Разработка методов семантического поиска в Интернете, основанных на древовидных лингвистических шаблонах

И. И. Филиппов¹, Д. Е. Пальчунов^{1,2}, П. А. Степанов¹

¹ *Новосибирский государственный университет
Новосибирск, Россия*

² *Институт математики им. С. Л. Соболева СО РАН
Новосибирск, Россия*

Аннотация

Статья посвящена разработке методов поиска информации в Интернете. В настоящее время задача быстрого автоматизированного извлечения данных из различных источников является чрезвычайно актуальной. Цель данной работы – создание инструментария для решения этой задачи. Идея подхода заключается в разработке полуавтоматических методов составления лингвистических шаблонов для заданного фрагмента атомарной диаграммы. Процесс построения шаблонов по данному фрагменту атомарной диаграммы выполняется в 2 этапа: 1) порождение простых шаблонов; 2) с помощью полученных простых шаблонов создание более сложных или реже встречающихся шаблонов. С помощью полученных шаблонов можно находить необходимую информацию, обрабатывая тексты естественного языка, представленные в Интернете.

Ключевые слова

лингвистические шаблоны, синтаксические деревья, фрагменты атомарных диаграмм, семантический поиск информации, извлечение знаний

Благодарности

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

Для цитирования

Филиппов И. И., Пальчунов Д. Е., Степанов П. А. Разработка методов семантического поиска в Интернете, основанных на древовидных лингвистических шаблонах // Вестник НГУ. Серия: Информационные технологии. 2019. Т. 17, № 3. С. 111–122. DOI 10.25205/1818-7900-2019-17-3-111-122

Development of Methods of Semantic Search on the Internet, Based on Treelike Linguistic Templates

I. I. Filippov¹, D. E. Palchunov^{1,2}, P. A. Stepanov¹

¹ *Novosibirsk State University
Novosibirsk, Russian Federation*

² *Sobolev Institute of Mathematics SB RAS
Novosibirsk, Russian Federation*

Abstract

The article is devoted to development of methods of search of information on the Internet. Currently the issue of fast and automatic data extraction from different sources is very actual. The aim of this work is creation of instrument to solve this problem. The idea of the approach is to develop semi-automatic methods of linguistic templates construction for given fragment of atomic diagram. Algorithm of templates construction for the fragment of atomic diagram is performed in 2 stages. On the 1st stage generation of the simple templates is done, on the 2nd stage more seldom and

complicated templates are found, using templates from 1st part of algorithm. Processing texts in natural language on the Internet using gotten templates, needed information may be found.

Keywords

linguistic templates, syntax trees, fragments of atomic diagrams, semantic search, knowledge extraction

For citation

Filippov I. I., Palchunov D. E., Stepanov P. A. Development of Methods of Semantic Search on the Internet, Based on Treelike Linguistic Templates. *Vestnik NSU. Series: Information Technologies*, 2019, vol. 17, no. 3, p. 111–122. (in Russ.) DOI 10.25205/1818-7900-2019-17-3-111-122

Введение

В современном мире задача быстрого и автоматизированного извлечения данных является очень актуальной. Например, когда возникает необходимость найти некоторую информацию в Интернете, нужно просмотреть множество страниц, попробовать несколько вариантов запросов, проанализировать большое количество информации и выбрать что-то действительно удовлетворяющее требованиям. Для некоторых запросов подобный процесс может потребовать очень много времени и усилий. Поэтому было бы крайне полезным наличие программного инструмента, который мог бы предложить пользователю не огромное количество различных Интернет-ресурсов, а несколько, например 3–5, в которых содержалась бы нужная ему информация. Разработка такой программной системы является целью данной работы.

Предлагаемый подход основан на разработке полуавтоматических методов порождения лингвистических шаблонов для поискового запроса, формально представленного в виде некоторого фрагмента атомарной диаграммы алгебраической системы. Построение лингвистического шаблона для конкретного фрагмента атомарной диаграммы выполняется в два этапа. На первом этапе происходит поиск простых шаблонов. На втором этапе на основе построенных простых шаблонов выполняется поиск более сложных вариантов шаблонов или редко встречающихся в текстах. С помощью полученных шаблонов происходит поиск и извлечение необходимой информации путем проверки соответствия текста естественного языка лингвистическому шаблону.

В настоящее время существует множество систем, решающих подобные задачи [1–4]. Самые популярные среди них – поисковые системы, такие как Гугл, Яндекс и др. Они основаны на поиске по ключевым словам с применением различных эвристик и способны выдать нужную информацию для простых запросов, однако возникают проблемы с более сложными поисковыми запросами.

Также задачу поиска информации призваны решать различные вопросно-ответные системы [5; 6]. Их важным преимуществом является то, что они используют различные методы извлечения знаний, способны уточнять запрос, учитывать контекст, также могут использовать методы машинного обучения. Таким образом, подобные системы способны лучше отвечать на вопросы пользователя, однако у них также есть недостатки.

Существует два вида вопросно-ответных систем: системы, отвечающие на вопросы пользователя, используя внутреннюю базу данных, и системы, порождающие ответы при помощи знаний, содержащихся в Интернете [5]. В первом случае нужной информации может просто не быть в базе, либо она может быть устаревшей, а пользователю обычно не предоставляется возможность каким-либо образом расширить базу. С системами второго типа все немного сложнее. Поскольку они извлекают информацию из Интернета, они обычно пытаются различными способами свести сложные запросы к последовательности простых и далее раскрывать запрос по цепочке. При этом ответы на простые запросы получаются с помощью все тех же поисковых систем. Таким образом, ответ на запрос находится по ключевым словам.

Однако одна и та же информация с семантической точки зрения может быть представлена совершенно по-разному. Это справедливо даже для самых простых запросов. Например, для запроса «Где родился Эйнштейн?» ответ может быть представлен как «Эйнштейн родился в городе Ульм» либо, например, как «Город Ульм подарил миру Эйнштейна». Таким обра-

зом, поиск по ключевым словам «Эйнштейн» и «родился» не обнаружит второй вариант. Для сложных запросов подобные наблюдения еще более существенны.

Таким образом, цель данной работы – разработать программный инструмент, который будет выполнять поиск требуемой информации без необходимости представлять поисковый запрос в виде последовательности нескольких ключевых слов. Определенные методы и технологии семантического поиска были описаны в ряде работ [1–8]. Идея разрабатываемого подхода заключается в автоматизированном построении лингвистических шаблонов для конкретного поискового запроса пользователя. При этом пользователь может «описать» свой поисковый запрос (т. е. поставить в соответствие запросу множество лингвистических шаблонов), если готовый лингвистический шаблон, полностью соответствующий поисковой потребности, отсутствует в базе. Пользователь может контролировать процесс извлечения и порождения шаблонов, для того чтобы избежать построения ошибочных вариантов поисковых запросов. При создании лингвистических шаблонов можно рассматривать параметризованные фрагменты атомарных диаграмм [6] и для них строить семантически параметризованные лингвистические шаблоны.

Понятие параметризованного лингвистического шаблона

Методы, используемые в данной работе, основаны на понятиях фрагмента атомарной диаграммы [9; 10], параметризованного фрагмента атомарной диаграммы [6] и лингвистического шаблона [8; 11; 12]. Атомарной диаграммой модели называется множество истинных на ней атомарных предложений и отрицаний атомарных предложений [9; 10]. Атомарным предложением называется бескванторное предложение вида $P(c_1, \dots, c_n)$, где P – символ n -местного предиката, а $c_1, \dots, c_n$ – символы констант. Фрагмент атомарной диаграммы может быть параметризован, в таком случае часть предикатов, входящих в данный фрагмент атомарной диаграммы, являются заранее не определенными; при реализации информационного поиска предикаты-параметры оснащаются конкретными предикатами [6]. Каждому поисковому запросу пользователя ставится в соответствие конечный фрагмент атомарной диаграммы. Каждому конечному фрагменту атомарной диаграммы соответствует один или несколько лингвистических шаблонов. Таким образом, каждый поисковый запрос пользователя формально определяется множеством лингвистических шаблонов.

Лингвистический шаблон – это параметрический набор условий на искомую структуру текста [8; 11]. Он может быть представлен несколькими способами. Один из возможных вариантов – представление лингвистического шаблона в виде аналога регулярного выражения над множеством слов [8; 11].

Рассмотрим, например, шаблон вида *NOUN*(«им.п.») *NOUN*(«рд.п.»). Данный шаблон описывает последовательность из двух существительных, где первое находится в именительном падеже, а второе – в родительном. Данному шаблону удовлетворяют такие словосочетания, как «отец ребенка», «стены дома» и др. В данном шаблоне параметрически задано значение падежа; таким образом, слова в словосочетании, найденном при помощи данного шаблона, должны удовлетворять условиям, наложенным на падеж. При этом могут варьироваться другие параметры слов, такие как род, число и пр. Один шаблон может содержать несколько параметризаций. Недостатком подобных шаблонов является то, что они неустойчивы к порядку слов. Например, фраза «ребенка отец» уже не удовлетворяет данному шаблону, хотя с семантической точки зрения фразы «отец ребенка» и «ребенка отец» эквивалентны.

Другой вариант шаблона – так называемый древовидный шаблон. Древовидный шаблон – это дерево, в вершинах которого находятся слова, а ребро между вершинами обозначает, что между словами, находящимися в данных вершинах, существует связь. Пример такого шаблона для запроса «Где родился человек?» представлен на рис. 1. Древовидные шаблоны устойчивы к порядку слов, поскольку они учитывают лишь типы связей между словами, но не порядок их в предложении. Например, шаблону с рис. 1 удовлетворяют предложения

Рис. 1. Пример лингвистического шаблона

Fig. 1. Пример лингвистического шаблона

«Эйнштейн родился в Ульме», «Эйнштейн в Ульме родился», «В Ульме родился Эйнштейн» и т. д. В работе используются именно древовидные шаблоны.

Для того чтобы сформировать запрос к поисковой системе, необходимо расположить слова из шаблона в том порядке, в котором они были в изначальном предложении (можно выбрать любой, если вариантов несколько, поскольку поисковые системы не учитывают порядок), а также подставить в запрос конкретный субъект и исключить объект. Для конкретного человека запрос может выглядеть, например, так: «Пушкин родился в».

Порождение лингвистических шаблонов

Сначала опишем общий алгоритм порождения шаблонов:

- 1) поиск предложений, содержащих необходимую информацию;
- 2) извлечение шаблонов из найденных предложений;
- 3) расширение множества предложений и возвращение к шагу 2.

Рассмотрим подробно каждый из этих этапов.

Поиск предложений, содержащих необходимую информацию

На первом этапе необходимо найти предложения, в которых содержится информация о текущем запросе. Мы используем два основных метода получения таких предложений.

1. Национальный корпус русского языка [13; 14] – это информационно-справочная система, основанная на собрании русских текстов в электронной форме. Корпус насчитывает около 30 млн предложений различной тематики, таких как художественные тексты, газетная публицистика, технические тексты, научные тексты, тексты личной переписки и др. Корпус позволяет быстро выполнять лексико-грамматический поиск, используя различные морфологические, грамматические и семантические признаки. Также корпус позволяет получить результаты поиска в удобном для машинной обработки формате. Однако, поскольку корпус вручную обрабатывается и поддерживается, объем текстов в нем значительно меньше, чем в Интернете. Поэтому данный метод используется в основном на первом этапе, где выполняется поиск более простых предложений.

2. Поисковая система Гугл¹. Всем известная система выполняет поиск по всей сети и предоставляет результаты в удобочитаемом для человека формате. Для автоматического извлечения предложений необходимо потратить больше ресурсов, чем для извлечения с помощью корпуса русского языка, однако множество извлеченных предложений гораздо больше по объему и разнообразию.

На первом этапе необходимо найти наиболее простые предложения, отвечающие требованиям. В основном поиск производится по ключевым словам без учета морфологических признаков. Например, для запроса «Где родился человек?» можно выполнить поиск по ключевому слову «родиться». Очевидно, что не все предложения, содержащие слово «родиться», будут содержать информацию о месте рождения человека. Поэтому полученные данные необходимо обработать вручную, чтобы отсеять ненужные варианты.

Извлечение шаблонов из найденных предложений

Далее, по найденным предложениям нужно построить лингвистические шаблоны. Для получения шаблонов используются синтаксические деревья предложений, извлеченных на первом шаге. Синтаксическое дерево (дерево разбора) – структура в виде ориентированного дерева, представляющая синтаксическую структуру предложения. Пример синтаксического

¹ www.google.com

дерева для предложения «Дельта Волги является восьмым чудом света» приведен на рис. 2. Корректно построенное дерево разбора обладает следующими свойствами:

1) слова, находящиеся на разных вершинах одного ребра, связаны в исходном предложении. Причем слово, находящееся на нижнем уровне, является зависимым. Слово, находящееся в корне дерева, не имеет зависимого.

2) слова, находящиеся в разных ветках, не имеют прямой связи друг с другом и связаны только через зависимые слова. Например, слова «восьмым» и «света» не связаны напрямую, но они оба являются характеристиками слова «чудо».

Рис. 2. Пример синтаксического дерева

Fig. 2. Syntax tree example

Таким образом, можно сделать следующие выводы. Если мы хотим выяснить, как два слова связаны в исходном предложении, нам достаточно найти минимальное поддереву, которое содержит в себе искомые слова. В данное дерево не попадут следующие категории вершин:

1) вершины, располагающиеся на более нижних уровнях, чем вершины, содержащие искомые слова. Очевидно, что данные вершины не имеют смысла включать в дерево, так как искомые слова никак (даже транзитивно) от них не зависят;

2) вершины, не являющиеся непосредственными предками искомым, но находящиеся на том же уровне или выше. Данные вершины не имеют прямой связи с искомыми, а значит, их также не имеет смысла включать в итоговое дерево;

3) вершины, находящиеся выше, чем наименьший общий предок искомым вершин. С семантической точки зрения, слова, находящиеся в таких вершинах, несут более глобальный смысл, чем исходные слова. Такая ситуация может возникнуть, если оба исходных слова находятся в придаточной части предложения, которая сама по себе зависит от других частей предложения. Очевидно, что вершины из данной категории также не нужно включать в искомое дерево.

Возвращаясь к лингвистическим шаблонам, можно заметить, что итоговое дерево и есть искомый шаблон, если в качестве искомым слов использовать *субъект* и *объект* исходного

Оба метода имеют свои плюсы и свои минусы. В своей работе мы используем парсер Solarix [17]. Это парсер с общедоступной лицензией, основанный на правилах. Грамматика языка была описана авторами парсера, поэтому данную задачу в этой работе мне решать не придется. Парсер решает следующие задачи:

- 1) лексический анализ – разбивка текста на предложения и слова;
- 2) морфологический анализ – определение морфологических признаков с учетом контекста;
- 3) лемматизация – приведение слов к начальной форме;
- 4) синтаксический анализ – определение синтаксических связей слов в предложении, построение дерева разбора.

Таким образом, парсер решает все необходимые задачи, и для построения шаблонов необходимо ввести в парсер множество полученных предложений. Существует два режима работы парсера.

1. Восходящий анализ. Идея метода заключается в разборе сначала конкретных слов, затем в связывании слов в пары, далее связывание пар с другими словами и парами и т. д., пока все слова предложения не окажутся связанными в одну структуру. Такой метод работает достаточно быстро, однако, если грамматика не полностью описывает данное предложение, либо предложение содержит неоднозначности, парсер может сделать ошибочные предположения о связи слов и в результате получить неточный результат. В целом полученные результаты достаточно корректны, и выбросы (результаты, сильно отличающиеся от среднего) встречаются довольно редко.

2. Нисходящий анализ. Идея метода заключается в выдвижении гипотез о крупномасштабной структуре предложения, далее эта структура уточняется, рекурсивно спускаясь до уровня отдельных слов. Данный метод работает очень медленно, поскольку, сделав ошибочное предположение на ранних этапах и опровергнув его на более поздних, алгоритм выполняет большое количество лишних действий. Однако данный подход может выдать все возможные варианты парсинга предложения в случае наличия неоднозначностей.

Поскольку быстрый восходящий анализ работает корректно в большинстве случаев, целесообразно использовать именно его для выполнения первичной обработки предложений. Переключаться на нисходящий анализ разумно только в случае, если восходящий анализ покажет неудовлетворительные результаты. На выборке из 50 предложений восходящий анализ показывает результат 3 секунды, тогда как нисходящий анализ требует 1,5 минуты.

В результате анализа полученных шаблонов были выявлены две категории шаблонов, которые должны быть исключены из итоговых результатов.

1. Шаблоны, содержащие только слова ОБЪЕКТ и СУБЪЕКТ. Подобные шаблоны могут возникнуть, если в полученном дереве разбора субъект и объект находятся в соседних узлах. В целом такая ситуация является корректной, но очевидно, что данные шаблоны невозможно использовать для дальнейшего поиска, следовательно, они должны быть отсеяны.

2. Шаблоны, встречающиеся менее чем в 0,5 % случаев. Поскольку парсер показывает не 100 % верные результаты, необходимо сделать его более устойчивым к ошибкам. Один из способов – отсеять шаблоны, которые встречаются очень редко, поскольку они, скорее всего, были получены в результате ошибочного анализа. Величина 0,5 % была получена чисто эвристически.

Стоит также отметить, что в полученном шаблоне на месте субъекта и объекта могут находиться не только отдельные слова, но также и словосочетания, см. пример на рис. 4. Однако весь анализ для таких случаев остается неизменным, данное предложение также удовлетворяет шаблону из рис. 1.

Расширение множества предложений

После того как полученные предложения проанализированы и получен набор шаблонов, необходимо выявить более редкие предложения, которые были упущены на первом этапе, и также извлечь из них шаблоны. Для выявления более редких предложений мы используем

Рис. 4. Субъект и объект представлены несколькими словами

Fig. 4. Subject and object are presented as several words

поисковую систему Гугл ввиду наличия большого объема и разнообразия данных. Например, на первом этапе для предиката «Где родился человек?» используются все предложения, содержащие слово «родиться». Очевидно, что большое количество предложений при этом теряется, поскольку в предложении, где идет речь о месте рождения, наличие слова «родиться» совсем необязательно.

Для того чтобы получить больше предложений, необходимо из существующих выбрать несколько таких, в которых субъект и объект широко представлены в текстах, находящихся в Интернете. Далее, сделав запрос вида «СУБЪЕКТ ОБЪЕКТ» в поисковой системе, мы получим множество различных вариантов того, как субъект и объект могут быть связаны в исходном предложении. Конечно, данные предложения могут содержать информацию, которая не относится к текущему предикату, поэтому полученные предложения также должны быть обработаны вручную.

Далее из полученных предложений извлекаются лингвистические шаблоны аналогично тому, как это было сделано на первом этапе. При желании можно повторить второй и третий этапы для получения большего количества шаблонов.

Примеры обработки предикатов

В качестве примера рассмотрим процесс получения шаблонов для вопроса «Где родился человек?». На первом этапе с помощью корпуса русского языка извлекаются около 300 шаблонов, содержащих слово «родиться». При этом поиск проводится без учета морфологических признаков, т. е. варианты со словами «родился», «родилась» и т. д. также представлены в итоговых результатах. Из полученных предложений примерно 50 содержат информацию о месте рождения человека. В результате проведенного анализа был получен набор новых лингвистических шаблонов. Вот самые популярные из них:

- 1) СУБЪЕКТ родился в ОБЪЕКТ: 19 вхождений в исходное множество;
- 2) ОБЪЕКТ СУБЪЕКТ родился: 4 вхождения;
- 3) СУБЪЕКТ родилась в году в ОБЪЕКТ: 3 вхождения.

Далее мы выбираем следующие пары «субъект – объект» для последующего анализа: Эйнштейн – Ульм, Ньютон – Вулсторп, Чехов – Таганрог. Проанализировав около 120 результатов запросов, мы получаем около 40 различных предложений, удовлетворяющих предикату. В результате дальнейшего анализа, помимо шаблонов, аналогичных полученным на первом этапе, также были получены принципиально новые варианты, такие как: «ОБЪЕКТ родина СУБЪЕКТ», «Из уроженцев ОБЪЕКТ известен СУБЪЕКТ», «ОБЪЕКТ город подаривший миру СУБЪЕКТ» и др.

Другой пример более сложного предиката: «Кто является президентом страны?». Самые популярные шаблоны, полученные на первом этапе по ключевому слову «президент»:

- 1) Президент СУБЪЕКТ ОБЪЕКТ;
- 2) ОБЪЕКТ, президент СУБЪЕКТ;
- 3) ОБЪЕКТ является президентом СУБЪЕКТ.

Шаблоны второго этапа, не найденные на первом:

- 1) Глава СУБЪЕКТ ОБЪЕКТ;
- 2) ОБЪЕКТ на посту главы СУБЪЕКТ.

Полученные результаты

Разработана программная система, реализующая описанный выше алгоритм. Для проверки корректности работы алгоритма была проведена серия тестов. В программную систему были введены 3 запроса: «Место рождения человека», «Автор изобретения», «Президент страны». Для каждого запроса протестировано 10 возможных субъектов, результаты работы оценивались экспертом. Оценка ставилась из множества $\{0|1\}$, где 1 ставилась в случае, если на найденной странице действительно содержалась искомая информация, и 0 в противном случае.

В ходе анализа полученных результатов выявлены 3 типа ошибок, т. е. все полученные ошибочные результаты можно разделить на 3 категории.

1. Ошибка первой категории возникает в случае, если предложение удовлетворяет шаблону, однако в более глобальном контексте было сказано, что это предложение ошибочно. Пример: «Они ошибочно полагали, что лампочку изобрел Томас Эдисон». Стоит отметить, что во всех случаях на этой же странице содержалась информация о реальном положении вещей, поэтому такие страницы были отмечены как успешный результат.

2. Ошибки второй категории возникали в случае, если предложение удовлетворяет шаблону, но все равно не отвечает на запрос. Пример такой ошибки для предиката «Место рождения»: «Александр Пушкин родился в дворянской семье». Чтобы избежать данной ошибки, необходимо учесть более широкий контекст, чем предложение. Ведь теоретически, «дворянская семья» вполне может быть названием населенного пункта. Стоит отметить, что подобные ошибки встречались достаточно редко, хотя, несомненно, они являются наиболее опасными.

3. Ошибки третьей категории появляются в случае, если в момент последнего редактирования страницы информация была еще действительна, но на текущий момент она устарела. Или у автора была некорректная информация, или же он намеренно вводит читателя в заблуждение. Пример: «Президент США – Барак Обама». Подобные ошибки даже сложно считать ошибкой, ведь в таком случае даже человек не сможет удостовериться, что информация действительно верная. Однако они имеют место, поэтому также были отмечены в текущей работе.

Всего для каждого запроса программа выдает 5 результатов с предполагаемым ответом. Таким образом, для каждого предиката было получено 50 результатов. Далее приведена таблица с количеством ошибок каждого типа для каждого запроса.

Количество ошибок, %
Number of Errors, %

Запрос	Категория ошибок		
	первая	вторая	третья
Место рождения	0	10	10
Автор изобретения	0	0	4
Президент страны	0	8	14

Можно заметить, что ошибок первой категории в данном случае не было, хотя они и были обнаружены в процессе тестирования. Можно предположить, что такие ошибки аналогичны ошибкам третьей категории, однако между ними есть существенная разница. В первом случае в тексте явно сказано, что данная информация не верна, в третьем же случае информация представляется корректной. Ну и логично, что в первом случае сразу же приводится корректная информация.

Что касается ошибок второй категории, можно понять, что количество таких ошибок зависит от запроса. Например, для запроса *«автор изобретения»* алгоритм не допустил подобных ошибок. Ну, действительно, сложно придумать предложение, удовлетворяющее, например, шаблону *«СУБЪЕКТ изобрел ОБЪЕКТ»* и имеющее какой-то посторонний смысл.

Всего было получено 150 результатов работы программы, из которых 127 были корректными. Таким образом, можно сказать, что на тестовой выборке точность алгоритма составляет 84,6 %, причем для каждой пары «субъект – запрос» как минимум 3 из 5 результатов были корректными. В целом полученные результаты можно считать успешными.

Заключение

В работе предложен алгоритм автоматизированного извлечения лингвистических шаблонов из текстов естественного языка. Были использованы древовидные лингвистические шаблоны в силу того, что они являются более устойчивыми к порядку слов в предложении.

Разработаны и протестированы методы построения синтаксических деревьев по предложениям естественного языка. Предложены критерии определения валидности различных лингвистических шаблонов.

Разработана и реализована программная система, позволяющая осуществлять семантический поиск и извлекать требуемые знания в текстах естественного языка, представленных в Интернете. Разработанная программная система протестирована, полученная точность 84,6 % представляется достаточно хорошей для решения рассмотренных задач и достижения поставленных целей.

Список литературы

1. **Осипов Г. С., Тихомиров И. А., Смирнов И. В.** Семантический поиск в сети Интернет средствами поисковой машины Exactus. URL: <https://refdb.ru/look/3439993.html> (дата обращения 18.05.2019).
2. **Нгуен Ба Нгок, Тузовский А. Ф.** Обзор подходов семантического поиска // Докл. ТУ-СУРа. 2010. № 2 (22), ч. 2. С. 234–237.
3. **Перминов С. В., Афанасьев С. В.** Семантический способ поиска информационных аномалий через web // Тр. СПИИРАН. 2006. Т. 3, № 1. С. 279–287.
4. **Guha R., McCool R., Miller E.** Semantic search. In: Proc. of the 12th Int. Conf. on World Wide Web, 2003, p. 700–709. DOI 10.1145/775152.775250
5. **Пальчунов Д. Е.** Поиск и извлечение знаний: порождение новых знаний на основе анализа текстов естественного языка // Философия науки. 2009. № 4 (43). С. 70–90.
6. **Деревянко Д. В., Пальчунов Д. Е.** Формальные методы разработки вопросно-ответной системы на естественном языке // Вестник НГУ. Серия: Информационные технологии. 2014. Т. 12, № 3. С. 34–47.
7. **Власов Д. Ю., Пальчунов Д. Е., Степанов П. А.** Автоматизация извлечения отношений между понятиями из текстов естественного языка // Вестник НГУ. Серия: Информационные технологии. 2010. Т. 8, № 3. С. 23–33.
8. **Пальчунов Д. Е., Степанов П. А.** Применение теоретико-модельных методов извлечения онтологических знаний в предметной области информационной безопасности // Программная инженерия. 2013. № 11. С. 8–16.
9. **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.
10. **Корсун И. А., Пальчунов Д. Е.** Теоретико-модельные методы извлечения знаний о смысле понятий из текстов естественного языка // Вестник НГУ. Серия: Информационные технологии. 2016. Т. 14, № 3. С. 34–48.

11. **Степанов П. А.** Язык описания лингвистических шаблонов // Материалы Всерос. конф. с междунар. участием «Знания – Онтологии – Теории». 2013. Т. 2. С. 136–145.
12. **Степанов П. А., Пальчунов Д. Е., Мирзагитов А. А.** Методы анализа диалогов, основанные на теории речевых актов // Вестник НГУ. Серия: Информационные технологии. 2014. Т. 12, № 4. С. 102–111.
13. **Аброскин А. А.** Поиск по корпусу: проблемы и методы их решения // Национальный корпус русского языка: 2006–2008. Новые результаты и перспективы. 2009. С. 277–282.
14. **Апресян Ю. Д., Богуславский И. М., Иомдин Б. Л.** Синтаксически и семантически аннотированный корпус русского языка: современное состояние и перспективы // Национальный корпус русского языка: 2003–2005. С. 193–214.
15. **Антонова А. А.** Синтаксический анализатор для русского и английского языков // Информационно-аналитические аспекты в задачах управления: Сб. тр. ИСА РАН. М.: ЛКИ, 2007. Т. 29. С. 329–337.
16. Парсим русский язык // Хабрахабр. 2012. 20 июля. URL: <https://habr.com/ru/post/148124/> (дата обращения 18.05.2019).
17. Парсер – морфологический и синтаксический анализатор русскоязычных текстов // Инструменты для NLP разработчика: лексика, морфология, синтаксис русского языка. 2019. URL: <http://solarix.ru/parser.shtml> (дата обращения 18.05.2019).

References

1. **Osipov G. S., Tikhomirov I. A., Smirnov I. V.** Semanticheskii poisk v seti Internet sredstvami poiskovoi mashiny Exactus [Semantic search on the Internet by means of a search engine Exactus]. URL: <https://refdb.ru/look/3439993.html> (accessed 18.05.2019). (in Russ.)
2. **Ba Ngoc Nguyen, Tuzovsky A. F.** A review of semantic search approaches. *Proc. of TUSUR University*, 2010, № 2 (22), part 2, p. 234–237. (in Russ.)
3. **Perminov S. V., Afanasyev S. V.** Semantic Search for Information Anomalies through Web Technique. *SPIIRAS Proc.*, 2006, vol. 3, no. 1. (in Russ.)
4. **Guha R., McCool R., Miller E.** Semantic search. In: *Proc. of the 12th Int. Conf. on World Wide Web*, 2003, p. 700–709. DOI 10.1145/775152.775250
5. **Palchunov D. E.** Knowledge retrieval and elicitation: generation of new knowledge based on analysis of natural language texts. *Filosofiya nauki*, 2009, no. 4 (43), p. 70–90. (in Russ.)
6. **Derevyanko D. V., Palchunov D. E.** Formal methods of development of the question-answering system on natural language. *Vestnik NSU. Series: Information Technology*, 2014, vol. 12, no. 3, p. 34–47. (in Russ.)
7. **Vlasov D. Yu., Palchunov D. E., Stepanov P. A.** Automating the extraction of relations between concepts from natural language texts. *Vestnik NSU. Series: Information Technology*, 2010, vol. 8, no. 3, p. 23–33. (in Russ.)
8. **Palchunov D. E., Stepanov P. A.** The use of model-theoretic methods for extracting ontological knowledge in the domain of information security. *Programnaya ingeneriya*, 2013, no. 11, p. 8–16. (in Russ.)
9. **Makhasoeva O. G., Palchunov D. E.** Semi-automatic methods of a construction of the atomic diagrams from natural language texts. *Vestnik NSU. Series: Information Technology*, 2013, vol. 12, no. 2, p. 64–73. (in Russ.)
10. **Korsun I. A., Palchunov D. E.** Model-Theoretic Methods of Extraction of Knowledge on the Meaning of Concepts from the Natural Language Texts. *Vestnik NSU. Series: Information Technology*, 2016, vol. 14, no. 3, p. 34–48. (in Russ.)
11. **Stepanov P. A.** Linguistic patterns description language. In: *All-Russian Conference with international participation “Knowledge – Ontology – Theory”*. 2013, vol. 2, p. 136–145. (in Russ.)

12. **Stepanov P. A., Palchunov D. E., Mirzagitov A. A.** Methods of analysis of dialogues based on the theory of speech acts. *Vestnik NSU. Series: Information Technology*, 2014, vol. 12, no. 4, p. 102–111. (in Russ.)
13. **Abroskin A. A.** Poisk po korpusu: problem i metody ikh resheniya [Search in corpus: problems and methods of their solutions]. In: *Natsional'nyi korpus russkogo yazyka: 2006–2008. Novye rezul'taty i perspektivy*. 2009, p. 277–282. (in Russ.)
14. **Apresyan Yu. D., Boguslavskii I. M., Iomdin B. L.** Sintaksicheski i semanticheski annotirovannyi korpus russkogo yazyka: sovremennoe sostoyanie i perspektivy [Syntactically and semantically annotated corpus of the Russian language: current state and prospects]. In: *Natsional'nyi korpus russkogo yazyka: 2003–2005*. p. 193–214. (in Russ.)
15. **Antonova A. A.** Syntax analyzer for the Russian and English languages. In: *Information and analytical aspects in problems of management*. Moscow, LKI Publ., 2007, vol. 29, p. 329–337. (in Russ.)
16. **Parsim russkii yazyk** [Parse Russian language]. *Habrahabr*, 2012, June, 20. URL: <https://habr.com/ru/post/148124/> (accessed 18.05.2019). (in Russ.)
17. **Parser – morfologicheskii i sintaksicheskii analizator russkoyazychnykh tekstov** [Parser – morphological and syntactic analyzer of Russian texts]. In: *Instrumenty dlya NLP razrabotchika: leksika, morfologiya, sintaksis russkogo yazyka*, 2019. URL: <http://solarix.ru/parser.shtml> (accessed 18.05.2019). (in Russ.)

Материал поступил в редколлегию

Received

05.06.2019

Сведения об авторах / Information about the Authors

Филиппов Иннокентий Игоревич, студент, Новосибирский государственный университет (ул. Пирогова, 1, Новосибирск, 630090, Россия)

Innokentiy I. Filippov, Student, Novosibirsk State University (1 Pirogov Str., Novosibirsk, 630090, Russian Federation)

i.filippov1@g.nsu.ru

Пальчунов Дмитрий Евгеньевич, доктор физико-математических наук, ведущий научный сотрудник, Институт математики СО РАН (пр. Академика Коптюга, 4, 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)

palch@math.nsc.ru

Степанов Павел Андреевич, ассистент, Новосибирский государственный университет (ул. Пирогова, 1, Новосибирск, 630090, Россия)

Pavel A. Stepanov, Assistant, Novosibirsk State University (1 Pirogov Str., Novosibirsk, 630090, Russian Federation)

stefan.nsk@gmail.com