УДК 004.825
DOI 10.25205/1818-7900-2019-17-3-29-42

Разработка методов интеграции автоматических средств логического вывода для порождения знаний в онтологической модели

А. И. Капустина [1], **Д. Е. Пальчунов** [1, 2]

[1] *Новосибирский государственный университет*
Новосибирск, Россия
[2] *Институт математики им. С. Л. Соболева СО РАН*
Новосибирск, Россия

Аннотация

Статья посвящена разработке методов порождения новых знаний на основе анализа текстов естественного языка. Для извлечения знаний из текстов на естественном языке используется метод представления предложений в виде двухместных предикатов с введенной константой ситуацией. Для представления знаний в формальном виде используются бесквантворные предложения логики предикатов, а также язык OWL DL. Порождение новых знаний реализуется при помощи автоматических средств логического вывода с использованием заранее заданных шаблонов правил вывода. Разработана программная система, которая дает возможность пользователям получать ответы на определенные вопросы, относящиеся к данным текстам естественного языка. Ответы строятся на естественном языке, при этом используются не только явно содержащиеся в обрабатываемом документе знания, но и знания, порожденные при помощи автоматических средств логического вывода.

Ключевые слова

извлечение знаний, порождение знаний, фрагменты атомарных диаграмм, OWL DL, SWRL, SQWRL, автоматические средства логического вывода, онтология, онтологическая модель

Благодарности

Исследование выполнено при частичной финансовой поддержке Президиума СО РАН (проект «Инженерия интенсиональных онтологий в дедуктивных и информационных системах» Комплексной программы ФНИ СО РАН II.1).

Для цитирования

Капустина А. И., *Пальчунов Д. Е.* Разработка методов интеграции автоматических средств логического вывода для порождения знаний в онтологической модели // Вестник НГУ. Серия: Информационные технологии. 2019. Т. 17, № 3. С. 29–42. DOI 10.25205/1818-7900-2019-17-3-29-42

Development of Methods for Integrating Automatic Logical Inference Tools to Generate Knowledge in the Ontological Model

A. I. Kapustina [1], **D. E. Palchunov** [1, 2]

[1] *Novosibirsk State University*
Novosibirsk, Russian Federation
[2] *Sobolev Institute of Mathematics SB RAS*
Novosibirsk, Russian Federation

Abstract

The article is devoted to the development of new knowledge generation methods based on the analysis of natural language texts. To extract knowledge from natural language texts, the method of presenting sentences in the form of binary predicates with new constant-situation is used. For the representation of knowledge in a formal form, we use quantifier-free sentences of predicate logic, as well as the OWL DL language. The generation of new knowledge is re-

alized with the help of reasoners, using pre-defined patterns of inference rules. A software system has been developed that allows users to get answers to specific questions related to given natural language texts. The answers are built in a natural language, using not only the knowledge that is explicitly contained in the document being processed, but also the knowledge generated by the reasoners.

Keywords

knowledge extraction, knowledge generation, fragments of atomic diagrams, OWL DL, SWRL, SQWRL, reasoners, ontology, ontological model

Acknowledgements

The study was carried out with partial financial support from the Presidium of the SB RAS (the project "Engineering of Intensional Ontologies in Deductive and Information Systems" of the Comprehensive Program of the Federal Research Institute of SB RAS II.1).

For citation

Kapustina A. I., Palchunov D. E. Development of Methods for Integrating Automatic Logical Inference Tools to Generate Knowledge in the Ontological Model. *Vestnik NSU. Series: Information Technologies*, 2019, vol. 17, no. 3, p. 29–42. (in Russ.) DOI 10.25205/1818-7900-2019-17-3-29-42

Введение

Данная работа посвящена разработке и программной реализации методов порождения новых знаний по текстам естественного языка. Порождение новых знаний происходит при помощи автоматических средств логического вывода с использованием шаблонов, предварительно созданных для заданной предметной области. Знания, извлеченные из текстов, представляются в виде фрагмента атомарной диаграммы модели в сигнатуре, состоящей из символов двухместных предикатов и констант. Далее атомарные предложения транслируются в язык OWL DL, благодаря чему мы можем использовать автоматические средства логического вывода (ризонеры). Корректность порожденных знаний контролируется пользователями с помощью запросов к программной системе на естественном языке.

В своей жизни мы сталкиваемся с большим количеством разных документов. Очень часто документы имеют большие объемы, и при этом информация в документах может быть не структурирована. Ключевые моменты могут находиться в разных местах документа: одна часть важной информации может быть в начале документа, а другая часть – в конце. Для точного определения содержания документа нужно выяснять смысл каждого предложения в отдельности, а также семантическое влияние одних предложений на другие.

Сходные проблемы возникают, когда для решения какой-либо задачи нужно извлечь знания из нескольких связанных между собой документов и затем сопоставить знания, содержащиеся в разных документах.

Так, например, для определения порядка действий студента ФИТ в течение ряда лет его обучения, нужно изучить следующие регламентирующие документы: «Образовательная программа по направлению 09.04.01 информатика и вычислительная техника», «09.04.01 Программа государственной итоговой аттестации по образовательной программе высшего образования – программе магистратуры», «Приложения к программе государственной итоговой аттестации по образовательной программе высшего образования – программе магистратуры», «Приказы о проведении сессий» и другие связанные документы. В одних документах описывается порядок действий студентов, в других описаны сроки и т. п. Поэтому, чтобы извлечь правильную информацию из таких текстов естественного языка, нужно изучить все связанные между собой документы. Учитывая количество и объемы таких документов, сделать это может быть крайне затруднительным.

Целью разработки программной системы, описываемой в данной статье, является автоматизация извлечения знаний из текстовых документов с возможностью порождения новых знаний на естественном языке за счет сопоставления и комбинации извлеченных знаний и осуществления логического вывода.

Программная система должна решать следующие задачи:

- извлекать знания из текстовых документов на естественном языке и представлять их в формальном виде (на языке логики предикатов и OWL DL);
- осуществлять логический вывод с использованием ризонеров для OWL DL и таким образом порождать новые знания;
- предоставлять пользователям возможность задавать вопросы на естественном языке, относящиеся к рассматриваемым текстам документов.

В результате применения разработанных методов была создана программная система, предназначенная для порождения новых знаний по текстам на естественном языке, основанная на представлении извлеченных из текстов знаний в виде онтологической модели. Созданная программная система позволяет получать знания, которые явно не содержатся в документах, но логически следуют из них.

Модульная система для реализации онтологической модели

При создании онтологических моделей различных предметных областей мы основываемся на четырехуровневой модели представления знаний [1; 2]. В рамках этого подхода разработка онтологической модели предметной области включает в себя:

- создание онтологии предметной области, т. е. задание набора ключевых понятий – сигнатуры онтологической модели и спецификация смысла этих ключевых понятий, задание взаимосвязей, отношений между понятиями и т. д.;
- формальное представление общих знаний (универсальных утверждений и регламентов, истинных для каждого прецедента данной предметной области);
- формальное представление набора прецедентов предметной области;
- формальное представление вероятностных, оценочных и экспертных знаний в виде предложений, имеющих нечеткие значения истинности [1; 2].

При этом происходит настройка взаимосвязи со сторонними программными системами, описание взаимодействия с другими онтологиями и др. [3–5]. В рамках нашего подхода создание онтологической модели можно представить как разработку модульной системы [6]. Модулями этой системы, в частности, являются: ядро, модуль пополнения онтологической модели, модуль взаимодействия с онтологической моделью.

Ядро модульной системы

Ядро системы позволяет создавать онтологии на языке OWL DL [1]. Создавать онтологию можно как посредством доступных редакторов онтологий с графическим интерфейсом, например Protégé, так и напрямую на языке OWL DL. Для ядра системы построения используется описанная выше четырехуровневая модель представления знаний [1; 2]: 1) онтология, 2) общие знания, 3) прецеденты, 4) вероятностные и оценочные знания.

Ядро системы, в частности, позволяет проверять созданные онтологические модели на непротиворечивость при помощи автоматических средств логического вывода.

Модуль пополнения онтологической модели

Модуль пополнения онтологической модели позволяет пополнять онтологическую модель новыми знаниями – как извлеченными из текстов естественного языка, так и полученными в результате логического вывода. Пополнение онтологической модели можно сделать несколькими способами.

Первый способ – это объединение онтологий. Для создания онтологических моделей существует несколько языков, основными из которых являются RDF, RDF Schema, OWL, логика описаний. Данный модуль позволяет объединять онтологии, написанные на разных языках. Для этого существуют программы для перевода с одного языка на другой.

[1] OWL 2 Web Ontology Language Document Overview (Second Edition) [Electronic resource]. W3C OWL Working Group. URL: http://www.w3.org/TR/owl2-overview

Второй способ – это использование правил вывода. Правила вывода могут быть написаны на разных языках. Для получения знаний с помощью правил вывода используются машины логического вывода.

Модуль взаимодействия с онтологической моделью

Модуль взаимодействия с онтологической моделью позволяет с помощью языков запросов SQWRL и SPARQL и правил вывода SWRL выполнять запросы к онтологической модели. Также есть возможность взаимодействовать со сторонними программными системами, такими как Ontograf (визуализация онтологии в виде графа), Logic Text (представление текста в виде атомарных диаграмм) и др.

Схема работы с онтологической моделью представлена на рис. 1.

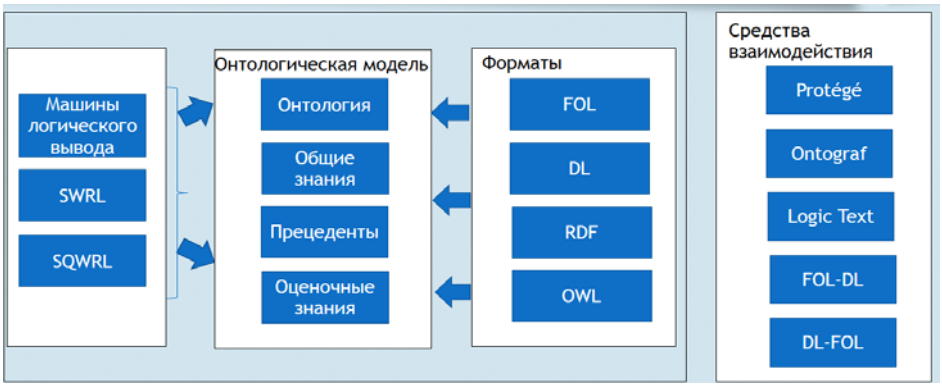

Рис. 1. Обобщенная схема работы с онтологической моделью

Fig. 1. General scheme of work with the ontological model

Автоматические системы логического вывода (ризонеры)

Семантический механизм рассуждений [7] – это часть программного обеспечения, способная вывести логические умозаключения (новые знания) из набора формализованных базовых понятий и представленных аксиом. Понятие семантического механизма рассуждений обобщает понятие машины логического вывода. Семантический механизм рассуждений предоставляет богатый набор механизмов для работы. Правила вывода обычно определяются с помощью языка описания онтологий. Многие семантические механизмы рассуждений используют логику предикатов первого порядка для выполнения рассуждений.

Для работы с онтологиями на языке OWL DL существует несколько основных машин логического вывода. Некоторые из них являются коммерческими программными продуктами: Bossam, RacerPro, OntoBroker [8]. Некоторые – свободными в использовании, но с закрытым исходным кодом: KAON2, Internet Business Logic, Cyc inference engine [9], а другие – бесплатным программным обеспечением с открытым исходным кодом: Drools [2], OpenRules [3], Pellet [10], Jena [4], Fact++ [11].

Для данной работы была выбрана машина логического вывода Pellet, так как она предоставляет наиболее полную поддержку OWL DL, а также имеет понятный программный интерфейс.

[2] Drools. Business Rules Management System. URL: http://www.drools.org
[3] OpenRules. User manual. URL: http://openrules.com/pdf/RulesSolver.UserManual.pdf
[4] Apache Jena. URL: https://jena.apache.org

Объединение правил вывода и онтологии

Применение машин логического вывода для онтологий можно осуществлять и без использования правил вывода. В таком случае машины логического вывода могут определять иерархию классов, принадлежность экземпляров к определенным классам, а также эквивалентность классов. Но новые знания в такой ситуации получить невозможно. Для этого нужно использовать правила вывода [12]. Для использования правил вывода при работе с онтологиями существуют различные инструменты: SWRL, DLP (Grosof), dl-programs (Eiter), DL-safe rules, Conceptual Logic Programs (CLP), AL-Log, DL+log [13].

Выделяют два основных подхода применения правил вывода в онтологиях: однородный (гомогенный) и гибридный [14].

Однородный подход

Однородный подход также называют семантической интеграцией. В однородном подходе онтологии и правила вывода используются на одинаковых условиях, т. е. создается единый язык. Одни и те же предикаты используются как для формулировки онтологических утверждений, так и для формулировки правил вывода. Правила в таком подходе можно использовать для определения классов и свойств. В данном подходе нет проблемы совместимости.

Недостатками данного подхода является то, что совмещение в одном языке разных средств (и для описания правил вывода, и для описания онтологии) сильно затрудняет реализацию. Также однородный подход часто может быть неприменим в работе из-за того, что онтология и правила вывода могут разрабатываться разными специалистами независимо друг от друга.

Примерами такого подхода являются SWRL [5] и DLP [15].

Гибридный подход

Гибридный подход отличается от гомогенного строгим семантическим разделением. Предикаты, используемые в онтологиях, и предикаты, используемые в правилах вывода, строго различаются. Вывод производится только с помощью отдельно реализуемых программ. Могут разрабатываться независимо от онтологий.

К недостаткам относится то, что правила не могут определять новых классов и свойств онтологий, за исключением некоторых специальных отношений. Хоть гибридный подход и позволяет разрабатывать правила вывода независимо от онтологий, это вносит ряд ограничений, для того чтобы гарантировать разрешимость.

Примерами такого подхода являются RIF [16] и Answer Set Programming (ASP) [17].

Системы для извлечения знания из текстов

В настоящее время существует ряд программных систем, предназначенных для анализа текстов естественного языка при помощи онтологий. Такими системами являются FRED [18], Apache Stanbol [6] и Text2Onto [19]. Рассмотрим особенности этих систем.

FRED – это анализатор текста, основанный на технологиях Semantic Web. Способен обрабатывать тексты естественного языка на 48 различных языках и преобразовывать его в связанные данные. Он разработан на языке Python, а также реализован в виде REST API и в виде набора библиотек языка Python. Для анализа текста FRED использует комбинаторную категориальную грамматику (C&C), теорию репрезентационного представления (DRT), семантику фреймов и шаблоны проектирования онтологий. Для определения именованного объекта анализатор FRED использует Apache Stanbol, TagMe, для устранения неоднозначности – Boxer и IMS. Результат обработки может представлять в формате RDF/OWL. Также FRED

[5] SWRL API: SWRL, SWRL API, SQWRL, SQWRL API. URL: https://github.com/protegeproject/swrlapi/wiki
[6] Apache Stanbol. Documentation Apache Stanbol. URL: https://stanbol.apache.org

генерирует графы знаний, которые могут быть интерпретированы машинами в соответствии с общей формальной семантикой. В основе работы лежат фреймы (Frame Semantic). Фрейм обычно выражается глаголами или другими лингвистическими конструкциями, которые могут распознаваться во входном тексте как *n*-арное отношение. Все *n*-арные отношения записываются как подклассы класса *dul:Event.* Предикаты представляются в виде классов, в названии которых есть сам объект. Пример такого именования: *fred:Love.* Вывод может быть представлен в виде графа (рис. 2).

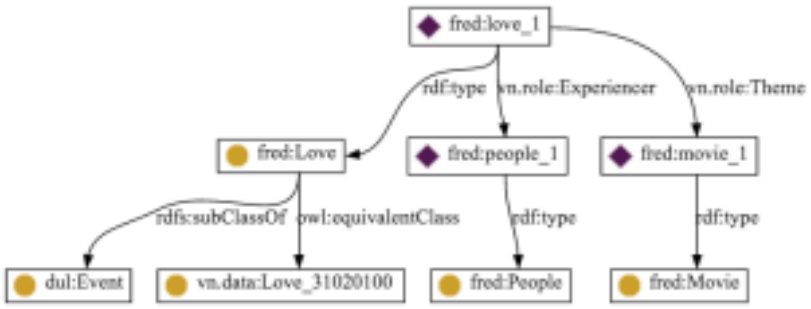

Рис. 2. Граф отношений
Fig. 2. Relationship graph

Text2Onto – программный продукт для пополнения онтологии из текстовых ресурсов. В основе его лежит вероятностная онтологическая модель (POM – Probabilistic Ontology Model). POM, используемая Text2Onto, представляет собой набор созданных примитивов моделирования, которые не зависят от конкретного языка представления онтологии. Фактически Text2Onto включает в себя библиотеку примитивов моделирования (MPL), которая определяет эти примитивы декларативным способом. Добавление новых примитивов происходит без изменения структуры онтологии. Это делает онтологию гибкой и расширяемой. Также созданные примитивы могут быть переведены на любой язык представления знаний, например RDFS3, OWL4 и F-Logic.

Text2Onto сочетает в себе машинное обучение и базовую лингвистическую обработку – токенизацию и разбор на предложения. Программа определяет в тексте некоторые виды отношений, такие как класс-подкласс, эквивалентность, экземпляры классов. Например, для выявления отношения подкласса в Text2Onto были реализованы различные алгоритмы, в частности реализовали алгоритм сопоставления шаблонов с использованием структуры WordNet.

Разработка программной системы

Краткое описание

Целью разрабатываемой программной системы является получение новых знаний на основе уже имеющихся знаний, содержащихся в текстах естественного языка, а также получения ответов на вопросы, относящиеся к этим текстам. В рамках работы в качестве примеров были взяты локальные нормативные документы факультета информационных технологий.

На вход программная система получает текстовый документ, который она преобразовывает в бесквантерные предложения логики предикатов. Для преобразования текстов в логику предикатов используется программа Logic Text [20]. С помощью данной программы можно получить фрагменты атомарных диаграмм по предложениям естественного языка.

Далее происходит преобразование многоместных предикатов в двухместные. Для того чтобы не потерять смысл предложений, вводятся константы, обозначающие ситуации. После этого полученные предложения, записанные в сигнатуре двухместных предикатов, транслируется в язык OWL.

На следующем шаге порождаются правила вывода по заранее созданным шаблонам. Шаблоны строятся для каждой предметной области отдельно. Для построения шаблонов используется логика предикатов рассматриваемой сигнатуры. С помощью машины логического вывода и созданных правил вывода в онтологическую модель добавляются новые знания. Для работы с онтологией на языке OWL и с правилами вывода, написанными на SWRL, используется машина логического вывода Pellet.

Шаблоны для запросов к онтологической модели также строятся на языке логики предикатов. Как и шаблоны для правил вывода, шаблоны для запросов строятся под конкретную предметную область. Заранее подготовленные шаблоны для запросов к онтологической модели заполняются с помощью пользовательского интерфейса, после чего происходит запрос к онтологической модели.

Учитывая новые выведенные знания, мы получаем ответ на запрос с помощью языка SQWRL, затем формулируем ответ на понятном для пользователя языке. В итоге программная система может отвечать на вопросы пользователей по текстам на естественном языке.

Подробное описание

Схема работы программной системы представлена на рис. 3.

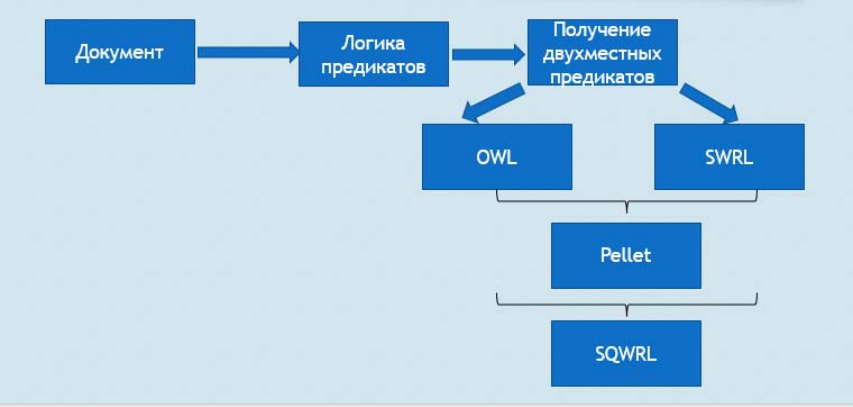

Рис. 3. Схема работы программной системы
Fig. 3. Relationship graph

В схеме работы можно выделить следующие основные этапы:

- преобразование текстовых документов выбранной предметной области в наборы атомарных предложений;
- конвертация фрагментов атомарных диаграмм в онтологическую модель;
- порождение правил вывода для построенной онтологической модели;
- вывод новых знаний;
- проверка полученных результатов.

Рассмотрим подробно каждый из этапов.

Преобразование текстовых документов выбранной предметной области в наборы атомарных предложений

Для построения по текстам естественного языка фрагмента атомарной диаграммы используется программная система LogicText [21]. Знания, представленные в тексте, формализуют-

ся на языке многоместных предикатов. Однако многоместные предикаты нельзя представить на языке OWL, так как OWL поддерживает только двухместные предикаты. Поэтому нужно преобразовать многоместные предикаты в двухместные.

Конвертация фрагментов атомарных диаграмм в онтологическую модель. Онтологическая модель строится на основе четырехуровневой модели представления знаний [1; 2]: это онтология, общие знания, прецеденты, оценочные знания. В данном случае нам потребуются только первые три уровня.

Рассмотрим уровни онтологической модели на примере нормативных документов факультета.

Первый уровень – это онтология [22]. Онтология представляет собой формальное описание языка предметной области, цель которого – в явном виде определить смысл терминов, используемых в данной предметной области. Ключевые термины – это понятия, смысл которых, во-первых, является специфичным для данной предметной области и, во-вторых, одинаково понимается всеми экспертами предметной области.

Примерами таких знаний о смысле понятий являются: «*ВКР – это выпускная квалификационная работа. Студент является обучающимся. ГИА – это государственная итоговая аттестация*».

Второй уровень – это общие, универсальные знания о предметной области, которые считаются полностью истинными (на данный момент времени). В рамках формального описания учебного процесса это знания, извлеченные из различных нормативных документов. Общие знания извлекаются из текстов и содержат общую информацию о предметной области. Например: «*Занятия в учебном году начинаются 1-го сентября. Предзащита ВКР проходит в срок с 15 по 31 мая четвертого семестра обучения*».

Третий уровень онтологической модели – прецеденты (или частные знания). Это знания, которые истины в конкретной ситуации, т. е. для определенного экземпляра предметной области. Например: «*В 2019 году государственные аттестационные испытания по направлению подготовки 09.03.01 Информатика и вычислительная техника (уровень бакалавриата) проводятся с 24.06.2019 по 26.06.2019*».

Для пополнения онтологической модели знаниями, извлеченными из текстов естественного языка, происходит преобразование многоместных предикатов в двухместные. Для этого используется алгоритм, представленный в [23]. Преобразование многоместных предикатов в двухместные происходит с помощью введения сигнатуры дополнительных констант-ситуаций.

Порождение правил вывода. Для пополнения онтологической модели новыми знаниями необходимо создавать правила вывода. Чтобы не строить каждый раз правила вывода вручную, используются шаблоны правил вывода. Заполнение шаблонов правил вывода происходит в момент создания онтологической модели.

Шаблоны строятся для конкретной предметной области. При переводе n-местных предикатов в двухместные вводятся специальные отношения между константами-ситуациями [23]. Например:

- $Include(s_1, s_2)$ – отношение, обозначающее, что ситуация s_1 является частью более общей ситуации s_2;
- $Id(s_1, s_2)$ – отношение, обозначающее, что ситуации s_1 и s_2 эквивалентны.

Для отношения Id справедливы, например, следующие правила вывода:

$$\big(Id(s_1, s_2) \rightarrow Id(s_2, s_1)\big)$$

$$\Big(\big(Id(s_1, s_2)\ \&\ P(s_1, x)\big) \rightarrow P(s_2, x)\Big)$$

А для отношения $Include$ справедливы правила вывода:

$$\Big(\big(Include(s_1, s_2)\ \&\ Include(s_2, s_1)\big) \rightarrow Id(s_1, s_2)\Big)$$

$$\Big(\big(Include(s_1, s_2)\ \&\ Include(s_2, s_3)\big) \rightarrow Include(s_1, s_3)\Big)$$

Эти правила вывода являются примерами шаблонов. Вместо P мы можем подставить любой предикат из сигнатуры онтологической модели. s_1 и s_2 тоже могут быть любыми ситуациями, которые связанны предикатами $Include$ или Id соответственно. Шаблоны правил вывода могут заполняться автоматически и добавляться в онтологическую модель уже как готовые правила вывода.

Порождение новых знаний. Для получения по онтологической модели новых знаний используются правила вывода, построенные на предыдущем этапе, и машина логического вывода Pellet. Сначала машина логического вывода проверяет, что созданная онтологическая модель не является противоречивой. Если мы выявили, что какие-либо знания противоречат друг другу, то специальная утилита для редактора онтологий Protégé подскажет, какие утверждения вступают в противоречие. Разрешить такие противоречия в рамках разрабатываемой системы можно только вручную.

После того как машина логического вывода подтвердила, что онтологическая модель является непротиворечивой, мы переходим к порождению новых знаний. Система Pellet использует порожденные ранее правила вывода и добавляет новые знания в онтологическую модель.

Проверка полученных результатов. Чтобы проверить, получилось ли вывести новые знания, используются запросы к онтологической модели, которые также строятся по шаблонам. Шаблоны запросов порождаются для конкретной предметной области аналогично шаблонам правил вывода.

Редактирование шаблона происходит пользователем через графический интерфейс. Шаблон имеет вид какого-либо вопроса. Например: «*Что должен сделать X?»* Вместо X может быть подставлен любой класс из онтологической модели, который связан с ситуацией свойством *кто* или *что*. При этом можно выбрать любой вопрос, который заполнился по шаблону. Если заполнение шаблона не может произойти однозначно, то пользователю предоставляется возможность самому выбрать правильный вопрос. Например, если вместо X можно подставить несколько слов: *студент*, *рецензент*, *научный руководитель* и т. д., то на экране появится выпадающий список со всеми этими словами. Пользователь сам выберет нужное и задаст вопрос. Также пользователь может задать несколько вопросов, по очереди выбирая нужное слово.

После того как пользователь сформировал вопрос, он представляется в виде SQWRL запроса к онтологической модели. Так как SQWRL имеет доступ и к тем знаниям, которые были выведены с помощью машины логического вывода, то в ответ пользователю попадет не только та информация, которая в явном виде содержится в текстах, но и та, которая была выведена с помощью правил вывода.

SQWRL возвращает ответ в виде таблицы. По данным из таблицы, учитывая ситуацию, с которой связано это слово, мы строим текстовый ответ пользователю и отображаем его на экране.

Чтобы проверить, какие знания получилось вывести, можно, например, задать следующие вопросы:

- *Что должен сделать студент?*
- *Когда научный руководитель должен предоставить отзыв на ВКР?*
- *Что должен сделать рецензент?*
- *и другие*

Рассмотрим ответ на вопрос: «*Что должен сделать рецензент?*». По этому вопросу составится SQWRL запрос вида

$$кто(?x, рецензент) \& \text{ что делать } (?x, ?y) \rightarrow sqwrl{:}\, select(?y)$$

Это основной запрос, по нему получим информацию о том, что должен сделать рецензент. В данном случае это будет: *проводить* и *предоставить*. Чтобы ответ получился полным, нужно сделать запросы, уже используя полученную информацию. В итоге получим ответ: *Рецензент должен проводить анализ и предоставлять письменные рецензии.* Далее пользователю предлагаются уточняющие вопросы. Например, *когда рецензент должен предоста-*

вить письменные рецензии? В ответ получим: *с 15 мая 4 семестра обучения до 31 мая 4 семестра обучения.*

Разработка шаблонов правил вывода и запросов к онтологической модели

Была разработана программная система на языке Java, работающая посредством OWL API с онтологической моделью, представленной на языке OWL 2 DL.

Разработка шаблонов правил вывода

Шаблоны правил вывода создаются индивидуально для каждой предметной области. Выше были рассмотрены примеры шаблонов, в частности:

$$\Big(\big(Id(s_1, s_2)\ \&\ P(s_1, x)\big)\ \rightarrow P(s_2, x)\Big)$$

Помимо шаблонов правил вывода со специальными отношениями, мы рассматриваем шаблоны, которые не связаны со специальными отношениями:

$$\textit{с какой даты}(?\,x, ?\,y)\&\ \textit{до какой даты}\ (?\,x, ?\,z)\&$$
$$swrlb\colon subtractDayTimeDurations(?\,d, ?\,y, ?\,z) \rightarrow \textit{иметь период}\ (?\,x, ?\,d)$$

Подставляя в шаблоны конкретные константы-ситуации и конкретные предикаты, программа порождает правила вывода. Это происходит автоматически через OWL API.

Разработка шаблонов запросов SQWRL

Запросы к онтологической модели строятся с помощью SQWRL. Для данной предметной области можно определить основные шаблоны запросов:

- *Что должен сделать X?*
- *Когда X должен сделать Y Z?*
- *Сколько времени есть у X для выполнения задачи Y?*
- *Сколько задач нужно сделать X в период Y?*

Пользователю выводятся данные шаблоны, вместо *X* или *Y* предлагаются на выбор слова. Слова берутся из онтологической модели. Например, для первого вопроса «*Что должен сделать X?*» для того, чтобы заполнить возможными значениями переменную *X*, в онтологическую модель делается запрос:

$$\textit{кто}(?\,x, ?\,y) \rightarrow sqwrl\colon select(?\,y)$$

Переменная *x* не содержится в $sqwrl\colon select$ потому, что нам не важно в данном случае, к какой ситуации относятся выведенные константы. Аналогичные запросы выполняются для всех переменных.

После того как пользователь сделает выбор по поводу заполнения переменных, создаются запросы к онтологии.

Ответы пользователям на естественном языке также строятся по шаблонам. Шаблоны ответов строятся аналогично шаблонам вопросов. Так, например, к шаблону вопроса *Что должен сделать X? с*троится шаблон ответа *X должен Y*. Здесь *Y* – это ответ, полученный с помощью SQWRL-запроса.

В общем виде из онтологической модели с помощью SQWRL-запроса может быть получено несколько действий *Y* объекта *X*. Обобщая шаблон на такой случай, получаем:

X должен Y и/или Z, и/или S, и/или … .

Рассмотрим примеры вопросов и их представление на SQWRL. Ответы представим сразу в читаемом для пользователя виде – на естественном языке.

- Когда *студент* должен *предъявить первый вариант ВКР?*

$$\textit{кто}(?\,x, \textit{студент})\ \&\ \textit{что делать}(?\,x, \textit{предъявлять})$$
$$\&\ \textit{что}(?\,x, \textit{первый вариант ВКР})$$
$$\&\ \textit{когда}(?\,x, ?\,a) {-}{>}\ sqwrl\colon select(?\,a)$$

На такой запрос мы не сможем получить ответ, так как промежутка времени не задано в предложении. Поэтому нужно сделать два других запроса:

$$кто(?\,x, студент)\ \&\ что\ делать(?\,x, предъявлять)$$
$$\&\ что(?\,x, первый\ вариант\ ВКР)$$
$$\&\ \text{до\_какой\_даты}(?\,x, ?\,a) -> \ sqwrl{:}\,select(?\,a)$$

или

$$кто(?\,x, студент)\ \&\ что\ делать(?\,x, предъявлять)$$
$$\&\ что(?\,x, первый\ вариант\ ВКР)$$
$$\&\ \text{с\_какой\_даты}(?\,x, ?\,a) -> \ sqwrl{:}\,select(?\,a)$$

Получим ответ по шаблону: X должен Y до Z.

Студент должен предъявить первый вариант ВКР до 15 апреля второго семестра обучения.

- Сколько времени есть у *научного руководителя* для выполнения задачи *отзыв на ВКР?*

$$кто(?\,x, научный\ руководитель)\ \&\ что\ (?\,x, отзыв\ на\ ВКР)$$
$$\&\ иметь\ период\ (?\,x, ?\,a) -> \ sqwrl{:}\,select(?\,a)$$

В запросах, начинающихся на «*сколько*», в ответ пользователю выдаем просто значение, полученное с помощью SQWRL.

Получим ответ*:16 дней.*

Апробация программной системы

Тестирование программной системы проходило на документах ФИТ НГУ, а именно: «09.04.01 Программа государственной итоговой аттестации по образовательной программе высшего образования – программе магистратуры», «Приказы о проведении сессий». Было обработано 30 предложений, содержащих информацию о сроках проведения этапов обучения, представления документов на кафедру и порядке проведения защиты квалификационной работы. По каждому предложению было составлено порядка 10 двухместных предикатов. По подготовленным предикатам составлена онтологическая модель, содержащая 367 аксиом. Автоматически были созданы 50 правил вывода. Программный продукт может отвечать на 6 видов вопросов, а именно: *Кто?, Что?, Что сделать?, Когда?, Какой промежуток?, Сколько видов работ?*

Заключение

Статья посвящена разработке методов извлечения знаний из текстов на естественном языке и порождения новых знаний, явно не содержащихся в текстах. Была создана программная система, предназначенная для порождения новых знаний на основе обработки текстов естественного языка. Разработанная программная система позволяет извлекать знания из текстов и представлять их в формальном виде в онтологической модели. Онтологическая модель строится на основе четырехуровневой модели представления знаний. Использование онтологической модели позволяет порождать новые знания с помощью добавления новых правил вывода, а также проверять полученные знания на непротиворечивость с помощью машины логического вывода.

Программная система дает возможность пользователям задавать вопросы на естественном языке, относящиеся к рассматриваемым документам. Ответы на вопросы строятся динамически, исходя из знаний, содержащихся в онтологической модели. При составлении ответа на вопрос используются как знания, явно содержащиеся в текстах, так и новые знания, порожденные в онтологической модели.

Список литературы

1. **Найданов Ч. А., Пальчунов Д. Е., Сазонова П. А.** Теоретико-модельные методы интеграции знаний, извлеченных из медицинских документов // Вестник НГУ. Серия: Информационные технологии. 2015. Т. 13, № 3. С. 29–41.
2. **Naydanov Ch., Palchunov D., Sazonova P.** Development of automated methods for the prevention of risks of critical conditions, based on the analysis of the knowledge extracted from the medical histories. *Siberian Scientific Medical Journal*, 2016, vol. 36, iss. 1, p. 105–113.

3. **Маслов В. А., Соколов С. М.** Обработка семантических запросов в среде Protégé на примере построения онтологии дорожных знаков // Препринты ИПМ им. М. В. Келдыша. 2018. № 260. 15 с. DOI 10.20948/prepr-2018-260
4. **Асламова В. С., Берестнева О. Г., Темникова Е. А.** Онтологическое моделирование предметной области учреждения дополнительного образования // Онтология проектирования. 2015. Т. 5, № 4 (18).
5. **Ефименко И. В., Хорошевский В. Ф.** Онтологическое моделирование экономики предприятий и отраслей современной России. Ч. 3: Российские исследования и разработки в области онтологического инжиниринга и бизнес-онтологии. М.: ИД ВШЭ, 2011. 68 с.
6. **Найданов Ч. А.** Разработка ядра онтологической модели, настраиваемой под предметную область // Вестник НГУ. Серия: Информационные технологии. 2019. Т. 17, № 1. С. 72–81. DOI 10.25205/1818-7900-2019-17-1-72-81
7. **Horrocks I.** Description logic reasoning. In: International Conference on Logic Programming and Automated Reasoning. Montevideo, Uruguay, 2000.
8. **Гаврилова Т. А., Хорошевский В. Ф.** Базы знаний интеллектуальных систем. СПб: Питер, 2000. 384 с.
9. **Baxter D., Shepard D., Siegel N., Gottesman B., Schneider D.** Interactive Natural Language Explanations of Cyc Inferences. In: AAAI 2005 International Symposium on Explanation-aware Computing, 2005. URL: https://www.aaai.org/Papers/Symposia/Fall/2005/FS-05-04/FS05-04-002.pdf
10. **Zuo M., Haarslev V.** Intelligent Tableau Algorithm for DL Reasoning. In: Proceedings of the 22nd International Conference on Automated Reasoning with Analytic Tableaux and Related Methods. France, Nancy, 2013, p. 273–287. URL: https://link.springer.com/chapter/10.1007/978-3-642-40537-2_23
11. **Tsarkov D., Horrocks I.** FaCT++ description logic reasoner: System description. In: Third International Joint Conference on Automated Reasoning, 2006, p. 292–297. URL: http://dl.acm.org/citation.cfm?id=2136140
12. **Курейчик В. В., Бова В. В.** Моделирование процесса представления знаний в интеллектуальных обучающих системах на основе компетентностного подхода // Открытое образование. 2014. № 3. URL: https://cyberleninka.ru/article/n/modelirovanie-protsessa-predstavleniya-znaniy-v-intellektualnyh-obuchayuschih-sistemah-na-osnove-kompetentnostnogo-podhoda
13. **Heymans S., Nieuwenborgh D. V., Vermeir D.** Conceptual logic programs. *Annals of Mathematics and Artificial Intelligence*, 2006, vol. 47 (1–2), p.103–137.
14. **Авдошин С. М., Шатилов М. П.** Онтологический инжиниринг // Бизнес-информатика. 2007. № 2. С. 32–36.
15. **Papadimitriou P., Garcia-Molina H.** Data Leakage Detection. URL: ilpubs.stanford.edu:8090/839/1/2008-23.pdf
16. **Калиниченко Л. А., Ступников С. А.** Анализ мотивации, целей и подходов проекта унификации языков на правилах. URL: http://synthesis.ipi.ac.ru/synthesis/publications/11ont-uni/11ont-uni.pdf
17. **Lifschitz V.** What is answer set programming? In: Proceedings of AAAI, 2008.
18. **Gangemi A., Presutti V., Reforgiato Recupero D., Giovanni Nuzzolese A., Draicchio F., Mongiovì M.** Semantic Web Machine Reading with FRED. *Semantic Web Journal*, 2017, vol. 8 (6), p. 873–893.
19. **Cimiano P., Völker J.** Text2onto. *Nat. Lang. Process Inform. Syst*., 2005, vol. 3513, p. 227–238. DOI 10.1007/11428817_21
20. **Махасоева О. Г., Пальчунов Д. Е.** Автоматизированные методы построения атомарной диаграммы модели по тексту естественного языка // Вестник НГУ. Серия: Информационные технологии. 2014. Т. 12, № 2. С. 64–73.

21. **Махасоева О. Г., Пальчунов Д. Е.** Программная система построения атомарной диаграммы модели по тексту естественного языка. Св-во о гос. регистрации программы для ЭВМ № 2014619198, зарегистрировано 10.09.2014.
22. **Gruber T. R.** Towards Principles for the Design of Ontologies Used for Knowledge Sharing. In: International Workshop on Formal Ontology. Padova, Italy, 1993.
23. **Ненашева Е. О., Пальчунов Д. Е.** Разработка автоматизированных методов преобразования предложений естественного языка в бескванторные формулы логики предикатов // Вестник НГУ. Серия: Информационные технологии. 2017. Т. 15, № 3. С. 49–63. DOI 10.25205/1818-7900-2017-15-3-49-63

References

1. **Naidanov Ch. A., Palchunov D. E., Sazonova P. A.** Model-theoretic methods of integration of knowledge extracted from medical documents. *Vestnik NSU. Series: Information Technologies*, 2015, vol. 13, no. 3, p. 29–41. (in Russ.)
2. **Naydanov Ch., Palchunov D., Sazonova P.** Development of automated methods for the prevention of risks of critical conditions, based on the analysis of the knowledge extracted from the medical histories. *Siberian Scientific Medical Journal*, 2016, vol. 36, iss. 1, p. 105–113.
3. **Maslov V. A., Sokolov S. M.** Processing of semantic queries in the Protégé environment using the example of constructing an ontology of road signs. In: Preprint IPM im. M. V. Keldysh, 2018, no. 260, 15 p. DOI 10.20948 / prepr-2018-260 (in Russ.)
4. **Aslamova V. S., Berestneva O. G., Temnikova E. A.** Ontological modeling of the subject area of the institution of additional education. *Design Ontology*, 2015, vol. 5, no. 4 (18). (in Russ.)
5. **Efimenko I. V., Khoroshevsky V. F.** Ontological modeling of the economics of enterprises and branches of modern Russia. Part 3. Russian research and development in the field of ontological engineering and business ontology. Moscow, Higher School of Economics Press, 2011, 68 p. (in Russ.)
6. **Naydanov Ch. A.** Development of an Ontological Model Core Customizable for a Specific Subject Domain. *Vestnik NSU. Series: Information Technologies*, 2019, vol. 17, no. 1, p. 72–81. (in Russ.) DOI 10.25205/1818-7900-2019-17-1-72-81
7. **Horrocks I.** Description logic reasoning. In: International Conference on Logic Programming and Automated Reasoning. Montevideo, Uruguay, 2000.
8. **Gavrilova T. A., Khoroshevsky V. F.** Knowledge Base of Intellectual Systems. St. Petersburg, Peter, 2000, 384 p.
9. **Baxter D., Shepard D., Siegel N., Gottesman B., Schneider D.** Interactive Natural Language Explanations of Cyc Inferences. In: AAAI 2005 International Symposium on Explanation-aware Computing, 2005. URL: https://www.aaai.org/Papers/Symposia/Fall/2005/FS-05-04/FS05-04-002.pdf
10. **Zuo M., Haarslev V.** Intelligent Tableau Algorithm for DL Reasoning. In: Proceedings of the 22nd International Conference on Automated Reasoning with Analytic Tableaux and Related Methods. France, Nancy, 2013, p. 273–287. URL: https://link.springer.com/chapter/10.1007/978-3-642-40537-2_23
11. **Tsarkov D., Horrocks I.** FaCT++ description logic reasoner: System description. In: Third International Joint Conference on Automated Reasoning, 2006, p. 292–297. URL: http://dl.acm.org/citation.cfm?id=2136140
12. **Kureichik V. V., Bova V. V.** Modeling the process of knowledge representation in intelligent tutoring systems based on the competence approach. *Open Education*, 2014, no. 3. URL: https://cyberleninka.ru/article/n/modelirovanie-protsessa-predstavleniya-znaniy-v-intellektualnyh-obuchayuschih-sistemah-na-osnove-kompetentnostnogo-podhoda (in Russ.)
13. **Heymans S., Nieuwenborgh D. V., Vermeir D.** Conceptual logic programs. *Annals of Mathematics and Artificial Intelligence*, 2006, vol. 47 (1–2), p.103–137.

14. **Avdoshin S. M., Shatilov M. P.** Ontological engineering. Business Informatics, 2007, no. 2, p. 32–36. (in Russ.)
15. **Papadimitriou P., Garcia-Molina H.** Data Leakage Detection. URL: ilpubs.stanford.edu: 8090/839/1/2008-23.pdf
16. **Kalinichenko L. A., Stupnikov S. A.** Analysis of the motivation, goals and approaches of the project of the unification of languages on the rules. URL: http://synthesis.ipi.ac.ru/synthesis/publications/11ont-uni/11ont-uni.pdf
17. **Lifschitz V.** What is answer set programming? In: Proceedings of AAAI, 2008.
18. **Gangemi A., Presutti V., Reforgiato Recupero D., Giovanni Nuzzolese A., Draicchio F., Mongiovì M.** Semantic Web Machine Reading with FRED. *Semantic Web Journal*, 2017, vol. 8 (6), p. 873–893.
19. **Cimiano P., Völker J.** Text2onto. *Nat. Lang. Process Inform. Syst*., 2005, vol. 3513, p. 227–238. DOI 10.1007/11428817_21
20. **Makhasoeva O. G., Palchunov D. E.** Semi-automatic methods of a construction of the atomic diagrams from natural language texts. *Vestnik NSU. Series: Information Technologies*, 2014, vol. 12, no. 2, p. 64–73. (in Russ.)
21. **Makhasoeva O. G., Palchunov D. E.** Software system for constructing an atomic diagram of a model from the text of a natural language. Certificate of state registration of computer programs № 2014619198, registered 10.09.2014.
22. **Gruber T. R.** Towards Principles for the Design of Ontologies Used for Knowledge Sharing. In: International Workshop on Formal Ontology. Padova, Italy, 1993.
23. **Nenasheva E. O., Palchunov D. E.** Semi-Automated Methods of Transforming Sentences from Natural Language into Quantifier-Free Formulas of Predicate Logic. *Vestnik NSU. Series: Information Technologies*, 2017, vol. 15, no. 3, p. 49–63. (in Russ.)

Материал поступил в редколлегию
Received
25.06.2019

Сведения об авторах / Information about the Authors

Капустина Алина Игоревна, студент, 2 курс магистратуры, факультет информационных технологий, Новосибирский государственный университет (ул. Пирогова, 1, Новосибирск, 630090, Россия)

Alina I. Kapustina, Student, 2 year master course, Department of Information Technology, Novosibirsk State University (1 Pirogov Str., Novosibirsk, 630090, Russian Federation)

a.kapustina@g.nsu.ru

Пальчунов Дмитрий Евгеньевич, доктор физико-математических наук, ведущий научный сотрудник, Институт математики СО РАН (пр. Академика Коптюга, 4, Новосибирск, 630090, Россия), заведующий кафедрой ОИ ФИТ, Новосибирский государственный университет (ул. Пирогова, 1, Новосибирск, 630090, Россия)

Dmitry E. Palchunov, Doctor of Physical and Mathematical Sciences, Leading Researcher, Sobolev Institute of Mathematics SB RAS (4 Academician Koptyug Ave., Novosibirsk, 630090, Russian Federation); Head of the Department of General Informatics of the Faculty of Information Technologies, Novosibirsk State University (1 Pirogov Str., Novosibirsk, 630090, Russian Federation)

palch@math.nsc.ru