МИНОБРНАУКИ РОССИИ
ФЕДЕРАЛЬНОЕ ГОСУДАРСТВЕННОЕ БЮДЖЕТНОЕ ОБРАЗОВАТЕЛЬНОЕ УЧРЕЖДЕНИЕ
ВЫСШЕГО ПРОФЕССИОНАЛЬНОГО ОБРАЗОВАНИЯ
«НОВОСИБИРСКИЙ НАЦИОНАЛЬНЫЙ ИССЛЕДОВАТЕЛЬСКИЙ ГОСУДАРСТВЕННЫЙ УНИВЕРСИТЕТ» (НОВОСИБИРСКИЙ ГОСУДАРСТВЕННЫЙ УНИВЕРСИТЕТ, НГУ)

Кафедра Систем Информатики Факультета информационных технологий
(название кафедры)

ВЫПУСКНАЯ КВАЛИФИКАЦИОННАЯ РАБОТА БАКАЛАВРА

……Селиванова Надежда Павловна……...
(фамилия, имя, отчество автора - студента –выпускника)

Методы и средства установления соответствия между онтологиями
(тема работы)

Направление подготовки 230100.62 ИНФОРМАТИКА И ВЫЧИСЛИТЕЛЬНАЯ ТЕХНИКА

Руководитель

Апанович З.В.
(фамилия , И., О.)
Кандидат физико-математических наук.
(уч.степень, уч.звание)

……..
(подпись, дата)

Автор

Селиванова Н.П.
(фамилия , И., О.)
ФИТ гр.0208.
(факультет, группа)

..……..
(подпись, дата)

Новосибирск, 2013г.

Содержание

Введение

Всемирная паутина Интернет стремительно входит буквально во все сферы нашей жизни. Средствам обработки данных в сети все сложнее и сложнее справляться с большим потоком информации, которая уже существует и, которая ежедневно добавляется в сеть. Кроме того, данные в Интернет организованы крайне стихийно и несистематично. Решением данной проблемы является новый этап развития Интернет – SemanticWeb, целью которого является представление информации в виде, пригодном для машинной обработки.

Это направление развивается с 2001 года. В связи с этим, в Интернете становятся доступными большие объемы информации, так же посвященные различным направлениям. В число таких ресурсов входят информационные системы, цифровые библиотеки и специализированные порталы, посвященные научным областям и основанные на онтологиях[1]. Важные научные направления, самые активные и влиятельные исследователи, организации, в которых они работают, и места, в которых расположены научные организации – вся эта информация становится доступной в rdf/xml формате. Она эволюционирует во времени и стремительно увеличивается в объеме. Исследование и анализ этих данных необходимы для оптимизации процессов управления научными исследованиями. Для обеспечения понимания этих стремительно расширяющихся данных нужны новые инструменты.

Одним из центральных понятий инженерии онтологий является понятие «отображение онтологий» (ontology mapping), под которым понимается деятельность по установлению соответствия между несколькими онтологиями или, другими словами, нахождение семантических связей подобных элементов из разных онтологий. С наиболее общей точки зрения важность задачи отображения онтологий обусловлена тем фактом, что мощность знаний, заключенных в онтологиях, проявляется в полной мере только в том случае, когда удается учесть взаимосвязи независимых онтологий - установление факта подобия сущностей в разных онтологиях означает извлечение из этих онтологий дополнительных знаний.

[1]Онтология (в информатике) – это описание некоторой предметной области с помощью объектов, их связей и правил.

Данная квалификационная работа посвящена исследованию и разрешению вопроса установления соответствия между двумя онтологиями.

Для решения описанной проблемы была проведена работа:

- Исследована предметная область Онтологий, язык запросов SPARQL;
- Проведен анализ существующих алгоритмов для установления соответствия между онтологиями;
- Изучены аналоги программных средств, предназначенных для решения проблем моей задачи;
- Разработан модуль для программы ”Визуализации онтологии и информационного наполнения семантических систем”. Этот модуль позволяет визуализацию одновременно двух онтологий при помощи поуровневого и радиального алгоритмов. А его функционал предназначен для установления соответствия между онтологиями в ручном режиме, с последующей отправкой SPARQL запроса к сервису для получения данных.

1. Определение онтологии

Онтология (в информатике) – это описание некоторой предметной области с помощью объектов, их связей и правил.

Онтология (в математике) – это тройка $O = (C, S, isa)$, где

1. $C = \{c_1, \ldots, c_m\}$ – множество классов, каждый из которых обозначает множество реальных объектов.
2. $S = \{s_1,\ldots, s_n\}$ – множество слотов, где каждый слот является либо свойством класса, принимающим значения типа String, Integer, Date, либо ролью, то есть соответствует отношению между классами.
3. $isa = \{isa_1,\ldots, isa_p\}$ – множество отношений наследования, определенных между классами.

Отношения наследования определяют частичный порядок между классами, организуя классы в одно или несколько деревьев. Для того чтобы это определение позволяло работать с информационным наполнением портала знаний, его необходимо дополнить элементом $I = \{i_1, .., i_q\}$, где i_{c_x} – это экземпляр некоторого класса c_x. Экземпляр класса c_x содержит конкретные значения для каждого слота, связанного с классом c_x или с его предками (в случае наличия множества isa).

Компоненты онтологий:

1. **Классы.**
2. **Индивиды.** Индивиды (Экземпляры) – это фактические данные, связанные с онтологией, и, в большинстве случаев, именно они интересуют пользователя. Однако, изображение экземпляров в виде вершин, соединенных ребрами с вершинами-классами часто неэффективно, потому что их может быть большое количество. Для визуализации объектов может быть использован такой способ, как представление экземпляров выбранного класса списком внутри отдельного окна.
3. **Таксономии (isa связи).** Визуализация таксономии, на которой основана онтология, является существенной для понимания отношений наследования между классами. Система визуализации должна обеспечивать, как минимум, целостное представление иерархической структуры таксономии. Весьма желательны также частичные изображения, позволяющие пользователю фокусироваться на определенном подмножестве таксономии. Для представления таксономии широко используются методы визуализации деревьев.
4. **Множественное наследование**. Визуализацию классов, имеющих более одного

родителя, достаточно сложно совместить с эффективным представлением таксономии. Для визуализации множественного наследования используются методы размещения графов, а не методы визуализации деревьев.

5. **Ассоциативные связи.** Еще одной сложной задачей является визуализация ассоциативных связей. Помимо ребра, изображающего ассоциативное отношение, весьма желательно размещение читаемой метки этого ребра. Множественное наследование и ролевые связи – это два типа связей, преобразующих онтологическую иерархию в граф. Графы визуализировать сложнее, чем деревья.
6. **Атрибуты.** Свойства, связанные с некоторой сущностью, также очень важны и должны быть полностью визуализированы.

   На сегодня под онтологией можно понимать:

   - надежный семантический базис в определении содержания;
   - общую логическую теорию, которая состоит из словаря и набора утверждений на некотором языке логики;
   - основу для коммуникации между людьми и компьютерными агентами.

Онтологии позволяют представить новые понятия так, что они становятся пригодными для машинной обработки. С помощью онтологии можно «перекинуть мостик» между новыми понятиями, с которыми система еще не встречалась, и описаниями уже известных классов, отношений, свойств и объектов реального мира.

На рисунке 1 представлены области применения, роли, типы, языки представления и владельцы онтологий.

Рисунок 1. Систематизация знаний в предметной области онтологий.

Предложенное на рисунке 1 описание детализирует систематизацию знаний об онтологиях в части областей применения онтологий, их роли и типа.

Построение онтологий – сложный и занимающий много времени процесс. Чтобы облегчить его, с середины 90-х годов начали создаваться первые среды для процесса разработки онтологий. Они обеспечили интерфейсы, которые позволили выполнять концептуализацию, реализацию, проверку непротиворечивости и документирование. За последние годы число инструментов работ с онтологиями резко возросло. (На сегодня сайт консорциума W3C, например, предоставляет список более чем 50 инструментов редактирования).

Онтологии могут храниться в различных форматах, таких как RDF, OWL и производные от них диалекты (RDFS, OWL DL, OWL Lite, OWL Full)

Одним из центральных понятий инженерии онтологий является понятие «отображение онтологий» (ontology mapping), под которым понимается деятельность по установлению соответствия между несколькими онтологиями или, другими словами, нахождение семантических связей подобных элементов из разных онтологий.

Близкой к проблеме отображения онтологий является проблема выравнивания онтологий (ontology alignment), которая заключается в том, чтобы установить различные виды соответствия между двумя онтологиями, а затем сохранить исходные онтологии вместе с информацией о найденных соответствиях с тем, чтобы в дальнейшем

использовать информацию о взаимосвязях онтологий. Выравнивание онтологий позволяет осуществлять трансляцию данных первой онтологии во вторую. Отметим также, что на основе отображения онтологий решается задача интеграции онтологий (ontology merging) – задача создания новой онтологии или ее фрагментов из двух и более исходных онтологий.

.

1.1. Модель RDF

RDF (Resource Description Framework) – это разработанный консорциумом W3C язык общего назначения для представления информации. RDF применяется для того, чтобы предоставить информацию о ресурсах в терминах простых свойств и значений этих свойств. Ресурсом может выступать любой элемент, имеющий некоторый URI (Uniform Resource Identifiers), при этом не обязательно, чтобы он был доступен через Интернет. Идея описания ресурсов в терминах простых свойств, их значений и применения URI для идентификации ресурсов дает возможность представить информацию о ресурсах в RDF, как граф узлов и дуг, представляющих ресурсы, их свойства и значения этих свойств [6] .

Идея состоит в том, чтобы одним простым способом можно было бы описать любой факт, притом в таком структурированном виде, чтобы его могли обрабатывать компьютерные программы.

Форматы файлов, реализующие абстрактную модель RDF:

1. Префиксные форматы записи утверждений;
2. RDF/XML – запись в виде XML-документа;
3. RDF/JSON – запись в виде JSON;
4. Запись в виде HTML-или XHTML- документа.

1.2. Модель OWL

OWL — (англ. Web Ontology Language) — язык онтологии для интернета на основе XML/Web стандарта. Язык веб-онтологий OWL предназначен для описания классов и отношений между ними, которые присущи веб-документам и приложениям. В основе языка - представление действительности в модели данных «объект-свойство». Язык OWL пригоден не только для описания Web страниц, но и любых объектов действительности.

Основные элементами онтологии являются простые и сложные классы, свойства классов, индивиды и свойства индивидов.

1.3. Язык запросов SPARQL

SPARQL (рекурсивный акроним от англ. SPARQL Protocol and RDF Query Language) — язык запросов к данным, представленным в модели RDF, а также протокол для передачи этих запросов и ответов на них. SPARQL является рекомендацией консорциума Всемирной паутины (W3C)[8]и был создан как язык запросов для Semantic Web.
Данный язык позволяет:

1. Извлекать определенные значения из структурированных данных в формате RDF.
2. Исследовать отношения между данными
3. Выполнять соединения данных из нескольких источников данных в одном запросе.
4. Преобразовывать RDF данные из одного словаря в другой

Рекомендация консорциума по языку запросов SPARQL накладывает определенные ограничения на формат. Поддерживается только RDF формат, представленный в виде графа, составленного из триплетов, - субъект, предикат, объект (рис. 2). Субъект (или Ресурс) должен быть представлен с определенным идентификатором URI (Uniform Resource Identifiers), который может быть сокращен, используя префиксы. Объект может быть литералом: строкой, числом либо принимать значения true или false.

Рисунок 2. Триплет.

1.3.1. Структура SPARQL запроса

Ниже приводится описание структуры SPARQL запроса [9]. Схема запроса отражена на рисунке 3.

1. Префиксные объявления служат для сокращения универсальных идентификаторов ресурса.
2. Источники запроса определяют, какие RDF-графы запрашиваются.

3. Пункт результата возвращает набор данных (выборку в случае оператора SELECT), удовлетворяющих заданному условию.
4. Критерии запроса определяют, что запросить в базовом наборе данных.
5. Модификаторы запроса ограничивают и упорядочивают результаты запроса.

```
PREFIX foo: <http://example.com/resources/>
# префиксные объявления
FROM ...
# источники запроса
SELECT ...
# состав результата
WHERE {...}
# шаблон запроса
ORDER BY ...
# модификаторы запроса
```

Рисунок 3. Общая схема SPARQL-запроса.

2. Постановка задачи

Необходимо реализовать возможность установления соответствия между двумя онтологиями. В качестве тестовых данных использовались онтология ОНС Открытого архива СО РАН [21] и AKT Reference Ontology[18], на которой основано большое количество научны знаний. С последующим получением данных в объединённом формате.

Особенность онтологии ОНС состоит в том, что многие сущности, описанные в других онтологиях при помощи отношений, в данной онтологии представлены в виде экземпляров классов. Эта специфика компенсирует отсутствие атрибутов у RDF предикатов. Например, такие классы как «dating», «naming», «authorship» из ОНС заменяют предикаты «has-author», «has-date», «has-name». Таким образом, одна или несколько групп вида (класс1-отношение1-класс2) из AKT Reference Ontology онтологии соответствует одной или нескольких групп вида (класс3-отношение2-класс4-отношение3-класс5) из ОНС онтологии. Поэтому разработанное средство должно иметь возможность устанавливать соответствие между группами классов и отношений.[1]

После анализа результатов работы программных средств, предназначенных для установления отображения между онтологиями, было выявлено, что для установления соответствия с высоким уровнем точности необходимо разработать алгоритм, учитывающий специфику онтологии.

Требовалось разработать модуль для программы «Визуализации онтологии и информационного наполнения семантических систем» (ВИНСС), позволяющий:

1. Визуализировать две онтологии;
2. Находить подобные сущности;
3. Анализировать и устанавливать соответствия в интерактивном режиме;
4. Формировать автоматический SPARQL запрос, для выгрузки данных полученных результатов.

2.1. Входные данные

Входными данными являются две онтологии, а так же некоторые настройки для удобной визуализации этих онтологий.

В качестве тестового примера использовались:

1. «AKT Reference Ontology» – представляет знания о людях, проектах, публикациях, географических данных [19]. Была разработана для представления знаний CS AKTive портала, описана в формате OWL. Фрагмент данной онтологии представлен на рисунке 4.

```
<owl:ObjectProperty rdf:ID="owned-by">
  <rdfs:domain>
    <owl:Class>
      <owl:unionOf rdf:parseType="Collection">
    <owl:Class rdf:about="#Information-Bearing-Object"/>
    <owl:Class rdf:about="#Technology"/>
    <owl:Class rdf:about="#Method"/>
  </owl:unionOf>
    </owl:Class>
  </rdfs:domain>
  <rdfs:range rdf:resource="#Legal-Agent"/>
  <rdfs:isDefinedBy rdf:resource="&base;"/>
</owl:ObjectProperty>

<owl:Class rdf:ID="Abstract-Information">
  <rdfs:comment>Information in general, independent of an object
  <rdfs:subClassOf rdf:resource="&support;Intangible-Thing"/>
  <rdfs:isDefinedBy rdf:resource="&base;"/>
</owl:Class>
```

Рисунок 4. Фрагмент описания AKT Reference Ontology онтологии.

2. «Онтология неспецифических сущностей» (ОНС) – открытый архив Сибирского отделения Российской Академии Наук. Эта онтология так же описана в формате OWL(Рис. 5).

```
<owl:Class rdf:ID="sys-obj">
  <rdfs:subClassOf rdf:resource="#Thing" />
</owl:Class>
<owl:DatatypeProperty rdf:ID="name">
  <rdfs:label xml:lang="ru">имя</rdfs:label>
  <rdfs:label xml:lang="en">name</rdfs:label>
  <rdfs:domain rdf:resource="#sys-obj" />
  <rdfs:range rdf:resource="#text" />
</owl:DatatypeProperty>
<owl:DatatypeProperty rdf:ID="description">
  <rdfs:label xml:lang="ru">описание</rdfs:label>
  <rdfs:label xml:lang="en">description</rdfs:label>
  <rdfs:domain rdf:resource="#sys-obj" />
  <rdfs:range rdf:resource="#text" />
</owl:DatatypeProperty>
<owl:Class rdf:ID="person">
  <rdfs:label xml:lang="ru">Персона</rdfs:label>
  <rdfs:label xml:lang="en">Person</rdfs:label>
  <rdfs:subClassOf rdf:resource="#sys-obj" />
</owl:Class>
```

Рисунок 5.Фрагмент описания онтологии ОНС.

2.2. Выходные данные

После первого этапа “Поиска подобных сущностей”, пользователь получает список подобных классов второй онтологии выбранных алгоритмом. Например, для класса “city” ОНС онтологии программа выдает результат, показанный на рисунке 6.

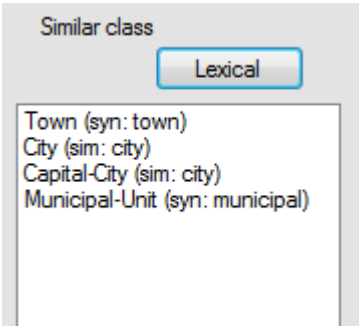

Рисунок 6.Результат работы алгоритма “Поиска подобных сущностей”

Результатом второго этапа ручного установления соответствия между онтологиями является Лог выбранных вершин и классов представленный на рисунке 7.

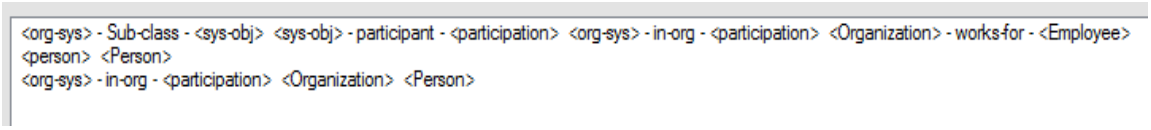

Рисунок 7.Результат ручного выбора соответствующих групп классов и отношений

Итоговые выходные данные это результаты SPARQL запросов объединённых данных двух онтологий.

3. Программные средства, предназначенные для манипуляциями с онтологиями

Инструментальные средства, которые помогают находить соответствия между онтологиями, могут использоваться:

- для объединения двух онтологий с целью создания одной новой (PROMPT [10], Chimaera [11], OntoMerge [12]);
- для определения функции преобразования из одной онтологии в другую (OntoMorph [13]);
- для определения отображения между концептами в двух онтологиях, находя пары связанных концептов (например, Agreement Maker [17], OBSERVER[14], FCA-Merge [15]);
- для определения правил отображения для связи только релевантных частей исходных онтологий (ONION [16]).

Рассмотрим теперь вышеупомянутые средства более подробно.

PROMPT - дополнение к системе Protege, реализованное в виде плагина, служит для объединения и группировки онтологий. При объединении двух отологий PROMPT создает список предлагаемых операций. Операция может состоять, например, из объединения двух терминов или копирования терминов в новую онтологию. Пользователь может выполнить операцию, выбирая одну из предлагаемых или определяя непосредственно операцию. PROMPT выполняет выбранную операцию и дополнительные изменения, вызванные этой операцией. Потом список предлагаемых операций модифицируется и создается список конфликтов и возможных решений этих конфликтов. Это повторяется до тех пор, пока не будет готова новая онтология.

OntoMorph определяет набор операторов преобразования, которые можно применить к онтологии. Затем человек-эксперт использует начальный список пар и исходных онтологий для определения набора операторов, которые должны примениться к исходным онтологиям для устранения различий между ними, и OntoMorph применяет эти операторы. Таким образом, совокупность операций может выполняться за один шаг.

Однако, человек-эксперт не получает никакого руководства за исключением начального списка пар.

FCA-Merge - метод для сравнения онтологий, которые имеют набор общих экземпляров или набор общих документов, аннотируемых с помощью концептов исходных онтологий. Основываясь на этой информации, FCA-Merge использует математические методы из Formal Concept Analysis для того чтобы произвести решетку концептов, связывающую концепты исходных онтологий. Алгоритм предлагает отношения эквивалентности и подкласс-суперкласс. Затем инженер онтологии может анализировать результат и использовать его как руководство для создания объединенной онтологии. Однако предположение, что две объединяемые онтологии используют общий набор экземпляров или имеют набор документов, в котором каждый документ аннотируется терминами обоих источников слишком жесткое и на практике такая ситуация происходит редко. В качестве альтернативы, авторы предлагают использовать методы обработки естественного языка для аннотации набора документов концептами из этих двух онтологий.

Система ONION основана на алгебре онтологии. Поэтому, она предоставляет инструменты для определения правил артикуляции (соединения) между онтологиями. Правила артикуляции обычно учитывают только релевантные части исходных онтологий. Для того чтобы предложить соединение, ONION использует и лексические методы, и методы на основе графов. Метод нахождения лексического подобия между именами концептов использует словари и методы семантической индексации, основанные на местонахождении группы слов в тексте.

Система Agreement Maker - это расширяемая структура, для построения соответствия между онтологиями. Данная система содержит множество различных алгоритмов, предназначенных для определения отображения. Так же она позволяет эффективно объединить результаты сразу нескольких алгоритмов вместе.

3.1. Сравнительный анализ программных средств

Разнообразие инструментов установления соответствия между онтологиями делает сложным их непосредственное сравнение. Фактически, когда разработчик должен решить вопрос, какой инструмент является наиболее подходящим, все будет зависеть от конкретной задачи. Например, если объединяемые онтологии совместно используют набор экземпляров, то лучше всех может работать FCA-Merge. Если онтологии имеют

экземпляры, но совместно их не используют, и многие значения слотов содержат текст, лучшим выбором может стать GLUE. Если только части онтологий должны быть отображены, можно было бы выбрать инструмент ONION. Если онтологии имеют очень ограниченную структуру, а концепты имеют подробные определения на естественном языке (одном), инструментальные средства ISI/USC могут обеспечивать лучшие ответы. Если экземпляры вообще не доступны, и онтологии содержат много отношений между концептами, лучше всех может работать Prompt .

Исходя из поставленной задачи и онтологий, предназначенных для анализа, самый подходящий инструмент Agreement Maker. Так же у данного программного продукта есть преимущества:

- Высокая продуктивность работы (точность найденных отображений);
- Большой выбор алгоритмов;
- Удобство использования;
- Возможность накладывать и объединять результаты, полученные различными алгоритмами.

3.2. Анализ результатов полученных Agreement Maker

Проанализировав результаты работы различных алгоритмов Agreement Maker на онтологиях AKT и ONC, я получила маленький процент найденных соответствий между классами и отношениями, это связанно с особенностью ONC онтологии. Внешний вид программы с результатами работы алгоритмов для онтологий ONC и AKT показан на рисунке 6. Из чего были сделаны выводы о том, что стандартные алгоритмы, предназначенные для отображения онтологий, не подходят для использования на предложенных мне онтологиях. Необходимо учитывать специфику построения ONC онтологии. Данная система не позволяет устанавливать соответствие для групп классов и отношений, что необходимо при рассмотрении тестовых онтологий.

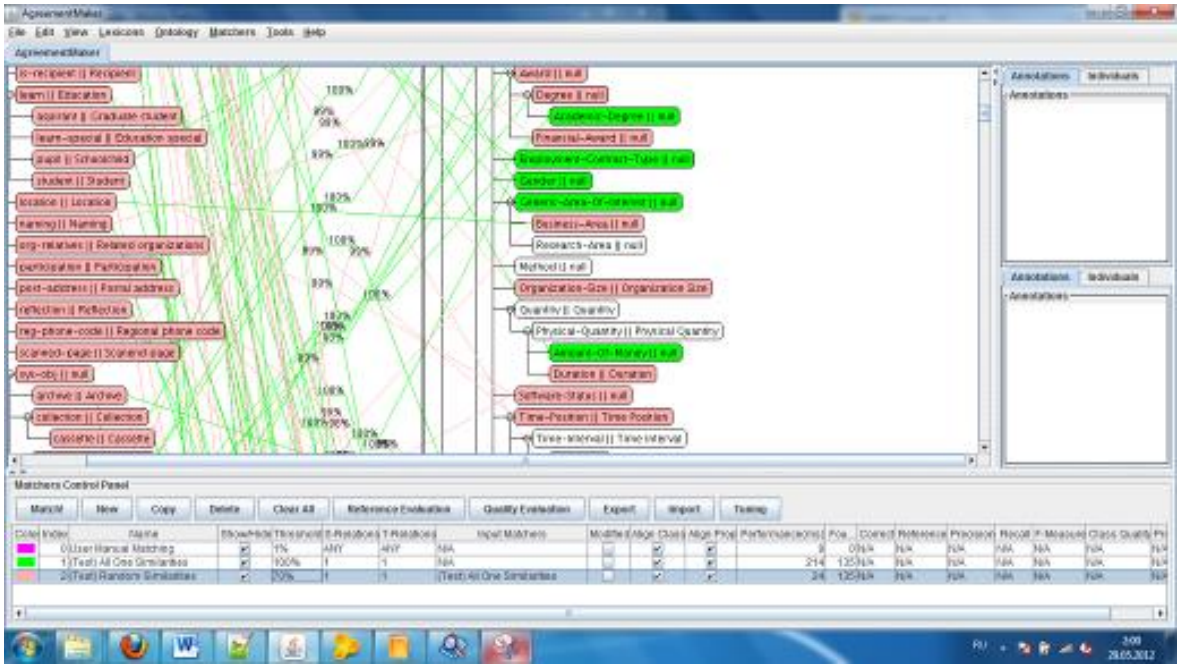

Рисунок 8. Результат работы программы.

На рисунке 8 представлены результаты работы приложения Agreement Maker на онтологиях ONC (справа) и AKT Reference Ontology (слева). Найдены соответствия некоторых классов, таких как ONC:location – AKT:Location, но при этом довольно много некорректных соответствий таких как ONC:student – AKT:Software-status или ONC:naming – AKT:Degree

Решением возникших проблем является алгоритм лексического анализа и инструмент для установления соответствия в интерактивном режиме.

4. Существующие методы установления соответствия между двумя онтологиями

В этом разделе проведен обзор различных алгоритмов предназначенных для решений задачи отображение онтологий.[2]

Проблема отображения онтологий заключается в том, что:

1. Сущности (классы, свойства, связи, объекты), имеющие одинаковые имена могут иметь разный смысл.
2. Сущности (классы, свойства, связи, объекты), имеющие одинаковый смысл, могут иметь разные имена.

Современные методы установления отображения онтологий носят междисциплинарный характер. Среди них можно выделить 4 группы: лингвистические (терминологические, лексические), статистические (экстенсиональные), структурные и логические (формальные, семантические) методы. Рассмотрим каждый из них отдельно.

4.1. Лингвистический/Лексический/Текстовый анализ

На данном этапе определяется сходство между сущностями на основе сравнения имен сущностей (оценка количества совпадающих символов, общие части слов, например, «Цели» и «Целевые установки») или путем анализа синонимичных терминов. Для выявления синонимичных терминов могут использоваться существующие словари общей и профессиональной лексики, тезаурусы. Данный вид анализа можно считать исходным для установления соответствия между сущностями.

4.2. Структурный анализ

Анализ внутренней структуры

В данном случае оценка сходства производится на основе анализа доменов и областей допустимых значений для атрибутов и связей. Методы анализа внутренней структуры иногда называются методами на основе ограничений. Сущностей со схожей внутренней структурой, а также свойств с похожими доменом и областью значений может

быть достаточно много, поэтому данные методы используются только для формирования кластеров сходных понятий и требуют сочетания с другими методами.

Анализ сходства по иерархическим связям

Оценка схожести двух сущностей двух онтологий может быть основана на позициях данных сущностей в иерархии классов. Если две сущности двух онтологий схожи, то их «соседи» также как-то схожи. Такое утверждение может использоваться по-разному и порождает ряд возможных критериев (признаков) для сходства двух сущностей:

1. Их прямые супер-сущности (или все супер-сущности) уже являются схожими;
2. Их сущности-братья (или все их сущности-братья) уже являются схожими;
3. Их прямые сущности-потомки (или все их сущности-потомки) уже являются схожими;
4. Все их сущности-листья (сущности, не имеющие потомков, находящиеся в дереве, корнем которой является рассматриваемая сущность) уже являются схожими;
5. Все (или большинство) сущности на пути от корня к рассматриваемой сущности уже являются схожими.

Конечно, использование данных критериев должно сопровождаться другими критериями.

Анализ сходства по перекрестным связям

Определение сходства между сущностями может быть основано также на анализе связей сущностей. Если класс *A1* связан с классом *B1* связью типа R1 в одной онтологии, а класс *A2* связан с *B2* связью типа *R2* в другой онтологии, и если известно, что *B1* и *B2* – схожи, *R1* и *R2* – схожи, можно предположить схожесть *A1* и *A2*. Подобным образом можно говорить и сходстве типов связей – *R1* и *R2* если известно, что *A1* и *A2* – схожи, *B1* и *B2* – схожи. Таким образом, оценивается схожесть элементов онтологии.

Например, классы «Компания» и «Университет» будут оценены как схожие, поскольку они имеют схожую связь типа «имеет подчиненного» с классом «Сотрудник» и классом «Профессор», которые были признаны схожими.

4.3. Экстенсиональный (статистический) анализ

Для оценки экстенсионального соответствия классов используются существующие экземпляры классов. Для установки соответствия между сущностями используются следующие диагностические правила:

1. Если сущность *C1* эквивалентна сущности *C2,* тогда невозможно найти объект *O1:C1,* такой что не *O1:C2* и наоборот.
2. Если сущность *C1* является подклассом *C2,* тога невозможно найти объект *O1:C1,* такой что *O1:C2* и при этом *C1* не эквивалентен *C2.*

Анализ экстенсионала позволяет также идентифицировать классы-роли, когда возникает два разных класса для описания одного экстенсионала.

4.4. Логический анализ

Логический анализ основан на выявлении родовых классов сопоставляемых классов и анализе наложенных на них ограничений.

Например, в одной онтологии может существовать класс «Микро-компания», который является видовым классом для класса «Компания» с наложенным ограничением на «Число сотрудников» <5.

В другой онтологии может существовать класс «Малое предприятие», который является видовым классом для класса «Фирма» с наложенным ограничением на «Число работников» <10. При анализе соответствия между классами «Микро-компания» и «Малое предприятие» выявляются родовые классы «Компания» и «Фирма». При наличии информации о соответствие данных классов производится сравнение ограничений наложенных на данные родовые классы. Для этого сравниваются свойства классов «Число сотрудников» и «Число работников», если данные свойства схожи, то проводится сравнение наложенных ограничений <5 (сотрудников) и <10 (работников). В результате делается заключение, что «Микро-фирма»⊂«Малое предприятие».

После получения локальных соответствий между сущностями определяется глобальное соответствие между сущностями.

Практические рекомендации по расстановке приоритетов между результатами различных способов локального анализа:

При наличие баз знаний, включающих в себя экземпляры отображаемых онтологий, приоритетное значение имеют результаты экстенсионального анализа.

Однако результаты любого анализа следует согласовывать с результатами, полученными с использованием других видов анализа. Особенно важно такое согласование при установке соответствия между классами ролями, исполнители которых (экстенсионал) могут выполнять одновременно несколько ролей.

Поскольку предлагаемая методика ориентирована на «ручную» интеграцию онтологий путем поиска компромисса и согласования мнений, традиционные метрики сходства сущностей отображаемых онтологий не рассчитываются.

4.5. Результат анализа существующих методов

Проведя исследование, методов установления отображения онтологий был выбран лексический анализ. Структурные алгоритмы не подходят в связи со спецификой онтологии. Логический анализ подразумевает существование ограничений, которых в выбранных форматах реализации онтологий нет. Таким образом, самым продуктивным является создание лексического алгоритма.

5. Этапы разработки модуля для отображения онтологий

Разработку модуля удобнее было разбить на четыре этапа:

1. Адаптация программы ”Визуализации онтологии и информационного наполнения семантических систем” для внедрения модуля.
2. Разработка алгоритма лексического анализа.
3. Создание инструментов для установления соответствия между онтологиями.
4. Получение данных по построенным соотношениям. Решением данной задачи является разработка SPARQL-запроса с автоматической подстановкой классов и отношений, и сохранением полученных данных.

Итоговый вид программы после реализации модуля представлен на рисунке 9.

Рисунок 9. Основной вид программного продукта с модулем, предназначенным для построения отображения онтологий.

5.1. Адаптация программного обеспечения для внедрения модуля

Для написания модуля по отображению онтологий необходимо было выбрать инструмент визуализации онтологий.

Программа “Визуализации онтологии и информационного наполнения семантических систем” (ВОИНСС) предназначена для визуализации онтологий различными методами. Имеет удобный пользовательский интерфейс для просмотра и поиска классов и отношений между ними. Язык реализации C#. Для учебных целей мне был предоставлен исходный код данного продукта.

Изначальный вид программы ВОИНСС представлен на рисунке 10.

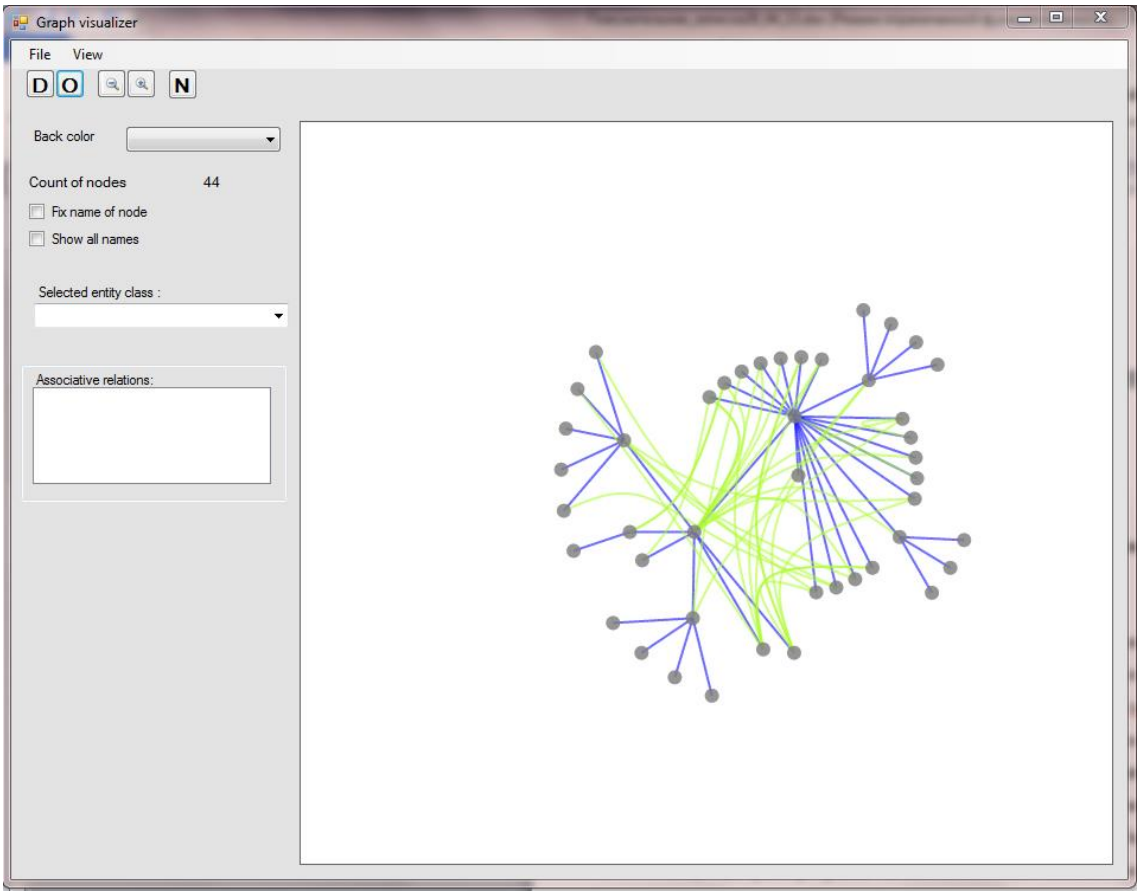

Рисунок 10. Визуализации онтологии и информационного наполнения семантических систем.

Для решения задачи установления соответствия, прежде всего, было необходима одновременная визуализация двух онтологий. Изменения, связанные с этим коснулись окна ввода данных. Для каждой онтологии выделена своя вкладка (рис. 11).

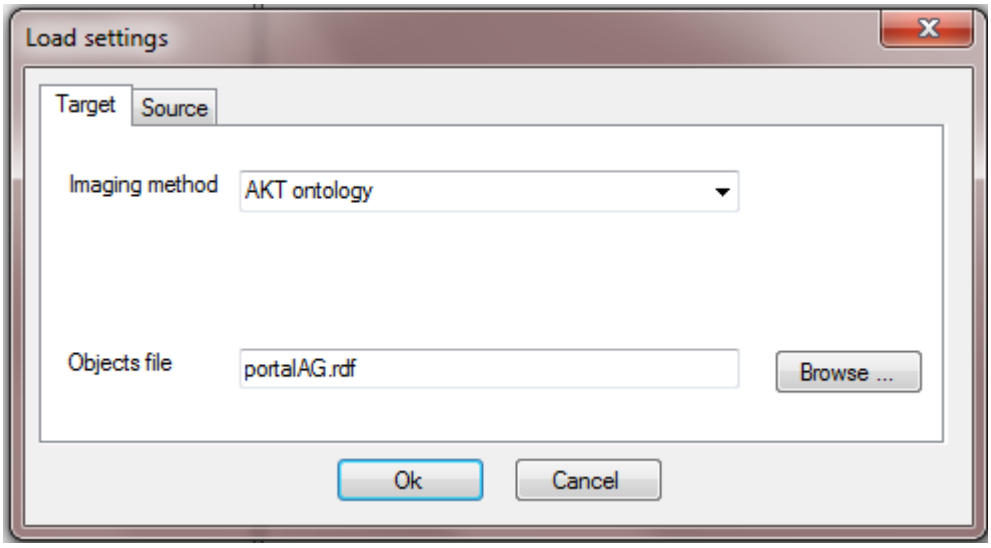

Рисунок 11. Загрузка данных.

Настройка параметров отображения не изменилась, так как соответствующие друг другу цвета и размеры классов и отношений удобны в восприятии.

Основное окно вывода онтологий поделили на две части. Эти области изолированы друг от друга, поэтому есть возможность перемещать и просматривать участки онтологий независимо. Зеркально функционалу для работы с онтологией добавлен аналогичный функционал для второй онтологии. Данный инструментарий представляет собой «Список Классов» и «Список Ассоциативных» связей, который позволяет просматривать и визуализировать список связей выбранного класса. На рисунке 12 в красной рамке расположен инструментарий и показана его работа.

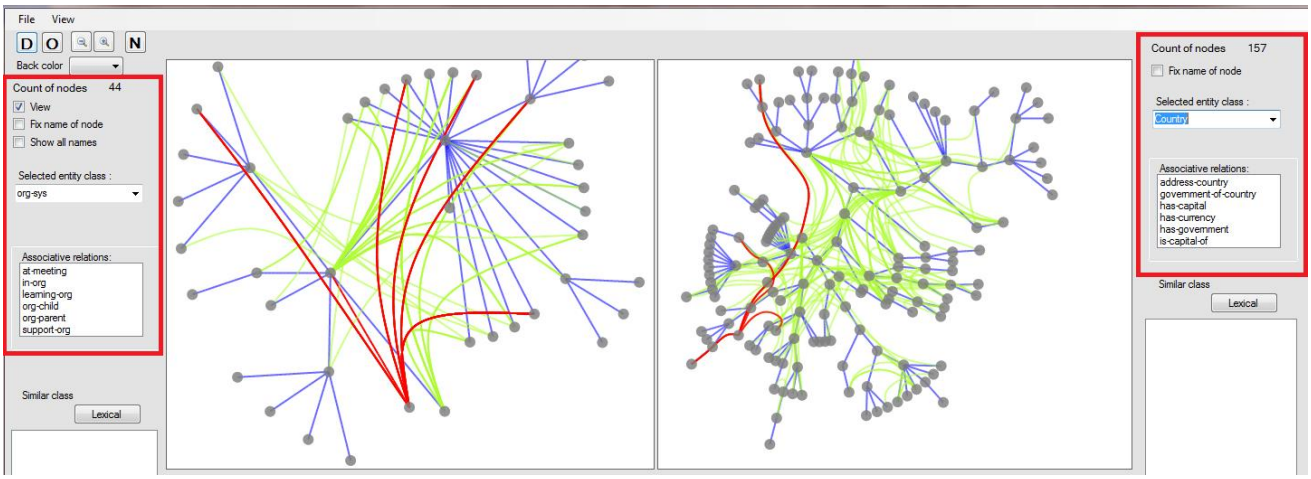

Рисунок 12. Визуализация двух онтологий ВОИНСС.

5.2. Разработка алгоритма лексического анализа

Данный алгоритм позволяет находить подобные классы из второй онтологии для выбранного пользователем класса. Он предназначен подготовки к последующему ручному построению соответствий между группами классов и отношений.

На рисунке 13 в красных областях располагается инструментарий для нахождения подобных классов. В него входит кнопка запуска работы алгоритма для класса, выбранного в «Списке Классов», а так же список классов из второй онтологии, подобных выбранному. В списке «Подобных Классов» есть уточнение, по какому принципу данный класс был отфильтрован алгоритмом. Если выбрать класс из списка «Подобных Классов», он становиться активным в инструментарии второй онтологии и визуализируется на графе. Для каждой онтологии есть свой инструментарий. На рисунке 13 показана работа алгоритма для класса city первой онтологии и класса Capital –City второй онтологии.

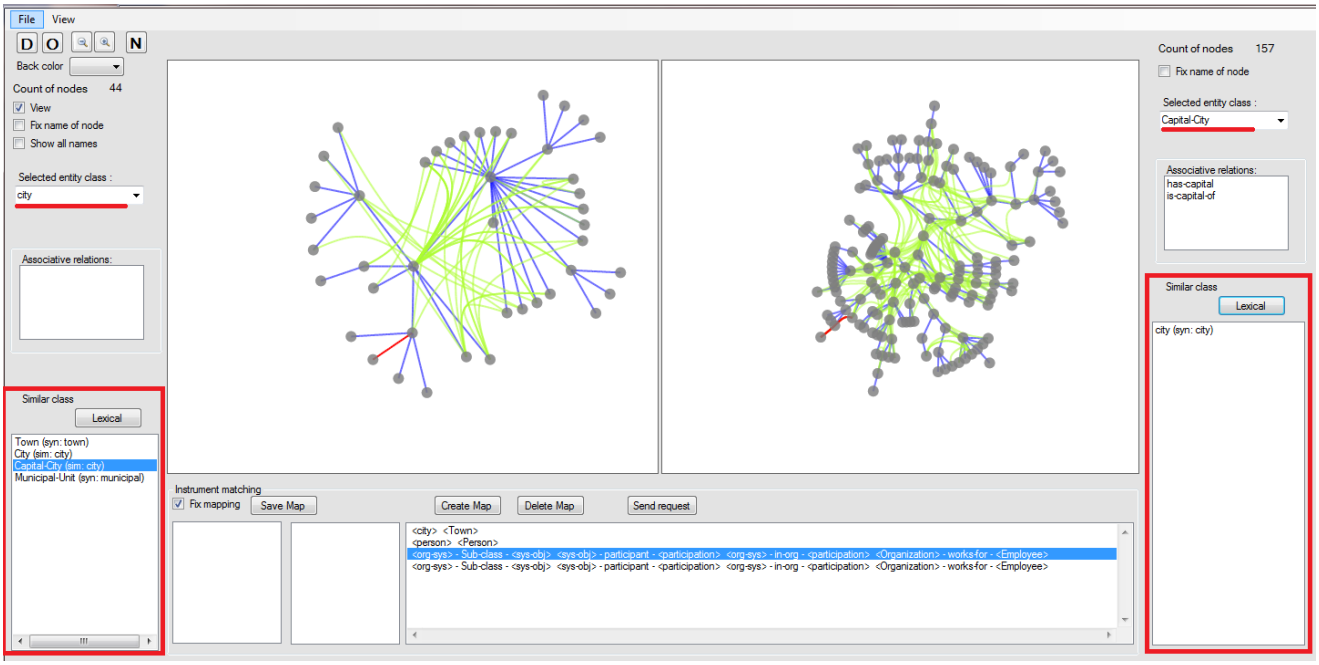

Рисунок 13. Инструментарий для поиска подобных классов.

5.2.1. Алгоритм поиска подобных классов

Ранее были исследованы существующие методы установления соответствия. По результатам их анализа необходимо было разработать алгоритм лексического анализа Классов онтологий, на основе критерия подобия идентификаторов. Критерий формулируется следующим образом: если метки двух сущностей подобны, то эти сущности подобны.

Для реализации данного алгоритма потребовался тезаурус синонимов. Для этого был найден web сервис, который выдает список синонимов по заданному значению. Программа обращается к сервису, передает искомое слово, а затем производиться грамматический разбор полученной HTML страницы. В результате мы получаем список синонимов.

Описание алгоритма:

Входные данные: Исходный класс первой онтологии, Список классов второй онтологии

Выходные данные: Список подобных классов

Данный алгоритм состоит из шагов:

1. Разбиваем исходный класс на подслова относительно разделителя «-»;

2. Запускаем алгоритм относительно каждого подслова кроме предлогов;

2.1. Выбираем список синонимов для подслова;

2.2. Перебираем все синонимы, в том числе под слово *synonym;*

2.2.1. Для каждого класса из списка классов второй онтологии делаем:

2.2.1.1. Разбиваем класс второй онтологии на подслова относительно разделителя «-»;

2.2.1.2. Для каждого подслова класса второй онтологии *sub_second*, кроме предлогов проводим сравнение;

2.2.1.2.1. Если расстояние Левенштейна между *sub_second* и *synonym* меньше, чем целая часть значения минимальной длины из этих слов, умноженная на 0.1. Тогда добавляем класс второй онтологии в «Список подобных классов», если его еще там нет;

2.2.1.2.2. Если *sub_second* является подсловом слова *synonym,* или наоборот, тогда добавляем класс второй онтологии в «Список подобных классов», если его еще там нет;

2.2.1.2.3. Если *sub_second* в точности является *synonym.* Тогда добавляем класс второй онтологии в «Список подобных классов», если его еще там нет;

3. Если не найдено ни одного совпадения программа выводит сообщение «No matches».

Данный алгоритм регистро-независимый, учитывает грамматические ошибки, сокращение слов, находит синонимичные термины и учитывает составные классы.

5.2.2. Расстояние Левенштейна

Расстояние Левенштейна (также редакционное расстояние или дистанция редактирования) между двумя строками — это минимальное количество операций вставки одного символа, удаления одного символа и замены одного символа на другой, необходимых для превращения одной строки в другую.

Пусть S_1 и S_2 — две строки (длиной M и N соответственно) над некоторым алфавитом, тогда редакционное расстояние (расстояние Левенштейна) $\mathrm{d}(S_1, S_2)$ можно подсчитать по следующей рекуррентной формуле $\mathrm{d}(S_1, S_2) = D(M, N)$, где

$$D(i,j) = \begin{cases} 0 & ;\ i = 0,\ j = 0 \\ i & ;\ j = 0,\ i > 0 \\ j & ;\ i = 0,\ j > 0 \\ \min(& \\ \quad D(i, j-1) + 1, & \\ \quad D(i-1, j) + 1, & ;\ j > 0,\ i > 0 \\ \quad D(i-1, j-1) + \mathrm{m}(S_1[i], S_2[j]) & \\ ) & \end{cases},$$

Где $\mathrm{m}(a, b)$ равна нулю, если $a = b$ и единице в противном случае; $\min(a, b, c)$ возвращает наименьший из аргументов.

Здесь шаг по i символизирует удаление (*D*) из первой строки, по j — вставку (*I*) в первую строку, а шаг по обоим индексам символизирует замену символа (*R*) или отсутствие изменений (*M*).

5.3. Создание инструментов для установления соответствия между онтологиями

Инструменты установления соответствия помещены в нижнюю часть программы. Для их активации необходимо кликнуть по “Fix matching” (рис. 14).

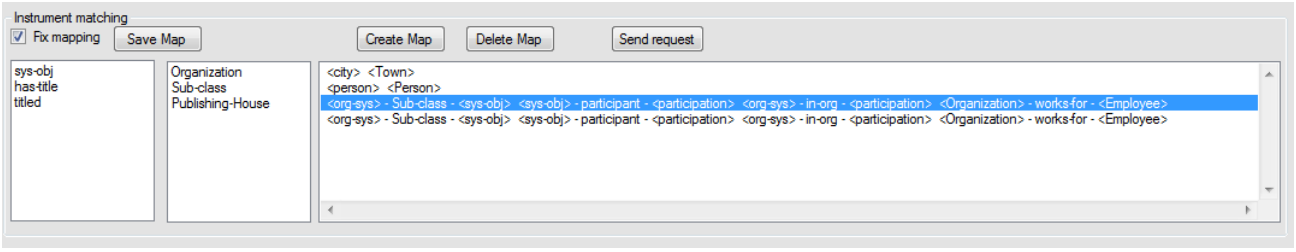

Рисунок 14. Инструменты для ручного установления соответствия.

Данный инструмент позволяет выделять классы, для этого достаточно кликнуть по необходимой вершине в любом Окне визуализации онтологии. При этом произойдет добавление Класса в “Список классов и отношений”, а так же выделение этой вершины другим цветом, как видно на рисунке 15. Если кликнуть по уже выбранной вершине выделение снимается.

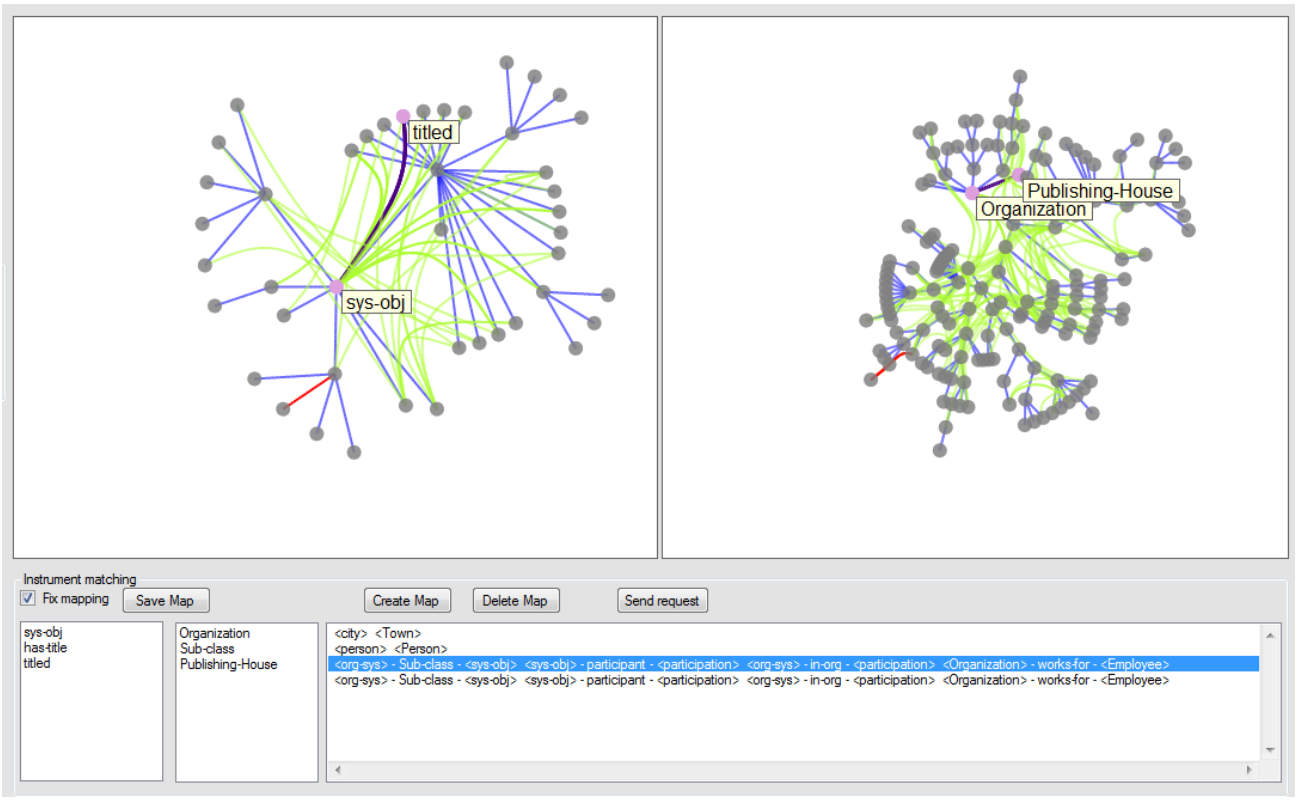

Рисунок 15. Основное окно работы с инструментом ручного установления соответствия.

При добавлении второй и более вершины программа запускает поиск отношений между всеми добавленными вершинами и так же их выделяет другим цветом и вносит их в “Список классов и отношений”. Это видно на рисунке 15

После выбора всех сопоставляемых друг другу вершин в обеих онтологиях. Пользователь может нажать “Save Map”, что приведет к сохранению в “Log” построенного соответствия. Результаты работы на рисунке 16.

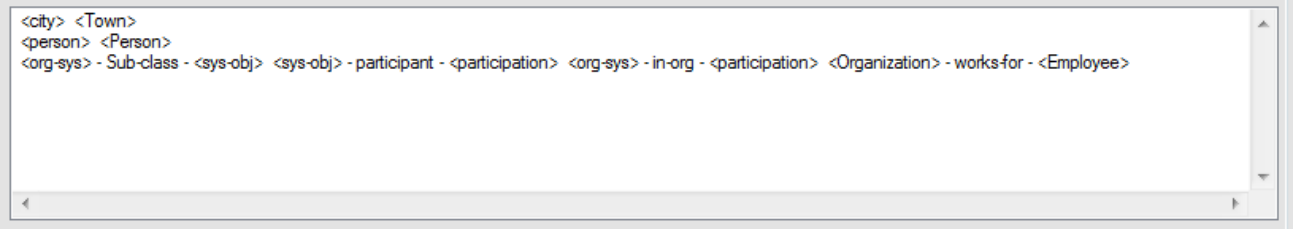

Рисунок 16. Лог построенных соответствий.

Пользователь может создавать новые соответствия при помощи кнопки «Create Map». Удалять из лога уже построенные соответствия - «Delete Map». А так же изменять, для этого необходимо в логе выбрать соответствие, которое собираемся редактировать, оно появиться в “Список классов и отношений” а так же отобразиться в окне визуализации.

5.4. Получение данных по построенным соотношениям

Получение данных в необходимом формате осуществляется при помощи SPARQL запроса.

Программа позволяет генерировать запросы для различных структур автоматически.

Шаблон SPARQL запроса в сервис для рассмотренных онтологий будет иметь такой вид:

```
PREFIX : <demo#>
PREFIX akt: <http://www.aktors.org/ontology/portal#>
PREFIX akts: <http://www.aktors.org/ontology/support#>
PREFIX rdfs: <http://www.w3.org/2000/01/rdf-schema#>
CONSTRUCT {
?p a :Class4.
?p rdfs:label ?Label.
?p :relation2?instance1.
?p :relation3 ?instance2.
}
WHERE {
        ?instance1 akt:relation1 ?instance2.
        ?instance1 a akt:Class1.
        ?instance2 a akt:Class2.
        ?instance1 akt:label ?instance1_label.
       ?instance2 akts:label ?instance2_label.
       BIND(Concat( str(?instance1_label), str (?instance2_label2)) As ?Label)
}
```

Где *«Class 1-relation1- Class2»* структура из АКТ онтологии, а «Class3- relation2-Class4-relation3-Class5» структура из ОНС онтологии.

Запросы при помощи «Окна построения запросов» можно редактировать, а так же задавать имя файла, в который будут записаны результаты. И редактировать точку доступа. По умолчанию стоит - demo.openlinksw.com [20] точка доступа. На ней проводилось тестирование для онтологий ОНС и АКТ.

6. Руководство по эксплуатации программного средства

Для работы с приложением необходимо открыть онтологии, по которым будем в дальнейшем строить соответствия. Возьмем онтологию AKT Reference Ontology (слева) и OHC (справа).

Найдем нужный класс OHC:org-sys, для этого выберем класс в «Списке Классов». «Список отношений» показывает связи которые идут от выбранной вершины. Выберем из «Списка отношений» связь OHC:in-org, как видно она связывает сущности OHC:org-sys и OHC:participation. Для того чтобы найти подобные классы для OHC:org-sys, запустим алгоритм лексического поиска. Это осуществляется при помощи кнопки «Lexical». На рисунке 17 виден результат поиска. Из этого длинного списка нам явно больше всего подходит AKT:Organization. Кликнув по нему в списке, мы делаем его активным классом в инструменте для правой онтологии (Рис. 17). Аналогично OHC онтологии, рассматриваем связи AKT онтологии, при помощи инструментов справа.

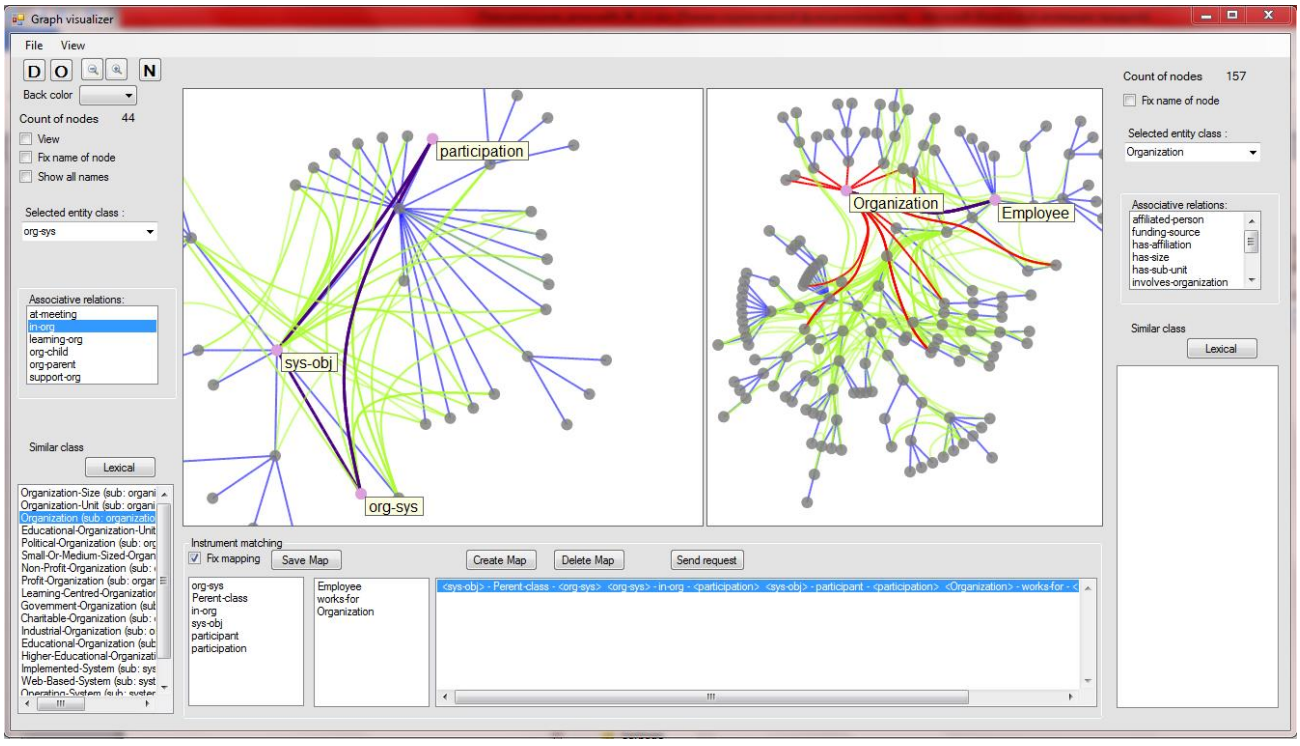

Рисунок 17. Приложение ВОИНСС с модулем по установлению соответствия в работе.

Для интерактивного установления соответствия делаем активной галочку «Fix mapping». Далее выбираем нужные нам вершины, кликая по узлам графа в окне

визуализации вершин. На рисунке 17 выбраны вершины OHC:org-sys, OHC:participation, OHC:sys-obj для левой части и AKT:Organization, AKT:Emploee - для правой. Отношения между выбранными классами, как видно, так же показываются другим цветом и отображаются в «Списке Классов и Отношений».

При нажатии на кнопку «Send request», пользователь получает возможность отправить SPARQL-запрос. В появившемся окне (рис. 18) есть уже готовый запрос для выбранных структур. В нашем примере это запрос для структур вида:

1. «AKT:Organization - works-for - AKT:Emploee»
2. «OHC:org-sys – in-org - OHC:sys-obj – participant - OHC:participation»

Данный запрос можно поменять или можно отправить на выбранную точку доступа. И получить ответ в xml-файл, имя которого так же можно выбрать в «Окне построения запроса»

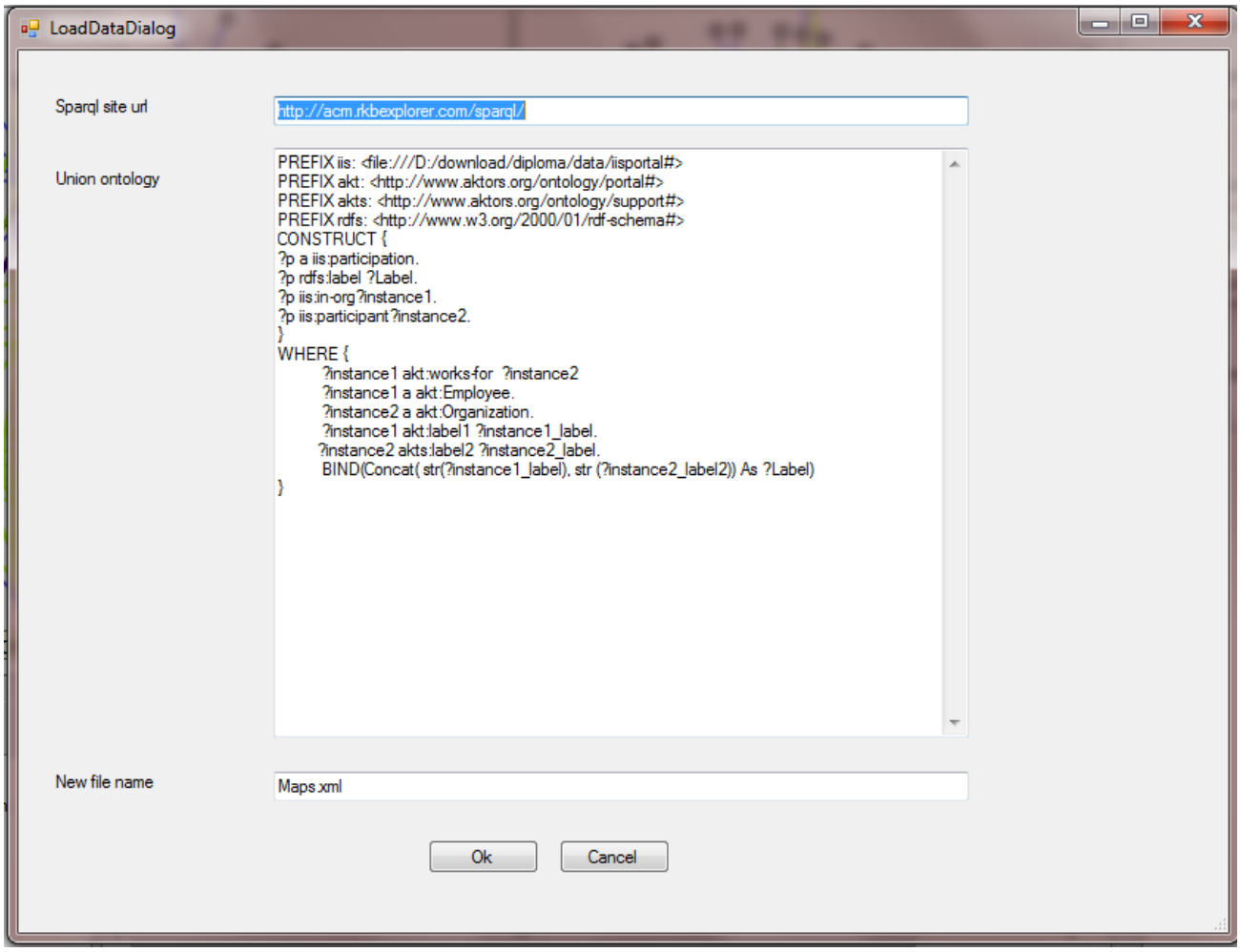

Рисунок 18. Окно построения запроса.

7. Заключение

В ходе выполнения данной работы получены следующие результаты. Изучены подразделы новой предметной области Semantic Web, такие как понятие онтологии в целом, форматы хранения онтологий RDF и OWL,а так же язык запросов к данным – SPARQL. Исследованы программные средства, позволяющие различные манипуляции с онтологиями, и особенно подробно рассмотрена система Agreement Maker, как инструмент для решения задачи установления соответствия между двумя онтологиями. Изучены существующие методы для отображения двух онтологий. Разработан и реализован лексический алгоритм установления соответствия между классами. Разработаны новые функциональные возможности для программы ”Визуализации онтологии и информационного наполнения семантических систем”, позволяющая устанавливать соответствия в интерактивном режиме. Добавлена возможность автоматического формирования SPARQL запроса для получения данных по построенным соотношениям. Работоспособность этого модуля проверена на двух реальных онтологиях.

В дальнейшем планируется выгрузка списка построенных соответствий в RDF файл.

Список литературы

1. **Z.V. Apanovich, A.G. Marchuk** Experiments on ontology based semantic systems integration// Bull. Nov. Comp. Center, Comp.Science, 34(2012), pp. 43-54.
2. **А.С. Клещев, И.Л. Артемьева** Отношения между онтологиями предметных областей Часть1. Онтологии, представляющие одну и ту же концептуализацию, упрощение онтологий// Научно-техническая информация, Сер.2, 2002, №1. С.4-16
3. **Isabel F. Cruz, Flavio Palandri Antonelli** Efficient Selection of Mappings and Automatic Quality-driven Combination of Matching Methods// Fourth International Workshop on Ontology Matching, co-located with the International Semantic Web Conference, Chantilly, Virginia, 2009
4. Linked Open Data, 2011 [Электронный ресурс]. Режим доступа: http://www.w3.org/wiki/SweoIG/TaskForces/CommunityProjects/LinkingOpenData. Дата обращения: 10.03.2012.
5. **Эрик Прудхоммоукс** Язык запросов SPARQL для RDF [Электронный ресурс]. Режим доступа: http://shcherbak.net/translations/ru_sparql_shcherbak_net.html#docDataDesc Дата обращения: 10.03.2012.
6. **J. Broekstra, M. Klein, S. Decker, D. Fensel, F. van Harmelen, and I. Horrocks** Enabling knowledge representation on the web by extending RDF schema.//Proceedings of the tenth World Wide Web conference WWW'10.— pp. 467–478.— 2000.
7. **Ломов П. А., Шишаев М. Г.** Интеграция семантически связанных информационных ресурсов на основе онтологий RDF [Электронный ресурс]. Режим доступа: http://crider.rork.ru/index.php?option=com_content&task=view&id=14&Itemid=31 Дата обращения: 10.10.2012.
8. W3C Рекомендация SPARQL Query Language for RDF [Электронный ресурс]. Режим доступа: http://www.w3.org/TR/rdf-sparql-query/ Дата обращения: 11.10.2012.
9. SPARQL Implementation Coverage Report [Электронный ресурс]. Режим доступа: http://www.w3.org/2001/sw/DataAccess/tests/implementations Дата обращения: 11.10.2012.
10. Noy N., Musen M. The PROMPT Suite: Interactive Tools For Ontology Merging And Mapping // Stanford Medical Informatics, Stanford Univ., 2003.
11. McGuinness D., Fikes R., Rice J., Wilder S. An environment for merging and testing large ontologies // In Proc. of the Seventh Int. Conf., KR2000, Morgan Kaufmann Publishers, San Francisco, CA,2000.

12. Dou D., McDermott D., Qi P., Ontology translation by ontology merging and automated reasoning // EKAW'02 workshop on Ontologies for Multi-Agent Systems. SigËuenza, Spain, 2002.
13. Chalupsky H, OntoMorph: A translation system for symbolic knowledge // In: Cohn A. G., Giunchiglia F., Selman B. (Eds.), Principles of Knowledge Representation and Reasoning // Proc. of the Seventh Intern. Conf. (KR2000). Morgan Kaufmann Publishers, San Francisco, CA, 2000.
14. Mena E., Illarramendi A., Kashyap V., Sheth A. OBSERVER: An approach for query processing in global information systems based on interoperation across preexisting ontologies // Distributed and Parallel Databases, An International Journal, Vol.8, No.2, 2000.
15. Stumme G., Medche A. FCA-Merge: Bottom-up merging of ontologies // 7th Int. Conf. on Artificial Intelligence, (IJCAI'01), Seattle, WA, 2001, P.225–230.
16. Mitra P., Wiederhold G., Decker S. A scalable framework for interoperation of information sources //The 1st Intern. Semantic Web Working Symp., SWWS'01, Stanford University, CA, 2001.
17. Agreement Maker [Электронный ресурс]
    Режим доступа: https://agreementmaker.org/.
    Дата обращения: 09.10.2012.
18. AKT ontology description [Электронный ресурс].
    Режим доступа: http://www.aktors.org/ontology.
    Дата обращения: 11.11.2012.
19. DBLP dataset [Электронный ресурс]
    Режим доступа: http://dblp.rkbexplorer.com/.
    Дата обращения: 11.11.2012.
20. Virtuoso SPARQL Query Editor [Электронный ресурс]
    Режим доступа: http://demo.openlinksw.com/sparql
    Дата обращения: 11.01.2013.
21. Данные для ОНС онтологии [Электронный ресурс]
    Режим доступа: http://duh.iis.nsk.su/VirtuosoEndpoint/Home/Samples
    Дата обращения: 11.01.2013.