**Введение**

Для обращения к системе связных данных (LOD) используется язык SPARQL. Но для людей не знакомых с ним это делает использование LOD абсолютно невозможным (SPARQL не привычен и сложен для обывателя). Для решения этой проблемы исследователями, работающими над данной проблемой, были разработаны естественно-языковые интерфейсы. Они позволяют обращаться к связным данным с помощью запросов на естественном языке. Но все работы, сделанные в данной области, в качестве естественного языка предполагают использование английского или его подъязыков. Поэтому, целью данной работы ставится разработка интеллектуального естественно-языкового интерфейса для обращения к LOD на русском языке.

**Описания глав**

**Краткие сведения о семантической информационной системе LOD и ее применениях**

LOD (Linked Open Data) – открытые связные данные – система, подразумевающая использование методов публикации и связывание структурированных данных в Интернете. Основывается на четырех принципах, введенных Тимом Бернерс-Ли:

1. Необходимо использование URI () для объектов реального мира и абстрактных понятий, а не только для веб-документов и цифровых ресурсов;
2. Необходимо использование HTTP URI, что позволит использовать протокол HTTP для доступа к ресурсам;
3. Необходимо использование RDF (Resource Description Framework) в качестве единой модели данных для публикации данных;
4. Необходимо включать RDF утверждения на другие URI таким образом, чтобы была возможность обнаружения связанных данных.

RDF (Resource Description Framework) – фреймворк описания ресурсов. Представляет собой совокупность утверждений о ресурсах в виде, удобном для машинной обработки (формат RDF).

Формальная модель RDF состоит из следующих множеств:

* Множество «Ресурсы»;
* Множество «Литералы»;
* Подмножество «Ресурсов» – «Свойства»;
* Множество «Утверждения».

Последнее множество состоит из троек вида:

{субъект, предикат, объект},

где субъект – это элемент множества «Ресурсы»,

предикат – это элемент множества «Свойства»,

объект – это элемент либо множества «Ресурсы», либо множества «Литералы».

Но сама модель данных RDF не дает возможности ни для объявления свойств, ни для определения связей между свойствами и другими ресурсами. Для выполнения данных функций используется RDF Schema (он же RDFS).

Применение

**Основные конструкции языка запросов SPARQL**

SPARQL (рекурсивный акроним SPARQL Protocol and RDF Query Language) – язык запросов к данным, представленным в формате RDF. SPARQL для LOD является таким же инструментом, как и SQL для реляционных баз данных. SPARQL имеет сходную с SQL форму запроса, а именно «SELECT FROM WHERE», т.е. «Что выбрать», «В каком ресурсе» и «Каким параметрам должно соответствовать».

Простейший запрос SPARQL представлен в листинге 2. В качестве данных, к которым применяется запрос, используется RDF граф, приведенный в листинге 1.

Листинг 1 – RDF-тройка

<http://example.org/book/book1> <http://purl.org/dc/elements/1.1/title> "SPARQL Tutorial" .

Листинг 2 – Простейший запрос на языке SPARQL

SELECT ?title WHERE {<http://example.org/book/book1> <http://purl.org/dc/elements/1.1/title> ?title . }

Результатом выполнения данного запроса является строка «SPARQL Tutorial».

Запрос приведенный выше осуществляет поиск книги по заданному графу RDF. Запрос состоит из двух частей: части SELECT, в которой определяются переменные, которые будут отображаться в результатах запроса; и части WHERE, в которой представлен шаблон графа для сопоставления с графом данных, в данном запросе шаблон состоит из графа с одной переменной.

RDF предоставляет возможность использования литералов трех варианты литералов трех вариантов: с пометкой (тегом) языка, литерал со стандартным типом данных и литерал с произвольным типом данных (возможен еще с четвертый тип литерала – простой литерал без тегов языка и типов, но в следующих трёх примерах запросов SPARQL он рассматриваться не будет). Пример данных с использование трёх типов литералов приведены в листинге 3.

Листинг 3 – RDF граф с использованием литералов

@prefix dt: <http://example.org/datatype#> .

@prefix ns: <http://example.org/ns#> .

@prefix : <http://example.org/ns#> .

@prefix xsd: <http://www.w3.org/2001/XMLSchema#> .

:x ns:p "cat"@en .

:y ns:p "42"^^xsd:integer .

:z ns:p "abc"^^dt:specialDatatype .

В листингах 4 и 5 приведены на первый взгляд похожие запросы, но запрос в листинге 4 не будет иметь соответствие в указанных выше данных, в отличие от запроса в листинге 5, поскольку литералы «cat» и «cat@en» не являются одинаковыми.

Листинг 4 – Запрос с литералом без тега языка

SELECT ?v WHERE { ?v ?p "cat" }

Листинг 5 – Запрос с некорректным литералом

SELECT ?v WHERE { ?v ?p "cat@en" }

В листинге 6 приведен пример запроса для поиска литерала со стандартным типом. При использование в литерале стандартного типа возможно использование сокращенной формы записи, например просто 23 вместо "23"^^<http://www.w3.org/2001/XMLSchema#integer>. В этом случае задача определения конкретного типа возлагается на программное обеспечение реализующее выполнение запроса.

Листинг 6 – Пример запроса с литералом со стандартным типом

SELECT ?v WHERE { ?v ?p 42 }

Следующий пример — это запрос с литералом с произвольным типом (листинг 7). В данном случае литерал должен указываться полностью в явном виде, но при выполнении такого запроса обработчик запроса не обязан знать о указанном типе данных, поиск производится по сравнению литерала.

Листинг 7 – Пример запроса с литералом с произвольным типом

SELECT ?v WHERE { ?v ?p "abc"^^<http://example.org/datatype#specialDatatype> }

Результат выполнения запроса «http://example.org/ns#x».

Ограничения в WHERE

Возможности ограничения выборки искомых ресурсов будет показано на основе данных, указанных в листинге 8.

Листинг 8 – Данные для демонстрации запросов с ограничениями на искомую выборку

@prefix dc: <http://purl.org/dc/elements/1.1/> .

@prefix : <http://example.org/book/> .

@prefix ns: <http://example.org/ns#> .

:book1 dc:title "SPARQL Tutorial" .

:book1 ns:price 42 .

:book2 dc:title "The Semantic Web" .

:book2 ns:price 23 .

Язык SPARQL позволяет ограничивать выборку по содержанию строковыми литералами определенных значений, по числовым литералам на основе арифметических признаков (больше, меньше и т.п.) также возможны органичения по логическому типу и по дате (до указанной даты, после и т.п.).

В листингах 9 и 10 приведены запросы, ограничивающий выборку, по содержимому строкового литерала.

Листинг 9 – Запрос, ограничивающий выборку, ресурсами, название которых начинается с подстроки «SPARQL»

PREFIX dc: <http://purl.org/dc/elements/1.1/>

SELECT ?title

WHERE { ?x dc:title ?title

FILTER regex(?title, "^SPARQL")

}

При применении запроса к данным из листинга 8, результат выполнения запроса будет следующим

|  |
| --- |
| **title** |
| "SPARQL Tutorial" |

Листинг 10 – Запрос, ограничивающий выборку, ресурсами, название которых содержит подстроку «web»

PREFIX dc: <http://purl.org/dc/elements/1.1/>

SELECT ?title

WHERE { ?x dc:title ?title

FILTER regex(?title, "web", "i" )

}

При применении запроса к данным из листинга 8, результат выполнения запроса будет следующим

|  |
| --- |
| **title** |
| "The Semantic Web" |

В листингах 11 и 12 приведены примеры запросов ограничивающих выборку на основе значений численных литералов.

Листинг 11 – Запрос с ограничением выборки по числовому литералу

PREFIX dc: <http://purl.org/dc/elements/1.1/>

PREFIX ns: <http://example.org/ns#>

SELECT ?title ?price

WHERE { ?x ns:price ?price .

FILTER (?price < 30.5)

?x dc:title ?title . }

При применении запроса к данным из листинга 8, результат выполнения запроса будет следующим

|  |  |
| --- | --- |
| **title** | **price** |
| "The Semantic Web" | 23 |

Запросы с ограничением по логическому типу и дате имеют структуру запроса аналогичную запросам с ограничением по числовому типу.

**Основные современные подходы к формальному описанию семантической структуры текстов на естественном языке (ЕЯ)**

Одним из подходов к формальному описанию текстов на ЕЯ является AMR (Abstract Meaning Representaion) [1]. В рамках данного подхода каждое предложение представляется в виде корневого, направленного ациклического графа с метками на ребрах (отношения между понятиями) и листьях (сами понятия) [2]. В качестве понятий в AMR используются английские слова («boy» как в примере выше), фреймы из PropBank («believe-01») или ключевые слова, включающие специальные типы сущностей: даты, региона, расстояния и др, и союзы связки: «и», «или».

AMR учитывает все слова предложения, но при этом он является абстрактным, то есть один граф может представлять несколько разных предложений на естественном языке со схожим смыслом. В качестве примера можно привести следующий набор предложений:

* The boy desires the girl to believe him;
* The boy desires to be believed by the girl;
* The boy has a desire to be believed by the girl;
* The boy’s desire is for the girl to believe him.

У перечисленных выше предложений будет одинаковое описание семантической структуры (листинг 1).

Листинг 1 – Пример AMR

(w / want-01

:ARG0 (b / boy)

:ARG1 (b2 / believe-01

:ARG0 (g / girl)

:ARG1 b))

В этом примере проявляется возможность AMR абстрагироваться от синтаксической структуры, что позволяет выделять предложения с разной синтаксической структурой, но с одинаковой семантической.

Хоть AMR и не зависит от синтаксических особенностей, но он рассчитан только для описания семантической структуры предложений только на английском языке и без взаимосвязи с предыдущими предложениями.

Грамматика Монтегю и К-представление

**Основные подходы к разработке семантически-ориентированных ЕЯ-интерфейсов для взаимодействия с прикладными интеллектуальными системами**

**Что это вообще???**

**Основные подходы к разработке интеллектуальных интерфейсов для преобразования запроса к LOD на ЕЯ в запросы на языке SPARQL**

Аналогичными работа по разработке интеллектуальных интерфейсов для преобразования запроса к LOD на ЕЯ в запросы на языке SPARQL занимается французский исследователь Себастьян Ферре (Sébastien Ferré). Одной из его разработок является SQUALL. Он описан в двух работах Ферре: SQUALL: a Controlled Natural Language for Querying and Updating RDF Graphs и SQUALL: A Controlled Natural Language as Expressive as SPARQL 1.1.

SQUALL является контролируемым языком, т.е. такой версией естественного языка, которая получена с помощью ограничения в использовании грамматической вариативности, определенных речевых оборотов. Перечисленные выше ограничения проводятся с целью устранения (уменьшения) многозначности и сложность, что в свою очередь обеспечивает формальную логическую основу, т.е. формальные семантику и синтаксис. А язык с формальными семантикой и синтаксисом может быть сопоставлен с другим формальным языком. При разработке языка SQUALL Ферре придерживался следующих двух требований:

1. язык должен обладать выразительностью сопоставимой с выразительностью языка SPARQL;
2. обладая выразительностью SPARQL, язык должен быть высокоуровневым и иметь естественный синтаксис, абстрагированный от низкоуровневых понятий, таких как реляционная алгебра, которая используется в языке SPARQL.

Для перевода из языка SQUALL в язык SPARQL используется промежуточный язык – грамматика Монтегю. Для упрощения представления и дальнейшего перевода в SPARQL некоторые конструкторы заменяются на понятия, которые они отображают.

Перевод из промежуточного языка в SPARQL осуществляется с помощью сравнения логических конструкций с конструкциями языка SPARQL.

Формулы с конструкторами переводятся в запросы SPARQL, другие типы формул переводятся в запросы обновления SPARQL. Встреча при переводе переменной “?x” предполагает создание новой переменной SPARQL. Предикаты преобразуются в фильтры SPARQL, агрегации – в агрегативные подзапросы SPARQL.

Надо бы примеры и чью-то еще реализацию

**Список литературы**

1. Banarescu, L., Bonial, C., Cai, S., Georgescu, M., Griffitt, K., Hermjakob, U., Knight, K., Koehn, P., Palmer, M., Schneider, N. (2013). Abstract Meaning Representation for Sembanking. In: Proceedings of the 7th ACL Linguistic Annotation Workshop and Interoperability with Discourse, Sofia, Bulgaria, August 8-9, 2013 (2013)([www.aclweb.org/anthology/W13-2322](http://www.aclweb.org/anthology/W13-2322));
2. Banarescu, L., Bonial, C., Cai, S., Georgescu, M., Griffitt, K., Hermjakob, U., Knight, K., Koehn, P., Palmer, M., Schneider, N. (2019).   *Abstract Meaning Representation (AMR) 1.2.6 Specification*; [github.com/amrisi/amr-guidelines/blob/master/amr.md](http://github.com/amrisi/amr-guidelines/blob/master/amr.md);