Московский государственный университет имени М.В. Ломоносова
Факультет вычислительной математики и кибернетики
Кафедра алгоритмических языков

Домашнее задание

По курсу «Прикладные задачи компьютерной лингвистики»

Выполнил:
студент 624 группы
Баев Олег Дмитриевич

Москва, 2016

Постановка задачи

Задачей данной работы является разработка и реализация программы, извлекающей факты владения собственностью (ownership). Для извлечения фактов используется инженерный подход, основанный на применении лингвистических правил и шаблонов. Факт владения представляется владельцем и собственностью (организация), которая принадлежит данному владельцу. В качестве владельца может выступать как человек, так и организация. Предполагается, что персоны и организации уже выявлены.

Описание программы

Для реализации программы используется Томита-парсер – инструмент, созданный для извлечения структурированных данных из текста на естественном языке. Извлечение фактов происходит при помощи контекстно-свободных грамматик (правил, шаблонов) и словарей ключевых слов.

Парсер включает в себя три стандартных лингвистических процессора: токенизатор (разбиение на слова), сегментатор (разбиение на предложения) и морфологический анализатор (mystem).

Основные компоненты парсера: газеттир, набор КС-грамматик и множество описаний типов фактов, которые порождаются этими грамматиками в результате процедуры интерпретации.

Газеттир – это словарь ключевых слов, которые используются в процессе анализа КС-грамматиками. Каждая статья этого словаря задает множество слов и словосочетаний, объединенных общим свойством. Газеттир описывается в отдельном файле, имеющем специальный тип “.gzt”.

В данной работе используется газеттир, содержащий ФИО людей, т.е. три статьи для имени, фамилии и отчества, а также газеттир для названия организаций. В каждой статье указывается путь к файлу, содержащему ключевые слова. Например, в статье “org_name” в поле key содержится путь к файлу с названиями организаций:

```
TAuxDicArticle "org_name"
{
    key = { "../dicts/org_names.txt" type=FILE }
}
```

Факты – таблицы с колонками, которые называются полями фактов. Факты заполняются во время анализа парсером предложения. Как и чем заполнять поля фактов указывается в каждой конкретной грамматике. Это называется интерпретацией. Типы фактов описываются на специальном языке в файле “facttypes.proto”.

Факт владения Ownership содержит поле Property – название собственности (организации), а также поля, относящиеся к владельцу:

- Owner_surname – фамилия персоны;
- Owner_name – имя персоны;
- Owner_patronymic – отчество персоны;
- Owner_org_name – название организации-владельца.

Описывается данный факт следующим образом:

```
message Ownership: NFactType.TFact
{
  optional string Owner_surname = 1;
  optional string Owner_name = 2;
  optional string Owner_patronymic = 3;
  optional string Owner_org_name = 4;
  required string Property = 5;
}
```

Обязательность заполнения поля факта задается служебным словом required или optional, затем следует название типа поля, а затем его название и после знака равенства “=” номер поля факта. Если в процессе интерпретации остается незаполненным хотя бы одно обязательное (required) поле, то факт не извлекается. Поля, относящиеся к владельцу отмечены необязательными, т.к. владельцем является либо персона, либо организация.

Грамматика – множество правил на языке КС-грамматик, описывающих синтаксическую структуру выделяемых цепочек. Грамматический парсер запускается всегда на одном предложении. Перед запуском терминалы грамматики отображаются на слова (или словосочетания) предложения. Одному слову может соответствовать много терминальных символов. Таким образом, парсер получает на вход последовательность множеств терминальных символов. На выходе получаются цепочки слов, распознанные этой грамматикой. Описывается грамматика в файле типа “.cxx”.

Для извлечения факта владения собственностью (ownership) используются следующие правила (шаблоны):

```
S -> Owns Word<kwtype=~'org_name'>* Properties Person;
S -> Owns Person Properties;
S -> Person AnyWord<kwset=~['name','surname','patronymic']>* Owns
       AnyWord<kwtype=~'org_name'>* Property;
S -> Property AnyWord<kwset=~${dicts}>* Owns AnyWord<kwset=~${dicts}>*
       OrgOwner;
```

```
S -> FamilyOwner AnyWord<kwset=~${dicts}>* Owns
       AnyWord<kwset=~${dicts}>* Property;
S -> Person<gram="род"> Property;
S -> "продать" Property Person<gram="дат">;
```

Правило Owns описывает возможные отношения владельца и собственности, задается следующим образом:

```
Owns -> "основатель" | "владелец" | "совладелец" | "соучредитель" |
"создатель" | "собственник" | "хозяин" | "основать" | "владеть" |
"создать" | "принадлежать" | "открыть" | "подконтрольный" | "входить"
| "доля";
```

Правило Person позволяет извлекать персоны, используя газеттир для ФИО людей, и записывать данные в поля факта (interp). Один из шаблонов для определения персоны:

```
Person -> (Word<kwtype='surname'> interp (Ownership.Owner_surname))
          Word<kwtype='name'> interp (Ownership.Owner_name)
          (Word<kwtype='patronymic'> interp (Ownership.Owner_patronymic));
```

Выражение записанное в круглых скобках может встречаться один раз или не встречаться вообще. В kwtype записывается словарь, в котором должно содержаться слово. Также учитывается возможность использования инициалов. С помощью регулярных выражений один инициал можно описать следующим образом:

```
Initial -> Word<wff=/[А-Я]\./>;
```

Кроме персоны владельцем может выступать организация или семья, представляется правилом OrgOwner:

```
Org -> Word<kwtype='org_name'>;
Family -> "семья" Word<kwtype='surname'>;
FamilyOwner -> Family interp (Ownership.Owner_org_name);
OrgOwner -> Org interp (Ownership.Owner_org_name) | FamilyOwner;
```

Правило Property описывает собственность (организация):

```
Property -> Org interp (Ownership.Property);
```

Также задается правило Properties для нескольких организаций, отделенных запятой или союзом «и».

Рассмотрим один из шаблонов для извлечения факта владения:

```
S -> Owns Word<kwtype=~'org_name'>* Properties Person;
```

С помощью данного шаблона можно извлечь факт, например, из предложения:

«Основатель "ВКонтакте" Павел Дуров рассказал об успехах Telegram.»

Соответственно факт будет выглядеть следующим образом:

Owner_surname: Дуров

Owner_name: Павел

Property: ВКонтакте

Для запуска парсера необходимо задать корневой словарь “dic.gzt”, который содержит перечень всех используемых в проекте словарей и грамматик, а также конфигурационный файл “config.proto”, который сообщает парсеру, где искать все остальные файлы, как их интерпретировать и что делать. В конфигурационном файле указывается файл с текстом, из которого следует извлекать факты, и можно задать формат извлекаемых фактов, текстовый или xml, удобный для дальнейшей обработки.

Тестирование

Для тестирования используется размеченная коллекция соревнования FactRuEval 2016. Для каждого текста в файле “.facts” содержатся факты различных типов, включая факты владения собственностью (Ownership), например:

```
3573-0 Ownership
Owner obj3632 Дуров Павел
Property obj3633 ВКонтакте
```

В файле “.spans” указываются типизированные спаны – непрерывные цепочки слов в тексте, имеющие один или несколько из заранее определённых типов. Стоит выделить следующие типы спанов: для персон – surname (фамилия), name (имя), patronymic (отчество); для организаций – org_name (название). Несколько строк подобного файла:

```
87471 org_name 120 9 1786827 1  # 1786827 ВКонтакте
87472 name 131 5 1786829 1  # 1786829 Павел
87473 surname 137 5 1786830 1  # 1786830 Дуров
```

Был написан python-скрипт, извлекающий для каждого текста только факты владения и сохраняющий их в формате структуры данных python dict для удобства дальнейшего использования. Для каждого файла со спанами формируются отдельные файлы с фамилиями, именами, отчествами персон, а также файл с названиями организаций.

Для тестирования также используется python-скрипт, запускающий Томита-парсер для каждого текста коллекции с полученными ранее ФИО персон и названиями организаций. На выходе парсера получаются “.xml” файлы с извлеченными фактами.

Результаты

Для оценки работы программы был написан python-скрипт, который обрабатывает ".xml" файлы с извлеченными фактами и сравнивает результат с фактами, полученными из размеченной коллекции. Используется F-мера:

$F1 = \frac{2PR}{P+R}$, где

P – точность, отношение числа правильно извлеченных фактов к общему числу найденных фактов;

R – полнота, отношение числа правильно извлеченных фактов к общему числу фактов, которые содержатся в коллекции.

В итоге были получены следующие результаты:

```
P:  0.4310
R:  0.1470
F1: 0.2192
```

Для сравнения, лучший результат, полученный на соревновании FactRuEval 2016:

```
P:  0.5379
R:  0.1709
F1: 0.2594
```