся информация о них должны «найти свое место» в созданной таксономии понятий. Зачастую на этапе наполнения происходит расширение свойств объектов. Но, если существует хотя бы один объект, который нельзя классифицировать с помощью понятийного аппарата онтологии, онтологию необходимо переработать.

После наполнения онтологии первичными данными начинается самый длительный процесс – сопровождение онтологии, включающий пополнение онтологии новыми экземплярами, выделение новых свойств объектов и определение качества информации, заносимой в онтологию, посредством аксиом.

Взаимосвязь онтологии и нормативно-справочной информации

Онтология информационной системы фактически содержит всю информацию, необходимую для формирования НСИ. На рис. 2 представлена взаимосвязь между онтологией и системой нормативно-справочной информации.

Рассмотрим подробней основные элементы НСИ и способы их формирования за счет существующей онтологии информационной системы. *Справочники НСИ* формируются на основе таксономии понятий предметных онтологий. Каждый класс предметной онтологии, подразумевающий наличие экземпляров (абстрактные классы исключаются по понятным причинам), может быть преобразован в справочник с соответствующим названием. Структура справочника задается автоматически согласно свойствам сущностей. Если имеется мереологическая связь, ведущая к другой сущности онтологии, в справочнике необходимо предусмотреть поле, в котором будет храниться информация о «родительском объекте», частью которого является текущий объект. Возможно формирование и справочников другой структуры – со ссылками на составные части объектов, но в этом случае необходимо внести в онтологию соответствующие аксиомы. Наполнение справочников происходит автоматически за счет переноса экземпляров и свойств экземпляров.

Классификаторы НСИ частично представляют собой слоты значений свойств сущностей онтологии. Но иногда их недостаточно, поскольку НСИ должна также содержать перечни агрегирующих признаков (на основе которых формируются отчетные формы), которые в онтологии присутствуют неявно.

Основой методики классификации и кодирования служит общая таксономия понятий онтологии верхнего уровня и всех предметных онтологий. В создании методики кодирования принимают участие только классификационные связи онтологии информационной системы. Удобнее всего использовать фасетное кодирование, принимая за первый уровень классы, входящие в базовый класс «сущность».

Для оценки размерности кода необходимо посчитать максимальное количество вложенных классов (найти самую длинную цепочку сущностей, связанных классификационными отношениями) и оценить максимальное количество дочерних классов (найти сущность с максимальным количеством «дочерних сущностей»). Также требуется посчитать количество классов онтологии, у которых нет «родителя» (т.е. не определена классификационная связь). Все вычисления делаются автоматически на основе аксиом. Минимальная длина кода определяется как XX*YY + a, где XX – максимальное количество вложенных классов, YY – размерность максимального количества дочерних классов, a – размерность количества количество классов онтологии, у которых нет «родителя».

Обращаясь к Рис. 1, находим, что «сущность» будет иметь код 1.0.0, следующий элемент, не имеющий родителя, – «участок сети» будет иметь код 2.0.0, «линии» присваивается код 3.0.0, «трубопроводу» – 4.0.0. Далее коды присваиваются следующему уровню понятий: «генерация энергии» – 1.1.0, «передача энергии» – 1.2.0, «подающая линия» – 3.1.0, «обратная линия» – 3.2.0. Третий уровень классификации дает нам коды 1.1.1 для «котельной» и 1.2.1 – для «сети».

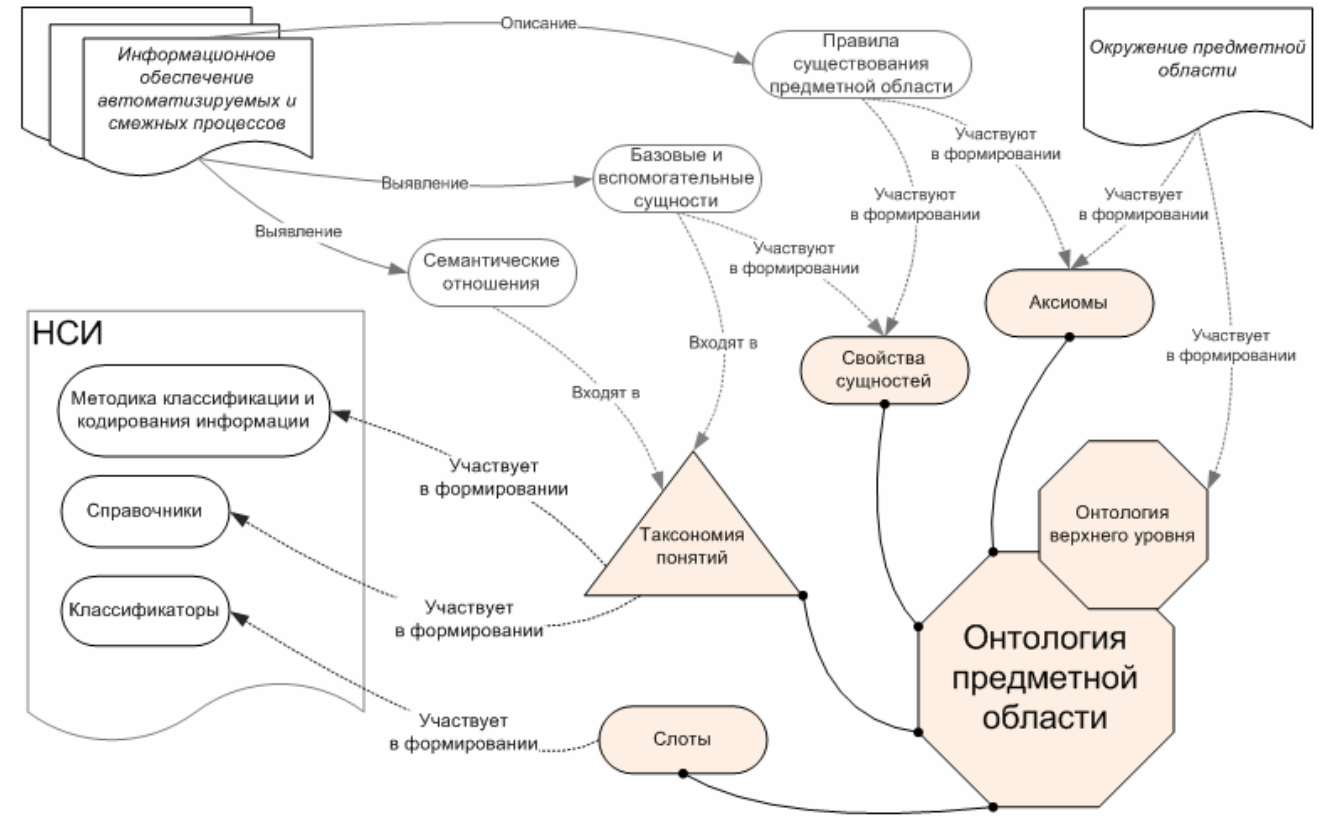


Рис. 2. Взаимосвязь онтологии и системы нормативно-справочной информации

Благодаря онтологии верхнего уровня и аппарата аксиом онтология становится хорошим средством навигации по отношениям между элементами НСИ. Возможности логического вывода онтологии позволяют описать алгоритмы расчета сложных интегральных величин и, при наличии интерпретатора конструкций логического вывода онтологии, формировать прямые запросы к хранилищу данных информационной системы.

ЗАКЛЮЧЕНИЕ

Первоочередной задачей при создании информационной системы является выделение и спецификация автоматизируемых процессов и объектов реального мира, которые в них вовлечены. Для систематизации и хранения знаний о предметной области и протекающих в ней процессах в нашей работе предложен подход к построению онтологии информационной системы. Создание онтологии решает вопрос унификации и систематизации данных на ранних этапах создания системы и служит базисом для создания системы нормативно-справочной информации.

Подход основан на формировании информационного обеспечения автоматизируемых процессов и определения окружения системы, на основе которых из предметной области выделяются основные сущности и отношения. Метод многоаспектной классификации, используемый при построении таксономии понятий, позволяет производить горизонтальное масштабирование онтологии без серьезных изменений в таксономии, а, следовательно, и в структуре хранилища данных, создаваемого на основе онтологии.

Направление дальнейших исследований. На момент написания статьи работа по созданию автором собственной онтологии находится на этапе формирования информационного обеспечения процессов, поэтому в статье не затронута проблема спецификации формальной онтологии на языке представления знаний. Планируется вести разработку на языке OWL (язык описания онтологий) с использованием онторедактора Protege. Результаты формализации онтологии станут темой следующих публикаций.

СПИСОК ЛИТЕРАТУРЫ

1. Мальцева С.В. Применение онтологических моделей для решения задач идентификации и мониторинга предметных областей // Бизнес-Информатика. – 2008. №3(05). – С. 18-24.
2. Волкова Г.А. Обзор методологий и методов построения онтологий с чистого листа // Материалы 3-й международной научно-практической конференции «Модель подготовки специалистов новой формации, адаптированных к инновационному развитию отраслей», г. Душанбе, 2-3 ноября 2012 г.
3. Ковалёв С.П. Применение онтологий при разработке распределенных автоматизированных информационно-измерительных систем // Автометрия. – 2008. – Т. 44, №2, – С. 41-49.
4. Волкова Г.А. Создание «онтологии всего». Проблемы классификации // Новые информационные технологии в автоматизированных системах: материалы шестнадцатого научно-практического семинара. – Моск. Ин-т элек-

троники и математики национального исследовательского университета «Высшая школа экономики». – М., 2013. – С. 293-300.

5. Карминский А.М., Черников Б.В. Информационные системы в экономике: в 2-х ч. Ч. 1. Методология создания: учеб. пособие. – М.: Финансы и статистика, 2006. – 336 с
6. Рубашкин В. Ш., Пивоварова Л. М. Методология наполнения онтологий — практика без теории? // Труды Второго Симпозиума «Онтологическое моделирование», г. Казань, 11–12 октября 2010 г. / ред. Л. А. Калиниченко. – М: ИПИ РАН, 2011.
7. Fox M.S., Chionglo J.C., Fadel F.G. A Common-Sense Model of the Enterprise // in 2nd IE Research Conference Proceedings, May 1993. – Los Angeles, 1993.
8. Приказ Министерства энергетики Российской Федерации от 30 декабря 2008 г. № 326 «Об организации в Министерстве энергетики Российской Федерации работы по утверждению нормативов технологических потерь электроэнергии при ее передаче по электрическим сетям».

Материал поступил в редакцию 31.05.13.

Сведения об авторе

ВОЛКОВА Галина Александровна – аспирант кафедры «Информационные технологии и автоматизированные системы» Московского института электроники и математики национального исследовательского университета «Высшая школа экономики» (МИЭМ НИУ ВШЭ)
e-mail: galina.volkova@eias.ru

УДК 002: 004.89

А.В. Бузмаков

Узорные структуры для анализа сложных последовательностей

Представлен метод поиска интересных паттернов в данных, описываемых сложными последовательностями, т. е. таких, у которых символы имеют структуру. Анализ формальных понятий и узорные структуры, относящиеся к прикладной теории решёток, позволяют решать рассматриваемую задачу в общем виде. Построение решётки узорных понятий представляет большую вычислительную сложность, поэтому исследуются возможности приближённого описания узорных структур, задаваемые через проекции. В статье проекции узорных структур используются не только для упрощения, но и для выделения паттернов, которые могут быть полезны для анализа предметной области. Возможность анализа таких проекций изучается на примере данных, задаваемых последовательностями госпитализаций пациентов.

Ключевые слова: *анализ формальных понятий, узорные структуры, анализ последовательностей*

ВВЕДЕНИЕ

Данные, представленные последовательностями, широко распространены в реальном мире – от анализа (майнинга) процессов [1] и анализа госпитализаций до анализа генов [2] и анализа текстов [3]. Анализ таких данных – важная и сложная задача. Существует несколько подходов к анализу последовательностей. Один из них основан на вычислении сходства или расстояния между двумя последовательностями [4, 5] с последующей кластеризацией; второй основан на перечислении частых подпоследовательностей на исходных данных с фокусом на эффективном перечислении и представлении [6–10]. Недостатком первого подхода является вовлечение только структурных элементов, общих для пары последовательностей, более того, анализ получающихся кластеров может быть достаточно сложен. Второй же подход обычно порождает очень большое количество паттернов (например, множество частых подпоследовательностей), из которых только малая часть представляет интерес. Для нахождения интересных паттернов используются различные индексы интересности, общая идея которых – нахождение статистически значимых паттернов, чьи характеристики, например частота, существенно отличаются от ожидаемых в случае нулевой гипотезы. В настоящей работе мы дополняем статистические методы алгебраическим подходом, основанным на прикладной теории решёток. Наш подход позволяет сфокусироваться только на тех паттернах, в которых эксперт может быть заинтересован, и затем среди них выбирать наиболее значимые путем анализа решётки формальных понятий. Анализ формальных понятий (АФП) [11] уже успешно использовался для анализа последовательностей [12, 13]. Более того, использование АФП и его расширения – узорных структур [14] – позволяет обрабатывать сложные последовательности, которые являются обобщением многих ранее изучавшихся. Так, например, подходы к анализу последовательностей, основанных на простом алфавите [13], последовательностей, каждый элемент которых является подмножеством некоторого множества [7–9, 12], а также последовательностей, в которых каждый элемент может содержать несколько измерений [15], являются частными случаями рассматриваемого подхода.

Проекции узорных структур позволяют эксперту перед анализом указать, в каких именно узорах он заинтересован, сокращая тем самым время обработки, используемую память и уменьшая сложность экспертного постанализа. Используя индекс устойчивости формального понятия, эксперт может анализировать полученные узоры в порядке ожидаемой значимости, позволяя быстро находить новые знания в данных, представленных последовательностями.

1. ОСНОВНЫЕ ОПРЕДЕЛЕНИЯ

1.1. Анализ формальных понятий (АФП)

АФП – это область прикладной теории решёток, методы которого используются для решения различных задач анализа и майнинга данных [11]. Приведём основные определения АФП согласно [11].

Определение 1. *Формальный контекст – это тройка (G, M, I) , в которой G – это множество объектов, M – множество признаков, $I \subseteq G \times M$ – бинарное отношение между G и M.*

В табл. 1 дан пример формального контекста. Между множествами подмножеств объектов и признаков можно задать соответствие Галуа с помощью следующих отображений:

Соответствие Галуа сопоставляет множеству объектов максимальное множество признаков, каждый из которых находится в отношении с каждым объектом.

Аналогично для множества признаков. Например, $\{g_1,g_2\}' = \{m_4\}$, в то время как $\{m_4\}' = \{g_1,g_2,g_4\}$. Соответствие Галуа лежит в основе формальных понятий и соответствующей решётки формальных понятий.

$$A' = \{m \in M \mid \forall g \in A, (g,m) \in I\}, \quad для\ A \subseteq G$$
$$B' = \{g \in A \mid \forall m \in M, (g,m) \in I\}, \quad для\ B \subseteq M\ .$$

Определение 2. *Формальное понятие – это пара (A, B) , где A – это подмножество объектов, $A \subseteq G$; B – признаков, $B \subseteq M$, причём $A'=B$, а $A=B'$. Множество объектов A называют объёмом, а множество признаков B – содержанием формального понятия (A, B).*

Примером формального понятия для контекста в табл. 1 является пара $(\{g_1,g_2,g_4\},\{m_4\})$, которой соответствует максимальное подмножество объектов, обладающих признаком m_4, в то время как мы не можем расширить множество признаков, не изменив множество объектов, соответствующих ему. Множество понятий упорядочено согласно теоретико-множественному включению объёмов или содержаний. Например, $(\{g_1\};\{m_1,m_4\}) \leq (\{g_1,g_2,g_4\},\{m_4\})$, так как $\{g_1\} \subseteq \{g_1,g_2,g_4\}$, или двойственно $\{m_4\} \subseteq \{m_1,m_4\}$. Данный частичный порядок является решёткой, т. е. для любой пары понятий существуют верхняя и нижняя грани. Рис. 1 показывает диаграмму решётки, соответствующей формальному контексту из табл. 1.

Существует несколько алгоритмов для нахождения множества формальных понятий, такие как 3O или CbO [16, 17], а также для нахождения решётки формальных понятий, такие как AddIntent [18]. Алгоритмическая сложность указанных алгоритмов составляет $O(|G||M||L|\min(|G|,|M|))$, где $|G|$ – количество объектов, $|M|$ – количество признаков, $|L|$ - конечный размер решётки. Стоит отметить, что размер решётки может быть экспоненциальным от числа объектов или признаков, точнее $2^{\min(|G|,|M|)}$.

Таблица 1

Простой формальный контекст

	m_1	m_2	m_3	m_4
g_1	x			x
g_2			x	x
g_3		x		
g_4			x	x

1.2. Устойчивость формальных понятий

Поскольку размер решётки формальных понятий может быть экспоненциальным по отношению к размеру контекста, для выбора наиболее важных понятий должны быть использованы различные индексы, такие как устойчивость [19, 20], вероятность или независимость понятия [21]. Авторы последней работы на примере случайных контекстов показали, что устойчивость более надёжна для зашумлённых данных, поэтому в настоящей работе будет использоваться именно она. В данной работе используется упрощённое определение устойчивости, согласно [22, 23], которое практически совпадает с устойчивостью по [19, 20].

Определение 3. *Устойчивостью формального понятия c (Stab(c)) называется отношение количества подмножеств объема понятия (c.Extent), описание которых совпадает с содержанием (c.Intent), к количеству подмножеств понятия. (Здесь и далее $\wp(P)$ означает множество всех подмножеств множества P).*

$$Stab(c) := \frac{\left|\{s \in \wp(c,Extent) \mid s' = c.Intent\}\right|}{\left|\wp(c.Extent)\right|} = \frac{\left|\{s \in \wp(c,Extent) \mid s' = c.Intent\}\right|}{2^{|c.Extent|}} \quad (1)$$

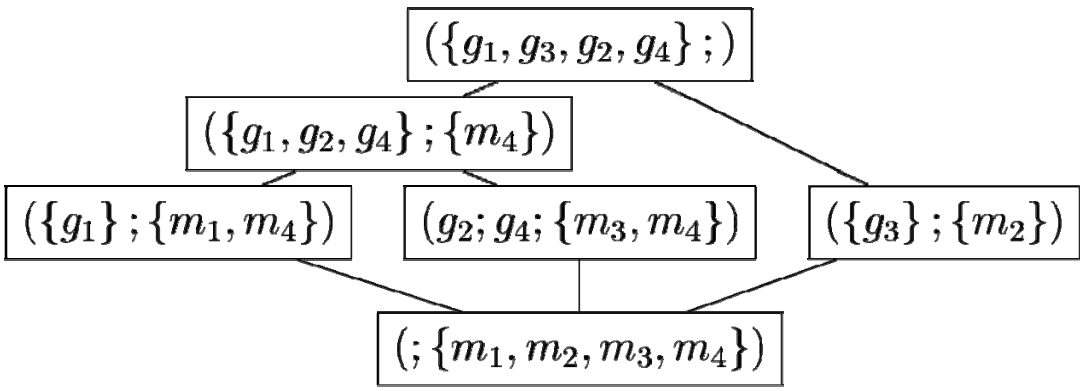


Рис. 1. Решётка понятий для контекста из табл. 1

Другими словами, индекс устойчивости показывает, насколько данное понятие зависит от выборки, по которой построен контекст. Чем больше индекс устойчивости, тем больше вероятность того, что при изменении выборки содержание рассматриваемого понятия будет присутствовать в решётке формальных понятий. То есть более вероятно, что содержание устойчивого понятия является характеристикой исследуемых объектов, чем случайным совпадением в выборке. На рис. 2 показана решётка формальных понятий для контекста из табл. 2; содержания понятий для простоты опущены, в квадратных скобках приведён соответствующий индекс устойчивости. Например, чтобы посчитать устойчивость для выделенного понятия, необходимо к каждому подмножеству объема ($\{g_1,g_2,g_3,g_4\}$) применить операцию $(\cdot)'$ и затем подсчитать количество тех подмножеств, чьё описание равно $\{m_6\}$. Среди всех подмножеств объема, только $\{g_1\},\{g_2\},\{g_3\},\{g_4\}$ и $\varnothing$ имеют описание, отличное от $\{m_6\}$, и, таким образом, устойчивость этого понятия равна $1-\frac{5}{2^4}=0.69$.

Таблица 2

Демонстрационный формальный контекст для расчёта устойчивости

	m_1	m_2	m_3	m_4	m_5	m_6
g_1	x					x
g_2		x				x
g_3			x			x
g_4				x		x
g_5					x	

Насколько нам известно, наилучший алгоритм [24] вычисления устойчивости для всей решётки имеет алгоритмическую сложность, в худшем случае квадратичную от размера решётки, $O(|L|^2)$. Как показывают компьютерные эксперименты, вычисление индекса устойчивости может занимать больше времени, чем вычисление самой решётки. Таким образом, эффективная оценка этого индекса является необходимой. Можно заметить, что если у понятия c, имеющего объём мощности $n=|c.Extent|$, есть наследник *child*, имеющий объём мощности $m=|child.Extent|$ (что означает, что $child.Extent \subseteq c.Extent$), то все подмножества $S \subseteq child.Extent$ не могут иметь описание (S'), равное содержанию родительского понятия. Однако, все подмножества объёма понятия *c*, которые не являются подмножествами ни одного наследника данного понятия, будут иметь то же самое описание, что и понятие *c*, и значит, должны включаться в числитель формулы 1. Таким образом, мы получаем следующую оценку для индекса устойчивости понятия:

$$1-\sum_{ch\in children} 2^{-Diff(c,ch)} \le Stab(c) \le 1-\max_{ch\in children}(2^{-Diff(c,ch)}) \quad (2)$$

где *Children* – это множество всех наследников данного понятия в решётке, а $Diff(c,ch) := |c.Extent \setminus ch.Extent|$ – разница в количестве объектов между понятием и некоторым его наследником. Например, все понятия с индексом устойчивости не менее 0.97 будут среди таких понятий, что

$$\max_{ch\in Children}(Diff(c,ch)) \ge -\log(1-0.97)-5.06 \quad (3)$$

В работах [19, 20] приводятся две оценки индекса устойчивости понятия. Первая дает оценку сверху и снизу при условии, что мы добавляем несколько объектов в формальный контекст и знаем точную меру устойчивости для начального контекста.

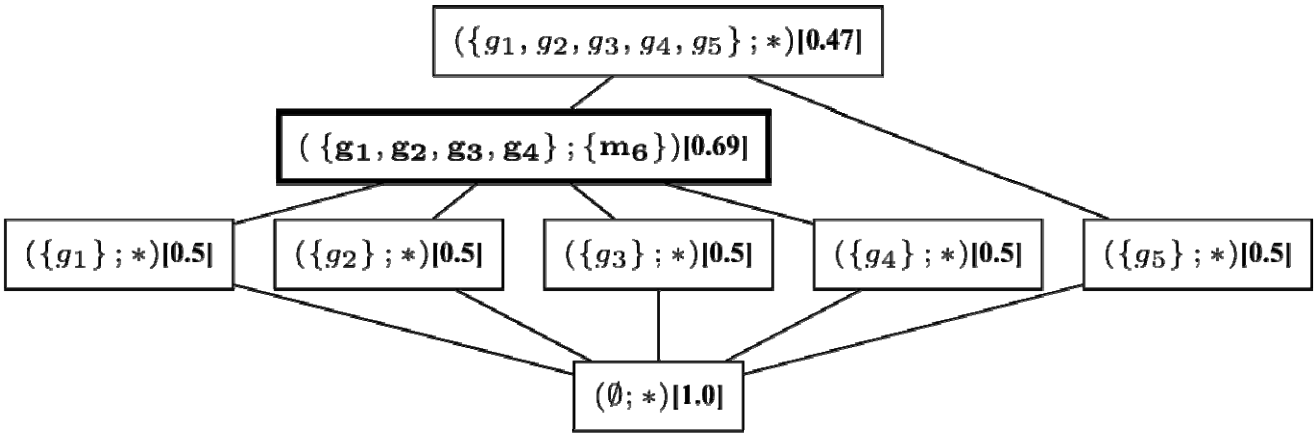


Рис. 2. Решётка формальных понятий для контекста, показанного в табл. 2, с соответствующими индексами устойчивости

Вторая оценивает индекс устойчивости путем рассмотрения подмножеств, имеющих определенную мощность. Первая оценка не применима в нашем случае, вторая же может иметь большую вычислительную сложность в силу необходимости анализа подмножеств определенной мощности. В работе [23] приводится оценка сверху, которая совпадает с оценкой в правой части неравенства (2), в то время как левая часть является одним из результатов нашей статьи. Другой подход к приближенному подсчету индекса устойчивости основан на статистическом методе Монте-Карло [25]. В этом методе случайным образом выбираются подмножества объема понятия, на основании которых оценивается устойчивость. В силу большого количества итераций метод может медленно работать в масштабах всей решетки. Приближенный подсчет устойчивости методом Монте-Карло может быть скомбинирован с методом, предложенным выше, для уточнения оценки устойчивости только тех понятий, которые представляют интерес.

1.3. Узорные структуры

АФП преобразует формальный контекст, представленный как бинарное отношение, в решётку формальных понятий, но во многих случаях исследуемые «объекты» могут иметь более сложное описание, чем множество некоторых наперед заданных признаков. Например, *исследуя множество объектов, возможно ли их исследовать без выделения специальных бинарных признаков?* Узорные структуры дают ответ на этот вопрос, являясь расширением АФП для работы со сложными данными [14], такими как данные, описываемые численными значениями, множествами последовательностей или графов.

Определение 4. *Узорная структура – это тройка* $(G, (D, \sqcap), \delta)$, *где* G *– множество объектов,* $(D, \sqcap)$ *– полная полурешётка всевозможных описаний, а* $\delta: G \to D$ *– функция, которая сопоставляет каждому объекту из множества* G *его описание из* D.

Полурешёточная операция $\sqcap$ соответствует операции сходства между двумя описаниями. Так, для описания формального контекста как узорной структуры перепишем формальный контекст следующим образом (см. рис. 3): каждому объекту в качестве описания запишем множество признаков, с которыми данный объект связан отношением I, и каждому уникальному множеству признаков дадим уникальное имя. Тогда соответствие между объектами и множествами признаков будет функцией δ , в то время как множество всех подмножеств множества признаков с операцией пересечения множеств является полной полурешёткой.

Соответствие Галуа между множествами объектов и множеством описаний перепишется для узорной структуры следующим образом:

$$A^{\rightleftarrows} := \prod_{g \in A} \delta(g), \text{ для } A \subseteq G$$

$$d^{\rightleftarrows} := \left\{ g \in G \middle| d \sqsubseteq \delta(g) \right\}, \text{ для } d \in D$$

Здесь $\sqsubseteq$ – это отношение поглощения, однозначно задающееся через полурешёточную операцию как:
а $a \sqsubseteq b \Leftrightarrow a \sqcap b = a$.

Определение 5. *Узорное понятие узорной структуры* $(G, (D, \sqcap), \delta)$ *– это пара* (A, d), *в которой* $A \subseteq G$ *– подмножество множества объектов,* $d \in D$ *– одно из описаний из полурешётки, такие что* $A^{\rightleftarrows} = d$ *и* $d^{\rightleftarrows} = A$, A *называется объёмом понятия, а* d *– узорным содержанием.*

Как и в классическом случае, объём понятия – это максимальное множество объектов, разделяющих одно описание, которое не может быть дальше уточнено.

Узорная структура может быть построена на основе произвольного частичного порядка. Во многих задачах на описаниях данных существует некоторый частичный порядок, соответствующий отношениям «часть–целое», «подкласс–класс». Например, в данных описываемых графами, естественный частичный порядок соответствует отношению подграф–граф и может быть применён для анализа молекулярных графов молекул, основанного на анализе подструктур [26, 27].

	m_1	m_2	m_3	m_4
g_1	x			x
g_2			x	x
g_3		x		
g_4			x	x

Контекст (G, M, I)

⇨

	Description	Name
g_1	$\{m_1, m_4\}$	d_1
g_2	$\{m_3, m_4\}$	d_2
g_3	$\{m_2\}$	d_3
g_4	$\{m_3, m_4\}$	d_2

Узорная Структура $(G, (\wp(M), \cap), \delta)$.

Рис. 3. Преобразование формального контекста в узорную структуру

Пусть дан некоторый частичный порядок (P, $\leq$), тогда соответствующая ему решётка (D, $\sqcap$) задаётся как множество подмножеств P таких, что если $p \in P$ принадлежит элементу решётки $d \in D$, то все меньшие элементы также принадлежат этому элементу решётки, $\forall p \in d, \nexists q \in P, q \leq p : q \notin d$. При этом решёточной операцией является теоретико-множественная операция пересечения. Нетрудно заметить, что результат такой операции между $d_1, d_2 \in D$ даст некоторый элемент из D. Стоит отметить, что на практике множество $d \in D$ может иметь существенный размер и поэтому его эффективнее и осмысленнее представлять максимальными элементами данного множества, $\tilde{d} = \left\{ p \in d \middle| \nexists q \leq d : q > p \right\}$. Более того, во многих случаях это представление позволяет эффективнее реализовывать решёточную операцию.

1.4. Проекции узорных структур

Поскольку размер решётки узорных понятий может быть существенным, а сама решёточная операция может быть вычислительно сложной (например, на узорной структуре на графах необходимо вычислять изоморфизм подграфов), время построения решётки узорных понятий может занимать существенное время. Для сокращения времени работы алгоритмов построения узорных решёток были введены проекции узорных структур [14]. Проекция может быть рассмотрена как некоторый фильтр полурешётки описания с определенными математическими свойствами. Эти свойства позволяют доказать, что для любого понятия в спроецированной решётке существует соответствующее понятие в исходной решётке. Более того, индекс устойчивости спроецированного понятия не может быть меньше, чем индекс устойчивости исходного понятия. Далее узорные структуры даются по [14].

Проекция узорной структуры – это функция $\psi : D \to D$, которая является монотонной ($x \sqsubseteq y \Rightarrow \psi(x) \sqsubseteq \psi(y)$), сжимающей ($\psi(x) \sqsubseteq x$) и идемпотентной ($\psi(\psi(x)) = \psi(x)$) [14]. Узорная структура проецируется следующим образом. Мы должны спроецировать функцию-описание объектов, а также полурешётку описаний так: $\psi((G,(D,\sqcap),\delta)) = (G,(D_\psi,\sqcap_\psi),\psi \circ \delta)$, где $D_\psi = \psi(D) = \left\{ d \in D \middle| \exists d^* \in D : \psi(d^*) = d \right\}$ и $\forall x, y \in D, x \sqcap_\psi y := \psi(x \sqcap y)$.

2. УЗОРНАЯ СТРУКТУРА НА ПОСЛЕДОВАТЕЛЬНОСТЯХ

Многие явления могут быть рассмотрены как последовательности, в которых одно состояние сменяется другим. В качестве примера рассмотрим историю госпитализаций пациентов, которая является важной задачей, позволяющей оптимизировать процесс лечения больных [28].

2.1. История госпитализаций пациентов

Каждый пациент посещает различные госпитали по различным причинам. В данной выборке каждая госпитализация представлена именем больницы, в которую пациент был помещён, и множеством медицинских процедур, которые он прошёл для завершения лечения. В табл. 3 представлена выборка из трёх пациентов, первый из которых имел четыре госпитализации в одной и той же больнице H_1 с разными наборами процедур в течение госпитализаций. Для данной выборки существует четыре медицинских процедуры $P = \{P_1, P_2, P_3, P_4\}$, а на именах больниц определена иерархия, содержащая принадлежность больницы к определенному типу, $T_H = \{H_1, H_2, H_3, H_4, CL, CH, *\}$, где, H_1 и H_2 являются государственными больницами (CH), а H_3 и H_4 – частными клиниками (CL). Общий тип для двух больниц $h_1, h_2 \in T_H$ обозначается как $h_1 \sqcap h_2$, например $H_1 \sqcap H_2 = CH$. Одна из важных задач – нахождение характеристических подпоследовательностей госпитализаций, которые могут быть найдены посредством анализа наиболее устойчивых понятий в решётке узорных понятий на последовательностях. В дальнейшем будут подробно рассмотрены узорные структуры на последовательностях, здесь же мы только отметим, что решётка узорных понятий, соответствующая нашему примеру, показана на рис. 4.

Таблица 3

Выборка историй госпитализаций пациентов

Пациент	История госпитализаций
p^1	$\langle [H_1, \{a\}]; [H_1, \{c, d\}]; [H_1, \{a, b\}]; [H_1, \{d\}] \rangle$
p^2	$\langle [H_2, \{c, d\}]; [H_3, \{b, d\}]; [H_3, \{a, d\}] \rangle$
p^3	$\langle [H_4, \{c, d\}]; [H_1, \{b\}]; [H_4, \{a\}]; [H_1, \{a, d\}] \rangle$

2.2. Частичный порядок на сложных последовательностях и соответствующая полурешётка

Последовательность состоит из элементов, которые принадлежат некоторому множеству, называющемуся алфавитом. В классическом случае в задаче нахождения подпоследовательностей на алфавит не накладывается каких-либо специальных свойств. В дальнейшем понятие подпоследовательности было обобщено на случай алфавита, в котором элементы являются подмножеством некоторого базового множества [12], а также на случай алфавита с многокомпонентными и многоуровневыми элементами [15]. Здесь предлагается обобщение на случай алфавита, являющегося произвольной полурешёткой[1]. Такой подход допускает произвольные последовательности, не имеющие связей между элементами одной последовательности, кроме последовательной связи, включая данные, представленные в подразделе 2.1. В частности, если на элементах введён некоторый частичный порядок, он может быть несложно преобразован в соответствующую полурешётку.

Определение 6. *Пусть дана полурешётка* $(E,\sqcap_E)$, *тогда*

1. *для любого элемента, отличного от* $\perp$, $e \in E, e \neq \perp, \langle e \rangle$ *является последовательностью;*
2. *для любой последовательности* $s = \langle e_1;...;e_n \rangle$ *и любого элемента, отличного от* $\perp$, $e \in E, e \neq \perp$, $s \circ e = \langle e_1;...;e_n;e \rangle$ *также является последовательностью.*

Определение 7. *Последовательность* $t = \langle t_1;...;t_k \rangle$ *является подпоследовательностью для последовательности* $s = \langle s_1;...;s_n \rangle$, *что обозначается как* $t \le s$, *тогда и только тогда, когда* $k \le n$ *и существуют* $j_1;...;j_k$ *такие, что* $1 \le j_1 < j_2 < ... < j_k \le n$, *а также для любого* $i \in \{1,2,...,k\}$, $t_i \sqsubseteq_E s_{j_i}$.

С таким определением подпоследовательностей задача поиска максимальных подпоследовательностей может оказаться вычислительно сложной, поэтому для упрощения задачи только «сплошные» подпоследовательности (или подстроки) принимаются во внимание. Под ними понимаются только подпоследовательности без пропусков (формально, для всех $i > 1$, $j_i = j_{i-1} + 1$). В дальнейшем слово «подпоследовательность» будет относиться к «сплошной» подпоследовательности, если не оговорено обратное.

В нашем примере (раздел 2.1.) алфавитом последовательностей является $E = T_H \times \wp(P)$, на которой решёточная операция задаётся как $(h_1,P_1)\sqcap(h_2,P_2) = (h_1 \sqcap h_2, P_1 \cap P_2)$, где $h_1,h_2 \in T_H$ – имена больниц, а $P_1,P_2 \in \wp(P)$ – множества применённых медицинских процедур. Таким образом, последовательность ss^1 в табл. 4 является подпоследовательностью первого пациента (p^1 в табл. 3), потому что для $j_i = i+1$ (определение 7) $ss_1^1 \sqsubseteq p_{j_1}^1$ ('CH' является более общим описанием, чем больница H_1, и при этом $\{c,d\} \subseteq \{c,d\}$), $ss_2^1 \sqsubseteq p_{j_2}^1$ (одна и та же больница и $\{b\} \subseteq \{b,a\}$), $ss_3^1 \sqsubseteq p_{j_3}^1$ ('*'означает любую больницу и является более общим описанием, чем больница H_1, $\{d\} \subseteq \{d\}$).

Как было отмечено в предыдущей главе, для произвольного частичного порядка может быть задана соответствующая полурешётка. В этом разделе был задан частичный порядок на «сложных» последовательностях, а значит и соответствующая полурешётка. На рис. 4 показана решётка узорных понятий, соответствующая узорной структуре на последовательностях, заданных в табл. 4. В дальнейшем для обозначения узорных структур на последовательностях будет использоваться аббревиатура УСП.

2.3. Проекции узорных структур на последовательностях

Решётка узорных понятий может быть трудновычислимой и, более того, среди всех понятий такой решётки только малая часть является полезной для анализа исследуемой выборки. Проекции позволяют уменьшить размер решётки, и при правильно заданной проекции из результирующей решётки могут быть исключены только те понятия, которые не могут являться интересными для решаемой задачи анализа. Ниже приводятся некоторые виды проекций, которые могут быть полезны для анализа с помощью УСП.

Во многих случаях эксперт может быть заинтересован в нахождении достаточно длинных подпоследовательностей, так, например, последовательности длины 1, как правило, не представляют интереса, так как не содержат «последовательных» зависимостей. Для такого проецирования узорной структуры необходимо заменить каждый элемент полурешётки описаний на элемент, из которого удалены все короткие последовательности. В дальнейшем такие проекции УСП называются «проекциями минимальной длины или ПМД-проекциями.

Пример 1. *Если эксперт считает, что в нашем случае интересны только последовательности длины 3 и более, то между пациентами* p^2 *и* p^3 *из табл. 3 есть только одна максимальная подпоследовательность* ss_6 *из табл. 4.*

Пример 2. *Рисунок 5а показывает спроецированную решётку узорных понятий, соответствующую УСП, заданной табл. 3 при ПМД-проекции при длине допустимых в узоре последовательностей от 3 и более.*

Предложение 1. *ПМД-проекция – это монотонная, сужающая и идемпотентная функция на полурешётке описаний.*

[1] Стоит отметить, что в общем случае эта полурешётка отличается от полурешётки, используемой в определении узорных структур

Таблица 4

Некоторые подпоследовательности историй госпитализаций из табл. 3

	Последовательности
ss^1	$\langle[CH,\{c,d\}];[H_1,\{b\}];[*,\{d\}]\rangle$
ss^3	$\langle[CH,\{\}];[*,\{d\}];[*,\{a\}]\rangle$
ss^5	$\langle[*,\{a\}]\rangle$
ss^7	$\langle[CL,\{d\}];[CL,\{\}]\rangle$
ss^9	$\langle[CH,\{c,d\}]\rangle$
ss^{11}	$\langle[*,\{c,d\}];[*,\{b\}]\rangle$

	Последовательности
ss^2	$\langle[CH,\{c,d\}];[*,\{b\}];[*,\{d\}]\rangle$
ss^4	$\langle[*,\{c,d\}];[*,\{b\}]\rangle$
ss^6	$\langle[*,\{c,d\}];[CL,\{b\}];[CL,\{a\}]\rangle$
ss^8	$\langle[CL,\{\}];[CL,\{a,d\}]\rangle$
ss^{10}	$\langle[CL,\{b\}];[CL,\{a\}]\rangle$
ss^{12}	$\langle[*,\{a\}];[*,\{d\}]\rangle$

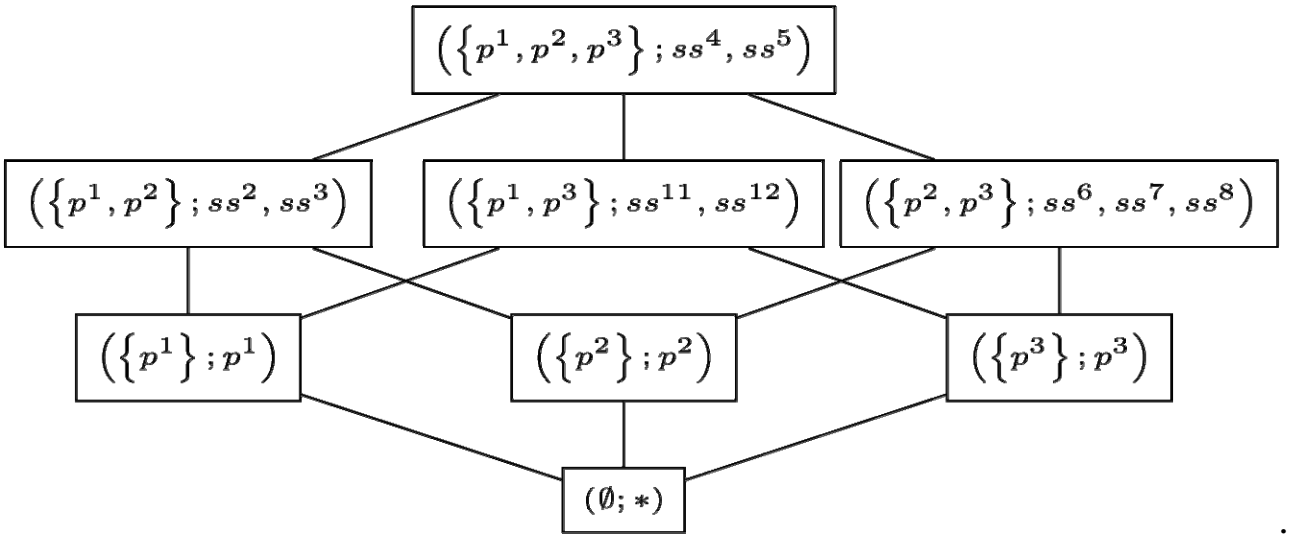
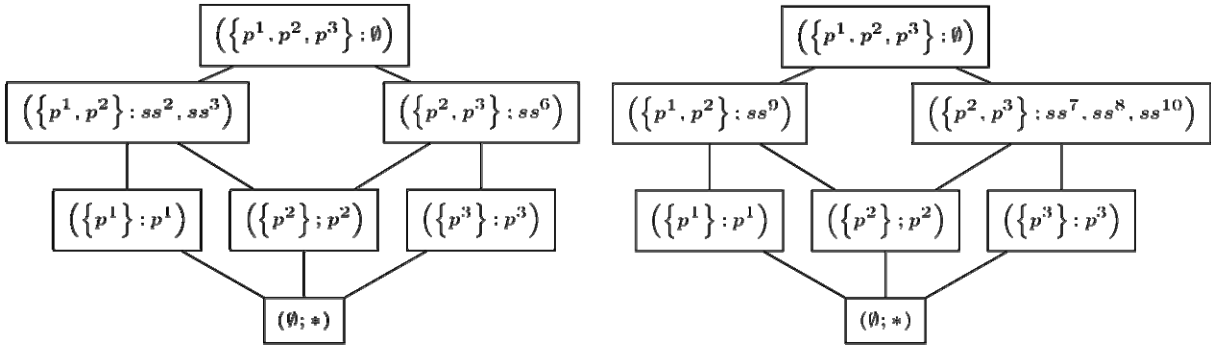


Рис. 4. Решётка узорных понятий для узорной структуры на последовательностях, заданной табл. 3. Содержания понятий ссылаются на последовательности из табл. 3 и 4



a: Проекция УСП с длиной не менее 3.

b: Проекция УСП, проецирующая алфавит (удалены описания с больницей '*')

Рис. 5. Спроецированные УСП для узорной структуры, заданной табл. 3. Узорное содержание ссылается на табл. 3 и 4

В силу того, что в сложных последовательностях алфавит является полурешёткой, возможно также применить проекцию и к алфавиту, *но будет ли это проекцией для УСП?*

Предложение 2. *Проекция алфавита УСП является монотонной, сужающей и идемпотентной функцией на полурешётке описаний.*

Такой вид проекций будем в дальнейшем называть алфавитной проекцией УСП. Здесь стоит отметить, что при проецировании алфавитной проекцией некоторые элементы полурешётки могут быть спроецированы на $\bot$. Согласно определению 6, такие элементы не могут быть элементом последовательности, что, в частности, означает, что при алфавитной проекции последовательности в узоре полурешётки должны быть «разрезаны» по элементам $\bot$.

Пример 3. *Эксперт хочет найти, как пациент меняет больницы в зависимости от процедур, которые он хочет пройти. В этом случае любой узор, в котором нет информации о больнице, является бесполезным (тип больницы – '*'), тогда можно ввести проекцию алфавита, в которой любой элемент полурешётки, который содержит больницу '*', должен быть спроецирован на $\bot$, все остальные элементы остаются без изменений. Так, для описанной проекции алфавита решётка узорных понятий примет вид, показанный на рис. 5b.*

3. ЭКСПЕРИМЕНТЫ И ОБСУЖДЕНИЕ

3.1. Аспекты реализации

Для построения решётки узорной структуры достаточно лишь немного изменить существующие алгоритмы построения решёток понятий. Теоретико-множественная операция пересечения должна быть заменена на полурешёточную операцию сходства, а все операции проверки того, что одно множество является подмножеством другого, должны быть заменены на проверку поглощения одного элемента полурешётки другим. Так алгоритм 1 показывает псевдокод алгоритма *Замыкай по одному (ЗО)* [16, 17], модифицированного для работы с узорными структурами. Для проведения экспериментов использовался алгоритм AddIntent, модифицированный по аналогии с *Замыкай по одному*. AddIntent, помимо понятий, порождает также и соответствующее отношение порядка на понятиях, необходимое для вычисления индекса устойчивости.

Теперь мы покажем как полурешёточные операции сходства и поглощения $(\sqcap, \sqsubseteq)$ могут реализовываться для ограниченных последовательностей. Пусть есть два множества последовательностей $S = \{s^1, ..., s^n\}$ и $T = \{t^1, ..., t^m\}$. Пересечение $S \sqcap T$ можно вычислить путём поиска максимальных последовательностей из всех максимальных общих подпоследовательностей для всех пар s^i и t^j.

Для нахождения всех общих подпоследовательностей между двумя последовательностями можно заметить, что, если $ss = \langle ss_1; ...; ss_l \rangle$ является подпоследовательностью $s = \langle s_1; ...; s_n \rangle$ при $j_i^s = k_s + i$ (см. определение 7) и является подпоследовательностью $t = \langle t_1; ...; t_m \rangle$ при $j_i^t = k_t + i$, то для любого индекса $i \in \{1, 2, ..., l\}$, $ss_i \sqsubseteq_E (s_{j_i}^s \sqcap t_{j_i^t})$. Таким образом, для нахождения всех максимальных подпоследовательностей для двух последовательностей, необходимо выровнять эти последовательности всеми возможными способами и описанным выше способом найти все максимальные общие подпоследовательности.

Ниже представлены два возможных выравнивания последовательностей s^1 and s^2.

Алгоритм 1. Версия ЗО, вычисляющая решётку узорных понятий.

Function `CloseByOne(`Ext, Int`)`
  Data: $(G, (D, \sqcap), \delta)$, **объем** Ext **и содержание** Int **некоторого понятия.**
  Result: **Все канонические предки** (Ext, Int) **в решётке понятий.**
  foreach $S \subseteq G$, $S \succ Ext$ **do**
    $NewInt \longleftarrow \prod_{g \in S} \delta(g)$; `/* ⊓ - пересечение */`
    $NewExt \longleftarrow \{g \in G \mid NewInt \sqsubseteq \delta(g)\}$; `/* ⊑ - поглощение */`
    if `IsCanonicExtension(`Ext, $NewExt$`)` **then**
      `SaveConcept((`$NewExt, NewInt$`));`
      `CloseByOne(`*NewExt,NewInt*`);`
`CloseByOne(`$\emptyset, \top$`);` `/* Находим все понятия решётки */`

Алгоритм 1: **Версия ЗО, вычисляющая решётку узорных понятий.**

Возможные выравнивания последовательностей s^1 and s^2

$$
\begin{array}{rl}
s^1 = & \langle \{a\}; \{c,d\}; \{b,a\}; \{d\} \rangle \\
s^2 = & \langle \{c,d\}; \{b,d\}; \{a,d\} \rangle \\
ss^l = & \langle \emptyset ; \{d\} \rangle
\end{array}
\qquad
\begin{array}{rl}
s^1 = & \langle \{a\}; \{c,d\}; \{b,a\}; \{d\} \rangle \\
s^2 = & \langle \{c,d\}; \{b,d\}; \{a,d\} \rangle \\
ss^r = & \langle \{c,d\}; \{b\}; \{d\} \rangle
\end{array}
$$

Результат пересечения для левого выравнивания ss^l не является максимальной общей подпоследовательностью, так как оно меньше правого ss^r, которое является одной из максимальных общих подпоследовательностей.

3.2. Компьютерные эксперименты

Эксперименты проводились на компьютере Intel(R) Core(TM) i7-2600 CPU @ 3.40GHz с 8 Гб оперативной памяти, работающем под операционной системой Ubuntu 12.04. Все применяющиеся алгоритмы реализованы на C++ без параллельных вычислений.

Исследуемая выборка данных построена по информации о госпитализациях пациентов за один год, с момента обнаружения у них раковых заболеваний. Выборка содержит около 2400 пациентов, а распределение длин показано на рис. 6. Описание одной госпитализации состоит из трех составляющих: имя больницы или клиники с ассоциированной таксономией расположения по городам и регионам, причина госпитализации (рак или химиотерапия) и множество медицинских процедур, которым подвергался пациент в течение этой госпитализации. Так, история некоторого пациента могла бы выглядеть следующим образом:

Данная последовательность моделирует траекторию пациента с тремя госпитализациями, первая из которых имеет место в больнице CH_1 и в течение которой диагностируется рак посредством процедур P_1 и P_2. В течение двух последующих госпитализаций пациент проходит курс химиотеропии в больнице CH_2. В большинстве случаев курс химиотерапии производится в одной больнице без каких-либо дополнительных процедур, поэтому для эффективной обработки узоров и для возможности исследовать вариацию числа госпитализаций для курса химиотерапии все подряд идущие госпитализации с химиотерапией объединяются в одну госпитализацию с дополнительным указанием числа повторов. Таким образом, представленная ранее история госпитализаций пациента будет иметь следующий вид:

$$\langle [CH_1, Рак, \{P_1, P_2\}]; [CH_2, Хим.Тер, \{\}]; [CH_2, Хим.Тер, \{\}] \rangle .$$

Для описанных данных естественной представляется следующая решёточная операция сходства: покомпонентное пересечение полей двух элементов последовательности, где пересечение полей с таксономией соответствует нахождению наименьшего общего предка в таксономии, для процедур – теоретико-множественное пересечение, а для количества повторов – интервал минимального размера, объединяющий оба значения.

$$\langle [CH_1, Рак, \{P_1, P_2\}]; [CH_2, Хим.Тер, \{\}][2] \rangle ,$$

Во многих работах [7–9, 12] каждый элемент последовательности является подмножеством некоторого множества. Такие последовательности представляют частный случай последовательностей, заданных определениями 6 и 7. В качестве первой части эксперимента посмотрим, какие зависимости может обнаружить предложенный подход в данных по траектории госпитализаций, представленных такими последовательностями. В этом случае невозможно рассматривать иерархии для территориального расположения и причин госпитализаций, так же как и информацию о числе повторов. Поэтому каждый элемент последовательности представляется как множество процедур, объединённое с именем больницы (нижний элемент иерархии территориального расположения), с причиной госпитализации (нижний элемент иерархии причин госпитализации).

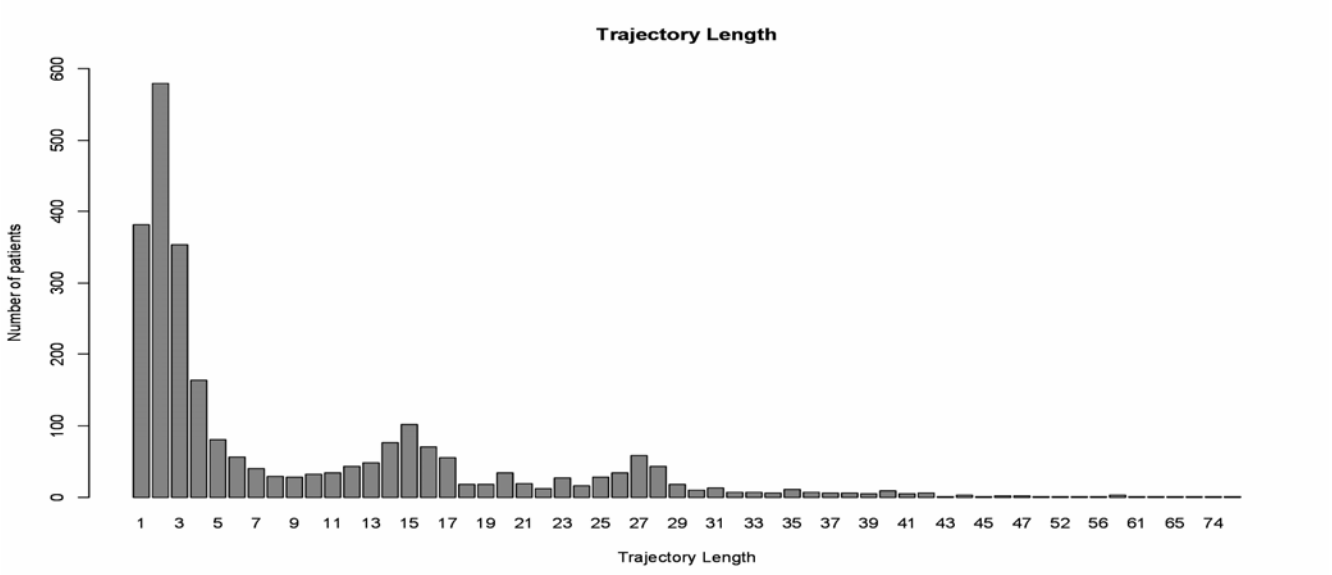


Рис. 6. Распределение количества госпитализаций за год для пациентов выборки

На таком описании данных строится решётка узорных понятий для разных ПМД-проекций. Табл. 5 показывает время построения и размеры решёток для разных проекций. Так, например, для проекции, допускающей только последовательности не короче двух элементов ($l = 2$), решётка строится за 4510 секунд и состоит из 554332 понятий. Решётка для неспроецированных данных не была построена до конца, так как вычисления занимали очень большое время. Здесь мы можем заметить, что ПМД-проекции несильно отличаются друг от друга по времени вычисления. Табл. 6 показывает содержания некоторых интересных понятий, выделенных с помощью индекса устойчивости. Так понятие c_1 содержит 8 последовательных госпитализаций для прохождения химиотерапии, объем которого включает 28% исследуемых пациентов, а понятие c_2 содержит 12 госпитализаций для прохождения химиотерапии, следующих за процедурой подготовки к курсу химиотерапии. Оба понятия позволяют оценить количество процедур в курсе химиотерапии и могут быть использованы для оптимизации процессов в больницах. Понятие c_3 описывает тех пациентов, которые последовательно подвергались двум радиографиям, что, вероятно, свидетельствует о контроле за состоянием больного. Во второй части эксперимента будет показано, как более общее определение последовательностей помогает находить похожие, но более интересные зависимости.

Во второй части эксперимента будут рассматриваться последовательности с элементами произвольного типа. Таким образом, возможно учесть всю информацию, доступную для анализа, такую как различные иерархии и количество повторов. В таком случае для исследуемой выборки построение решётки узорных понятий занимает очень много времени и памяти. Тем не менее, поскольку перед экспертом, как правило, стоит конкретная проблема, которую он хочет решить, будут применяться проекции УСП, которые соответствуют его задачам. Более того, ПМД-проекции позволяют отсекать неинтересные последовательности длины 1.

Табл. 7 показывает времена построения решёток узорных понятий, количество понятий в этих решётках, а также количество устойчивых понятий для различных типов проекций. Так, первый столбец соответствует проекции Ц!П2, для которой решётка была построена за 18 секунд, в этой решётке около 34700 понятий, среди которых устойчивых только 615. В имени проекции Р соответствует расположению больницы, Ц – цели госпитализации, П – множеству медицинских процедур, а И – интервалам изменений количества госпитализаций на курс химиотерапии. Число в самом конце задаёт минимально допустимую длину подпоследовательности, т. е. параметр ПМД-проекции. Ниже каждая из них рассмотрена подробнее. Табл. 8 показывает содержания некоторых интересных понятий, полученных для определённых проекций, с соответствующими поддержкой и рангом устойчивости (порядковым номером по индексу устойчивости).

Покажем, какие задачи эксперт может решать путём задания определенных проекций. Так, первая задача – определить *существующие последовательные связи причин госпитализаций и соответствующих процедур*. Здесь эксперт не интересуется территориальным расположением больницы, а также сначала ему не интересно и количество повторений химиотерапии. Более того, если он хочет исследовать последовательные связи причин госпитализаций, то любой узор с причиной «любая» является не интересным для эксперта. Таким образом, рассматриваемая проекция будет носить имя Ц!П, в которой мы интересуемся целью госпитализации, которая не может быть «любой» (Ц!), а также соответствующим набором процедур (П). Узор *#1* из табл. 8 отмечает типичную для раковых больных закономерность: если у пациента обнаружен рак, то он подвергается химиотерапии. Это самое устойчивое понятие, с существенной поддержкой. Узоры *#2* и *#3* описывают одно и то же явление, в котором утверждается, что рак у пациентов может быть найден во время операции аппендицита, что подтверждается медицинской литературой, с последующим лечением химиотерапией. Отметим, что узоры *#2* и *#3* отличаются по устойчивости, потому что узор *#2* был получен при ПМД-проекции, допускающей узоры длины 2, в то время как узор *#3* был получен при ПМД-проекции, допускающей только узоры длины 3 и более. Далее, если эксперт интересуется количеством повторений химиотерапии, то он может включить информацию о числе повторений в алфавит последовательностей, образую новую проекцию Ц!ПИ. Так, узор *#4* показывает, что в значительном числе случаев пациенты имеют от 8 до 24 последовательных госпитализаций для курса химиотерапии. Тут стоит отметить, что время, затраченное на вычисление решётки узорных понятий для проекции Ц!ПИ3, существенно превышает время построения решётки для проекции Ц!П3. Таким образом, правильно подобранная проекция не только упрощает анализ узоров, но и существенно влияет на время построения решётки.

Таблица 5

Время выполнения и размеры решёток для траекторий госпитализаций, описанных подмножествами

ПМД-проекция	Нет	$l = 2$	$l = 3$	$l = 4$	$l = 5$	$l = 6$
Время (сек.)	$> 1.4 * 10^5$	4510	3878	3722	3435	3120
Решётка	$> 2.8 * 10^6$	554332	271166	189353	137912	100492

Таблица 6

Интересные понятия для траекторий госпитализаций, описанных подмножествами

	Содержание	Объем
c_1	$\langle\{$**Хим.Тер.**$\} * 8\rangle$	28%
c_2	$\langle\{$**Хим.Подготовка**$\}; \{$**Хим.Тер.**$\} * 12\rangle$	7%
c_3	$\langle\{$**Радиография**$\} * 2\rangle$	19%

Хим.Тер. *означает химиотерапию*, Хим.Подготовка – *подготовку к химиотерапии*.

Таблица 7

Результаты экспериментов для разных типов проекций

Тип проекция	Ц!П2	Ц!П3	Ц!ПИ2	Ц!ПИ3	Р!Ц!2	Р!Ц!3
Время работы (s)	18	8	417	15	10	6
Число понятий (в тысячах)	34.7	8.67	1856	93.2	4.2	2.2
Число устойчивых($Stab \geq 0.97$) понятий	615	192	1117	311	131	45

Таблица 8

Содержания интересных понятий из решёток для разных проекций

#	Проекция	Содержание	Устойчивость	Поддержка
1	Ц!П2	⟨[Рак. {}]; [Хим.Тер.. {}]⟩	1	452
2	Ц!П2	⟨[Рак. {Апп.}]; [Хим.Подготовка. {}]; [Хим.Тер.. {}]⟩	4	293
3	Ц!П3	⟨[Рак. {Апп.}]; [Хим.Подготовка. {}]; [Хим.Тер.. {}]⟩	2	293
4	Ц!ПИ3	⟨[Рак. {}]; [Хим.Подготовка. {}]; [Хим.Тер.. {}] * [8. 24]⟩	4	193
5	Р!Ц!3	⟨[Регион А,Рак]; [Регион А,Хим.Подготовка]; ⋯ ⋯ [Конкретная больница в А,Хим.Тер.]⟩	5	29

Хим.Тер. *означает химиотеропию,* Хим.Подготовка – *подготовку к химиотерапии,* Апп. – *оперативное лечение аппендицита.*

Другая возможная задача эксперта – найти, *как пациенты предпочитают проходить своё лечение*. Так, проекция Р!Ц! оставляет в каждом узоре только информацию о территориальном расположении больницы и соответствующей причине госпитализации, где оба поля не могут быть пустыми. Узор *#5* – один из узоров, полученных в такой проекции. Этот узор отмечает тот факт, что пациенты по каким-то причинам предпочитают проходить химиотерапию в конкретной больнице региона А, независимо от того, в какой больнице этого региона у них обнаружили рак. Возможно, что это единственная больница региона, которая может проводить химиотерапию, либо качество услуг в данной больнице существенно выше, чем у конкурентов.

ЗАКЛЮЧЕНИЕ

В статье изучены теоретические и практические аспекты узорных структур на последовательностях. Использование формализма УСП позволило обобщить понятие подпоследовательности и эффективно обрабатывать данные с использованием введенных определений. УСП могут применяться на различных данных, представленных последовательностями, и позволяют выделять статистически-значимые зависимости, которые оказываются полезными для эксперта и для решения поставленной перед ним задачи.

В данной работе рассматривались узорные структуры со сплошными подпоследовательностями, что является некоторым ограничением. В дальнейшем планируется провести исследование подпоследовательностей в полном соответствии с их определени-

ем. Другим важным направлением исследования является вопрос построения и анализа ассоциативных правил на узорах.

СПИСОК ЛИТЕРАТУРЫ

1. van der Aalst W. Process mining: Discovery, conformance and enhancement of business processes. – Springer, 2011. – pp. I–XVI, 1–352.
2. Deshpande M., Karypis G. Evaluation of Techniques for Classifying Biological Sequences // Advances in Knowledge Discovery and Data Mining SE - 41 / ed. by M.-S. Chen, P. Yu, B. Liu. – Springer, Berlin, Heidelberg, 2002. – Vol. 2336 of *Lecture Notes in Computer Science.* – pp. 417–431.
3. Zhong N., Li Y., Wu S.-T. Effective Pattern Discovery for Text Mining // IEEE Transactions on Knowledge and Data Engineering. – 2012 – Vol. 24, № 1. – P. 30–44.
4. Li M., Badger J. H., Chen X. et al. An information-based sequence distance and its application to whole mitochondrial genome phylogeny // Bioinformatics. – 2001. – Vol. 17, № 2. – P. 149–154.
5. Cilibrasi R. L., Vitanyi P. M. B. The Google Similarity Distance // IEEE Transactions on Knowledge and Data Engineering. – 2007. – Vol. 19, № 3. – P. 370–383.
6. Han J., Pei J., Mortazavi-Asl B. et al. FreeSpan: frequent pattern-projected sequential pattern mining // Proceedings of the sixth ACM SIGKDD international conference on Knowledge discovery and data mining. – 2000. – P. 355–359.
7. Pei J., Han J., Mortazavi-Asl B. et al. PrefixSpan Mining Sequential Patterns Efficiently by Prefix Projected Pattern Growth // 17th International Conference on Data Engineering. – 2001. – P. 215–226.
8. Yan X., Han J., Afshar R. CloSpan: Mining Closed Sequential Patterns in Large Databases // SDM. – 2003. – P. 166–177.
9. Ding B., Lo D., Han J. et al. Efficient Mining of Closed Repetitive Gapped Subsequences from a Sequence Database // 2009 IEEE 25th International Conference on Data Engineering. – IEEE, 2009. – P. 1024–1035.
10. Räıssi C., Calders T., Poncelet P. Mining conjunctive sequential patterns // Data Min. Knowl. Discov. – 2008. – Vol. 17, № 1. – P. 77–93.
11. Ganter B., Wille R. Formal Concept Analysis: Mathematical Foundations. – 1st edn. – Secaucus, NJ, USA: Springer, 1997. – P. I–X, 1–284.
12. Casas-Garriga G. Summarizing Sequential Data with Closed Partial Orders // SDM. – 2005.
13. Ferré S. The Efficient Computation of Complete and Concise Substring Scales with Suffix Trees // Formal Concept Analysis SE - 7 / ed. by S. O. Kuznetsov, S. Schmidt. – Springer, 2007 – Vol. 4390 of *Lecture Notes in Computer Science.* – P. 98–113.
14. Ganter B., Kuznetsov S. O. Pattern Structures and Their Projections // Conceptual Structures: Broadening the Base SE - 10 / ed. by H. Delugach, G. Stumme. – Springer, Berlin, Heidelberg, 2001. – Vol. 2120 of *Lecture Notes in Computer Science.* – P. 129–142.
15. Plantevit M., Laurent A., Laurent D. et al. Mining multidimensional and multilevel sequential patterns // ACM Transactions on Knowledge Discovery from Data. – 2010 – Vol. 4, № 1. – P. 1–37.
16. Kuznetsov S. O. A fast algorithm for computing all intersections of objects in a finite semi-lattice // Automatic documentation and Mathematical linguistics. – 1993. – Vol. 27, № 5. – P. 11–21.
17. Kuznetsov S. O. Mathematical aspects of concept analysis // Journal of Mathematical Sciences. – 1996. – Vol. 80, № 2. – P. 1654–1698.
18. Merwe D. V. D., Obiedkov S., Kourie D. AddIntent: A new incremental algorithm for constructing concept lattices // Concept Lattices / ed. by G. Goos, J. Hartmanis, J. Leeuwen et al. – Springer, 2004. – Vol. 2961. – P. 372–385.
19. Kuznetsov S. O. Stability as an Estimate of the Degree of Substantiation of Hypotheses on the Basis of Operational Similarity // Automatic documentation and Mathematical linguistics. – 1990 – Vol. 24, № 6. – P. 62–75.
20. Kuznetsov S. O. On stability of a formal concept // Annals of Mathematics and Artificial Intelligence. – 2007. – Vol. 49, № 1–4. – P. 101–115.
21. Klimushkin M., Obiedkov S. A., Roth C. Approaches to the Selection of Relevant Concepts in the Case of Noisy Data // Proceedings of the 8th international conference on Formal Concept Analysis – ICFCA'10. – Springer, 2010. – P. 255–266.
22. Roth C., Obiedkov S., Kourie D. Towards concise representation for taxonomies of epistemic communities // Proceedings of the 4th international conference on Concept lattices and their applications – CLA'06. – Berlin, Heidelberg: Springer-Verlag, 2006. – P. 240–255.
23. Kuznetsov S., Obiedkov S., Roth C. Reducing the Representation Complexity of Lattice-Based Taxonomies // Conceptual Structures: Knowledge Architectures for Smart Applications SE - 18 / ed. by U. Priss, S. Polovina, R. Hill. – Springer, Berlin, Heidelberg, 2007 – Vol. 4604 of *Lecture Notes in Computer Science.* – P. 241–254.
24. Roth C., Obiedkov S., Kourie D. G. On succinct representation of knowledge community taxonomies with formal concept analysis. A Formal Concept Analysis Approach in Applied Epistemology // International Journal of Foundations of Computer Science. – 2008 – Vol. 19, № 02. – P. 383–404.
25. Babin M. A., Kuznetsov S. O. Approximating Concept Stability // Formal Concept Analysis SE - 7 / ed. by F. Domenach, D. Ignatov, J. Poelmans. –

Springer, Berlin, Heidelberg, 2012. – Vol. 7278 of *Lecture Notes in Computer Science*. – P. 7–15.

26. Ganter B., Grigoriev P. A., Kuznetsov S. O. et al. Concept-Based Data Mining with Scaled Labeled Graphs // Conceptual Structures at Work SE - 6 / ed. by K. Wolff, H. Pfeiffer, H. Delugach. – Springer, Berlin, Heidelberg, 2004 – Vol. 3127 of *Lecture Notes in Computer Science*. – P. 94–108.
27. Kuznetsov S. O., Samokhin M. V. Learning Closed Sets of Labeled Graphs for Chemical Applications // Inductive Logic Programming SE - 12 / ed. by S. Kramer, B. Pfahringer. – Springer, Berlin, Heidelberg, 2005. – Vol. 3625 of *Lecture Notes in Computer Science*. – P. 190–208.
28. Egho E., Jay N., Räıssi C. et al. A FCA-based analysis of sequential care trajectories // The 8th international conference on concept lattices and their applications. – 2011. – P. 363–376.

Материал поступил в редакцию 13.06.13.

Сведения об авторе

БУЗМАКОВ Алексей Владимирович - аспирант Высшей школы экономики на кафедре анализа данных и искусственного интеллекта факультета бизнес-информатики, Москва
e-mail: abuzmakov@gmail.com

УВАЖАЕМЫЕ ЧИТАТЕЛИ!

ЦЕНТР НАУЧНО-ИНФОРМАЦИОННОГО ОБСЛУЖИВАНИЯ ВИНИТИ РАН

ПРЕДОСТАВЛЯЕТ КОПИИ ПЕРВОИСТОЧНИКОВ

ВИНИТИ РАН осуществляет обслуживание копиями первоисточников, хранящихся в фонде научно-технической литературы ВИНИТИ, в фондах других библиотек, а также в доступных ВИНИТИ электронных ресурсах.

Фонд научно-технической литературы ВИНИТИ включает более 2 млн изданий по точным, естественным и техническим наукам, в т.ч.:

– отечественные и иностранные периодические и продолжающиеся издания – с 1987 г.;

– отечественные книги – с 1987 г.;

– иностранные книги – с 1991 г.;

– рукописи, депонированные в ВИНИТИ, – с 1962 г.

Заказы на бумажные или электронные копии первоисточников принимает Центр научно-информационного обслуживания (ЦНИО) ВИНИТИ. ЦНИО ВИНИТИ обслуживает коллективных (организации и учреждения) и индивидуальных пользователей.

Формы обслуживания:

– абонементная (на основе договоров и предоплаты);

– разовые заказы (с предоплатой заказа по счету);

– индивидуальная форма обслуживания в читальном зале ЦНИО ВИНИТИ.

На сайте ВИНИТИ (http://www.viniti.ru) представлен полный Электронный каталог научно-технической литературы (http://catalog.viniti.ru), зарегистрированной в ВИНИТИ с 1994 г. Доступ для просмотра и поиска по Каталогу свободный. Постоянные абоненты ЦНИО ВИНИТИ, имеющие логин и пароль для работы с Каталогом, могут делать заказ копий непосредственно через Каталог.

Услуги по изготовлению копий первоисточников из фондов других библиотек предоставляются только постоянным абонентам. Место хранения первоисточников указывается в Электронном каталоге.

За подробной информацией обращаться по адресу:

125190, Россия, Москва, ул. Усиевича, 20, ВИНИТИ РАН. ЦНИО

Телефоны: 8 (499)155-42-43, 155-42-09, 152-54-59

Факс: 8 (499) 943-00-60

E-mail: cnio@viniti.ru; **URL:** http://www.viniti.ru