**Слайд 2**

В современном мире интернет продолжает свое развитие и подходит к следующей стадии, которая называется Семантический веб (или интернет данных). На данной стадии предполагается, что данные становятся не только машиночитаемыми, но и «машинопонимаемыми». А благодаря машинопонимаемости возможно упростить, улучшить и ускорить поиск информации во многих сферах: научные публикации, бизнес, системы работы с географическими данными и многое другое. И в качестве технологической основы Семантического веба используется система открытых взаимосвязанных данных (Linked Open Data или сокращенно LOD).

Но единственным средством взаимодействия с системой LOD является язык запросов SPARQL. А он очень сложен в использовании для неспециалистов, что ограничивает использование системы LOD. В таком случае, очевидным решением является предоставление пользователю наиболее привычного для него интерфейса, а именно естественно-языкового. В связи с этим и появилась данная работа, целью которой является разработка естественно-языкового интерфейса для взаимодействия с системой открытых взаимосвязанных данных.

Для разработки такого интерфейса необходимо:

1. выбрать подход к семантическому представлению, т.е. выбрать промежуточное представление, которое будет описывать семантическую (смысловую) структуру входного запроса;
2. разработать алгоритмы для реализации преобразований «ЕЯ-запрос → SPARQL-запрос», в общем, и, в частности, алгоритмы для реализации преобразований «ЕЯ-запрос→Семантическое представление» и «Семантическое представление → SPARQL-запрос».

**Слайд 3**

Теперь стоит немного рассказать о самой системе открытых взаимосвязанных данных. Систему LOD можно представить в виде огромного, гигантского размеченного ориентированного графа, который в свою очередь состоит из элементарных графов, представляющих тройки (Субъект, Предикат, Объект). Таким образом обеспечивается, в первую очередь, взаимосвязанность данных, а, во вторую, их смысловая структура.

Система LOD состоит из отдельных онтологий (баз знаний). Но это разделение на части не отменяет взаимосвязанности даннывх.

**Слайд 4**

Для внутреннего представления данных в системе LOD используется язык RDF. Основной структурой данного языка являются триплеты – упорядоченные тройки вида (Субъект, Предикат, Объект). Это те самые тройки – элементарные графы из которых состоит LOD.

**Слайд 5**

Как я уже сказал, основным средством взаимодействия с системой взаимосвязанных данных является язык SPARQL. Он предоставляет возможность организации запросов к данным в формате RDF. SPARQL имеет SQL-подобный синтаксис, а именно структуру запросов (Select, From, Where) или простыми словами (Что искать?, В чем искать?, Какими свойствами должны обладать искомые сущности?).

На практике блок From опускается, потому что, конкретная система, обеспечивающая выполнение запросов, уже связана с конкретной онтологией.

На левой картинке приведен пример SPARQL-запроса. Данный запрос осуществляет выборку российских городов и результатом его выполнения является список названий городов России.

**Слайд 6**

Теперь можно перейти к выбору подхода к формальному описанию семантического представления текста на ЕЯ. Основными применяемыми подходами являются Абстрактное представление смысла и теория концептуальных представлений (К-представлений) В.А. Фомичева. Грамматика Монтегю является пионерской работой в области формального описания семантического представления, но в некоторых исследованиях она продолжает применяться.

Как можно заметить на слайде, был выбран подход теории К-представлений В.А. Фомичева. Почему выбран именно он, я расскажу далее.

**Слайд 7**

Сравнение проводилось по трем критериям:

* Язык текста;
* Типы текстов;
* Допустимая структура текстов, семантическое представление которых возможно описать.

АПС и грамматика Монтегю рассчитаны на описание семантической структуры текстов только на английском языке. В свою очередь теория К-представлений предоставляет возможность описания семантического представления текстов на русской, французском, немецком, английском и других языках.

Также подходы АПС и грамматики Монтегю имеют ограничения по типу текстов и их структуре: возможно построить СП только для отдельных повествовательных предложений (грамматика Монтегю еще допускает вопросы). Теория К-представлений не имеет подобных ограничений и предоставляет формальный аппарата для описания фраз-высказываний, повествовательных текстов (здесь стоит отметить, именно текстов, а не отдельных предложений), команд, вопросов.

Благодаря этим преимуществам и был выбран подход теории К-представлений.

**Слайд 8**

Для построения СП согласно подходу теории, К-представлений необходимо иметь Лингвистическую Базу Данных (ЛБД). В ней содержится информация о смысловых значениях лексем и возможных семантических отношениях между ними.

ЛБД состоит из 3 компонент:

* Морфологической базы данных;
* Лексико-семантического словаря;
* Словаря предложных фреймов

**Слайд 9**

Морфологическая база данных (МБД) предоставляет информацию о лексемах, терминах (несколько лексем могут выступать в тексте с единым смыслом и такой набор лексем в рамках данной работы называется термином) и морфологических признаков, которые они могут иметь.

Для определения морфологических признаков конкретного слова (падежа, числа, рода, начальной формы и т.п.) используется библиотека DeepMorphy. Для работы в ней применяется нейросеть, обученная для определения морфологических признаков слов на русском языке.

**Слайд 10**

В лексико-семантическом словаре содержится информация о семантических значениях, которые может принимать та или иная лексема или предложный фрейм. У лексемы может быть, как главное, так и дополнительное семантическое значение. Дополнительное семантическое значение определяет с каким смыслом данная лексема может участвовать в отношении с другой лексемой. Например, «контейнер» может иметь главное значение «емкость для хранения» и дополнительные смыслы «пространственный объект» в предложении «контейнер, который наверху», «динамический объект» в предложении «Подвинь тот контейнер» и т.п.

**Слайд 11**

В словаре предложных фреймов хранится информация об отношениях типа «Сущ1 + предлог + Сущ2» и «Сущ1 + Сущ2».

Конкретный фрейм имеет конкретное семантическое значение и определяется по дополнительным семантическим значениям существительных до и после предлога, самим предлогом и падежом, в котором может быть существительное после предлога. Например, словосочетанию «Город в России» может соответствовать предложный фрейм («в», «Простр. Объект», «Простр. Объект», Твор. Падеж, Расположение).

**Слайд 12**

Теперь стоит рассказать о возможных входных запросах для разработанного интерфейса.

Допустимая структура запроса представлена на слайде (Фрагмент1 Сущ1 Предлог Фрагмент2 Сущ2 Фрагмент3).

Фрагмент 1 и Фрагмент2 – являются последовательностью прилагательных. Эти части не обязательны и поэтому могут представляться пустой цепочкой.

Предлог так же, как и Фрагмент1 и Фрагмент2 может быть пустым.

Сущ1 и Сущ2 – существительные, это обязательные части запроса.

Фрагмент3 – может являться Искусственным именем (например, Airbus), словосочетанием сравнения с числом (например, меньше 50000).

**Слайд 13**

На этом слайде представлены запросы, с которыми может работать разработанный интерфейс.

**Слайд 14**

В рамках реализации преобразования «ЕЯ-запрос → СП» строится семантическое представление, со структурой представленной на слайде (А (B1, R1, C1) (B2, R2, C2) … (Bn, Rn, Cn)).

Говоря простым языком, такое представление определяет искомую сущность (буква A) и последовательность параметров и их значений (тройки (B, R, C)), которыми искомая сущность должна обладать.

В тройке (B, R, C) B определяет параметр, C определяет значение этого параметра, R определяет отношения между параметром и значение, т.е. данный параметр должен быть равен данному значению или должен быть меньше этого значения.

На основе данного представления строится SPARQL-запрос.

**Слайд 15**

В ходе разработки алгоритмов для реализации преобразования «Семантическое представление → SPARQL-запрос» пришлось столкнуться с рядом проблем, потребовавших дополнительного исследования.

Первая проблема – неоднозначность именования в онтологии.

В рамках одной и той же онтологии у одного и того же предиката возможно несколько имен. Например, как в таблице справа, Предикат связывающий город с количеством жителей для разных городов носит разное название. Это проблема не дает сделать однотипными SPARQL-запросы даже для однотипных входных запросов.

Проанализировав работы других исследователей, можно выделить 2 подхода к решению данной проблемы. Первый подход заключается создании собственной онтологии, даже с данными из других онтологий, но обязательно с собственной системой имен. Поскольку в таком случае онтология разрабатывается непосредственно для целевой системы, в нее закладывается необходимая семантическая структура и система имен. Но данный подход требует огромных затрат времени на разработку, поэтому в рамках данной работы не применялся.

Вторым решением, является использование онтологий со строго описанной системой имен. Единственной онтологией общего назначения с такой системой имен является онтология YAGO.

**Слайд 16**

Но у этой онтологии имеется другая проблема – недостаточная связность данных. Например, города связаны со страной с помощью предиката комментарий, значением которого является строка свободного формата с указанием принадлежности к стране (например, как на картинке снизу. Это предикат «Комментарий» объекта «Москва»). Хотя по философии связанных данных, это должно было быть реализовано с помощью предиката, например, country, значением которого была бы ссылка на объект «Россия».

Еще пример, у объектов автомобилей полностью отсутствуют содержательные поля: страна разработки, компания-производитель и т.п. Хоть эти данные и присутствует в онтологии, связь между ними не установлена.

Недостаточная связанность данных делает невозможным использование онтологии YAGO.

**Слайд 17**

Поэтому для преодоления проблемы неоднозначности именования в рамках данной работы применялся следующий подход. Он заключается в предварительном связывании параметров, использующихся в семантическом представлении, с параметрами, использующимися в онтологии.

Для этого организуется таблица связи параметров. Например, как в таблице справа.

**Слайд 18**

В качестве реализации таблицы связи в базу данных был добавлен компонент разрешения имен. Он состоит из трех таблиц:

* Таблица параметров, использующихся в К-представлении,
* Таблица параметров, использующихся в онтологии
* Таблица связи между ними.

**Слайд 19**

Переходим непосредственно к преобразованию семантического представления в SPARQL-запрос.

Условно SPARQL-запрос можно поделить на несколько частей:

* Заголовок
* Тройки равенства
* Тройки сравнения

**Слайд 20**

В заголовке определяется искомая сущность, т.е. сущности какого типа необходимо найти (например, самолёт, планета и т.д.). Тип сущности определяется на основе понятия указанного в К-представлении (фрагмент A).

Эта часть SPARQL-запроса обязательная, заголовок есть у каждого запроса.

**Слайд 21**

Следующий возможная часть SPARQL-запроса – тройка равенства. Условно можно выделить три типа троек сравнения:

* Соответствие значению из онтологии;
* Соответствие числу (сюда же можно отнести и дату);
* Тройка со специальной константой МАКС или МИН

У всех трех типов можно увидеть общую часть. Это определение предиката. При этом определение значение необходимо только для тройки с соответствием значению из онтологии. Для числа очевидно данная процедура не требуется.

А вот интересна тройка с константой МАКС. Это константа означает, что необходимо найти сущность с максимальным значением указанного параметра. Для реализации поиска такой сущности проводится ряд действий:

Все сущности сортируются в порядке убывания значения параметра и из них выбирается первый. Для этого в конец запроса добавляются ключевые слова, указывающие необходимость сортировки, порядок сортировки, указывается переменная, предварительно связанная с необходимым значением, по которой проводится сортировка и указывается, что необходимо вернуть только первую сущность из отсортированного списка.

Для константы МИН алгоритм аналогичен, но сортировка проводится по возрастанию.

**Слайд 22**

Тройка сравнения определяет верхнюю или нижнюю границу (только одну) диапазона значений, которые может принимать указанный параметр. Для этого со значением связывается переменная, к которой применяется синтаксическая конструкция языка SPARQL «Фильтр».

**Слайд 23**

Для программной реализации использовалась платформа .NET (6ой версии). Она предоставляет инструменты и технологии, позволяющие уменьшить время программной реализации. В частности, такими средствами являются LINQ to Entity Framework и WPF, применяемые для взаимодействия с базой данных и реализации оконного интерфейса, соответственно.

В качестве СУБД использовалась PostgreSQL.

Также использовались библиотеки DeepMorphy для определения морфологических признаков и dotNetRdf для выполнения SPARQL-запросов.

**Слайд 24**

На слайде представлен интерфейс приложения. В интерфейсе есть поле для ввода входного запроса на русском языке, таблица для вывода результатов выполнения запроса и кнопка, по нажатию на которую, начинается преобразование входного запроса в SPARQL-запрос.

Результатом выполнения запроса, является список ссылок, на страницы с описанием искомых сущностей. По клику на строку таблицу соответствующая страница открывается в браузере по умолчанию.

**Слайд 25**

В ходе данной работы:

* Был разработан и реализован семантически-ориентированный естественно-языковой интерфейс для взаимодействия с Системой открытых взаимосвязанных данных

Для этого было:

* Проведено сравнение подходов к формальному описания семантической структуры текстов и выбран подход теории К-представлений В.А. Фомичева
* Разработана и реализована лингвистическая база данных
* Разработаны алгоритмы для реализации преобразования «ЕЯ-запрос → SPARQL-запрос»
* Предложен принцип преобразования параметров запроса, позволяющий преодолевать проблему неоднозначности имен в онтологии