Разработка ядра синтаксического анализатора для русского языка на основе библиотек NLTK

А.Д. Москвина, Д. Орлова, П.В. Паничева, О.А. Митрофанова

Санкт-Петербургский государственный университет
moskvina.anya@gmail.com, frenezo@mail.ru, p.panicheva@spbu.ru, o.mitrofanova@spbu.ru

Аннотация

Целью проекта является создание синтаксического анализатора для русского языка с использованием инструментов NLTK для Python. В NLTK есть возможность создавать категориальные грамматики (feature-based grammars), основывающиеся на морфологической информации о входном тексте. При написании правил грамматики мы опираемся на систему морфологической разметки, принятую в морфоанализаторе PyMorphy2. На данный момент создано ядро синтаксического анализатора, рассчитанное на обработку основных синтаксических групп внутри простого предложения для русского языка. В будущем мы планируем расширить функционал парсера так, чтобы он мог анализировать любые грамматически верные предложения русского языка.

Ключевые слова: автоматическая обработка текстов, синтаксический анализатор, русский язык, NLTK.

1. Введение

1.1. Постановка проблемы

Методы автоматического анализа естественного языка и компьютерные технологии обработки информации — это наиболее динамично развивающиеся области компьютерной лингвистики в наши дни. Тем не менее, в этих областях остаются задачи, до сих пор не нашедшие общепризнанного решения, и среди них синтаксический анализ, наиболее сложный этап досемантической обработки текста, без которого невозможно осуществлять процедуры извлечения фактов, автоматического перевода, реферирования, построения вопросно-ответных систем и многие другие.

На сегодняшний день известны различные синтаксические анализаторы (парсеры), однако большинство из них разрабатывалось прежде всего для английского языка, например, Stanford Parser [http://nlp.stanford.edu/software/lex-

parser.shtml]. В области автоматической обработки русского языка на уровне синтаксиса накоплен богатый и разнообразный опыт, восходящий к исследованиям основателей отечественной математической лингвистики [1–5 и т.д.]. Идеи модели «Смысл <=> Текст» воплотились в многоцелевом лингвистическом процессоре «Этап-3» [http://iitp.ru/ru/science/works/452.htm] [6], с помощью которого был создан первый синтаксически размеченный корпус русских текстов СинТагРус [http://ruscorpora.ru/instruction-syntax.html] [7]. В начале 2000-х годов получил известность проект Диалинг (АОТ) [http://www.aot.ru/], в рамках которого появился доступный синтаксический анализатор с открытой документацией [8].

Реальное состояние дел в области синтаксического анализа русских текстов было оценено в 2012 году на соревновании парсеров [http://www.dialog-21.ru/digest/2012/?type=syntax], где участвовали 8 команд (среди них SyntAutom [9], DictaScope Syntax, SemSyn [10], Этап-3 [6], парсер Semantic Analyzer Group, АОТ [8], ABBY Syntactic and Semantic Parser [11], Link Grammar Parser [12]) [13]. По результатам был подготовлен «золотой стандарт» синтаксической разметки объемом 800 предложений с описанием допустимых расхождений в анализе. Благодаря соревнованию появился банк синтаксических структур RSTB, полученных от трех анализаторов: SyntAtom, SemSin, Russian Malt [http://otipl.philol.msu.ru/~soiza/testsynt/files/info.htm]. Хотя это соревнование не привело к выработке общепринятого стандарта для русскоязычных парсеров, тем не менее, наметились направления дальнейшей работы.

Сейчас многие научные лаборатории, работающие в сфере компьютерной лингвистики, используют свои собственные синтаксические анализаторы (ABBYY, RCO, Dictum и т.д.). Что касается процессоров, открытых для разработчиков и свободно распространяемых, то их, во-первых, мало и они разных типов (для русского, помимо АОТ, можно воспользоваться, к примеру, Link Grammar Parser [http://slashzone.ru/parser/] [12], MaltParser [http://corpus.leeds.ac.uk/mocky/] [14], а во-вторых, при работе с ними нужно дополнительно решать задачи по улучшению качества анализа и по синхронизации их с другими модулями конкретных лингвистических процессоров.

1.2. Цель и задачи обсуждаемого проекта

Нам представляется, что путь к решению проблемы автоматического синтаксического анализа лежит в сторону развития открытых некоммерческих лингвистических платформ, среди которых одно из первых мест занимает NLTK (Natural Language Toolkit) [http://www.nltk.org/] [15], набор библиотек для языка программирования Python, ориентированный на выполнение основных процедур автоматической обработки текстов: от графематического анализа до синтаксического парсинга, от составления словарей до построения сложных статистических моделей корпусов текстов. Синтаксический анализ основывается на данных морфологической разметки корпуса текстов и разрешения морфологической неоднозначности. Наш парсер ориентирован на широкодоступный формат морфологической аннотации русских текстов, представленный в тегсете морфоанализатора PyMorphy2 [http://pymorphy2.readthedocs.org/en/latest/] [16].

Итак, целью нашего проекта является создание открытого синтаксического парсера для русского языка на платформе NLTK. Для достижения этой цели требуется решить следующие задачи:

- разработать систему синтаксических правил для выделения синтаксических групп, составить категориальную грамматику в формате NLTK;
- синхронизировать правила категориальной грамматики для создаваемого парсера и морфологический анализатор для русского языка, работающий с общепринятой системой морфологической разметки (тегсетом);
- провести пилотные эксперименты, проанализировать типовые ошибки работы парсера.

2. Архитектура разрабатываемого синтаксического анализатора

2.1. Модуль синтаксического анализа в NLTK

Модуль синтаксического анализа в NLTK позволяет разработчикам самостоятельно создавать формальные грамматики различных типов для разных естественных языков и применять их в конкретных целях автоматической обработки текстов.

Формальная грамматика описывает потенциально бесконечный набор всех возможных синтаксически верных предложений (конструкций). Грамматики могут быть контекстно-свободными, вероятностными контекстно-свободными, лексикализованными и контекстно-зависимыми. С помощью набора синтаксических категорий и набора правил (productions) контекстно-свободная грамматика определяет, как фраза категории A может быть представлена в виде последовательности более маленьких частей α1… αn. [15].

Синтаксический анализ (парсинг) — это процедура нахождения одного или более вариантов разбора (деревьев), соответствующих грамматически правильным предложениям. В NLTK есть несколько готовых парсеров, предполагающих разные пути проверки соответствия предложений определенному синтаксическому формализму. Прежде всего, это нисходящий парсер (simple top-down parser), который предполагает последовательное применение правил для заданной левой части и сопоставление им предложения на входе. Также есть возможность применить восходящий парсер (simple bottom-up parser), который обрабатывает предложения по очереди и ищет для них соответствующую правую часть.

В правилах грамматики может применяется рекурсия, тогда категория из левой части продукции повторяется в правой. Рекурсия может быть прямая и непрямая. Благодаря использованию рекурсии в формальной грамматике мы получаем возможность коротко описать сложные вложенные конструкции.

Итак, встроенный в NLTK парсер обрабатывает поступающие на вход предложения на основании правил грамматики, хранящихся в файле определенного формата, и строит структуру составляющих (или несколько структур). Эти структуры отражают то, как слова и последовательности слов

сочетаются, формируя синтаксические группы. Появление более чем одного разбора говорит о наличии синтаксической неоднозначности.

2.2. Особенности категориальных грамматик

В нашем анализаторе мы используем вид категориальной грамматики, основанный на признаках категорий (feature-based grammar). Говоря о категориальной грамматике, мы подразумеваем наличие в ней таких категорий, как, например, именная группа (NP) или глагол (VERB). Особенностью рассматриваемого формализма является возможность работать с так называемыми структурами признаков, т.е., в нашим случае оперировать информацией о некоторых изменяемых параметрах этих категорий. Таким образом мы можем, например, явно указывать морфологические особенности компонентов сочетания:

S -> NP[CASE=nomn, NUMBER=?n, GENDER=?g] VP[NUMBER=?n, GENDER=?g]

Здесь мы используем переменные для выражения значений признаков. Переменная ?g, задающая признак GENDER (род) категории VP (глагольная группа), может обозначать как мужской, так и женский род. Однако, используя эту переменную в нескольких частях одного правила, а именно в именной группе и в глагольной группе, мы тем самым указываем, что их значение должно совпадать. Таким образом в приведенном примере мы задаем согласование подлежащего и сказуемого по числу и роду.

С грамматиками такого вида в NLTK используется алгоритм Эрли (Earley-chart parser), который находит и сохраняет фрагменты разбора предложения, а потом соединяет их в группы.

2.3. Правила выделения синтаксических групп для русского языка

Правила разрабатываемой нами категориальной грамматики для синтаксического парсера опираются на морфологическую информацию, используемую при разметке русскоязычных текстов. В нашем проекте задействован морфологический анализатор PyMorphy2. Чтобы использовать полученную с его помощью разметку для синтаксического анализа, мы создали функцию, которая представляет морфологические параметры словоформ PyMorphy2 в виде терминальных элементов в категориальной грамматике NLTK. Из всех тегов, предоставляемых морфоанализатором, мы отобрали наиболее значимые – именно эти грамматические параметры будут отражены в правилах нашей грамматики. Например, для существительных такими значимыми параметрами являются род, число и падеж. Все эти категории необходимы для выделения конструкций согласованного определения. Другой вариант использования информации о падеже у существительного — обязательный номинатив в позиции подлежащего при выделении конструкции предложения (клаузы). Очевидно, эти параметры должны быть специфическими для разных частей речи. Тем не менее, некоторые части речи (например, существительные и прилагательные, или все неизменяемые части речи) можно объединить в классы. И, наоборот, внутри одной части речи иногда приходится выделять отдельные классы (глаголы настоящего и прошедшего времени).

Значения для таких классов записываются в соответствующие поля в правилах типа:

NOUN[CASE=gent, GENDER=femn, NUMBER=sing, NF=u'ягода'] -> 'ягоды'
NOUN[CASE=nomn, GENDER=femn, NUMBER=plur, NF=u'ягода'] -> 'ягоды'
NOUN[CASE=accs, GENDER=femn, NUMBER=plur, NF=u'ягода'] -> 'ягоды'

Для всех слов анализируемого текста в правило записывается также часть речи, начальная форма и словоформа. Выше представлены правила, записанные в файл с грамматикой, соответствующие трём вариантам разбора, предлагаемого PyMorphy2 для словоформы «ягоды». Такие правила описывают морфологию отдельных слов.

Основная задача в создании нашего парсера — написание правил выделения синтаксических групп (grammar productions). Они описывают необходимые условия для объединения составляющих в ту или иную группу. К таким условиям относятся линейный порядок словоформ и ограничения по их морфологическим характеристикам. Правила грамматики записываются в отельный файл формата .fcfg, с которым работает NLTK.

При составлении правил мы учитывали опыт других систем, в частности, мы адаптировали для категориальной грамматики многие синтаксические группы, описанные в документации синтаксического модуля АОТ (http://www.aot.ru/docs/synan.html). Например, так выглядит правило выделения генитивной группы (ГЕНИТ_ИГ в АОТ):

NP[+gent, CASE=?c, GENDER=?g, NUMBER=?n] -> NP[CASE=?c, GENDER=?g, NUMBER=?n] NP[CASE=gent].

Правая часть правила задает линейный порядок составляющих группы, то есть именная группа + именная группа в родительном падеже, левая — её название и характеристики, которые она наследует от главного компонента сочетания (вершина группы в АОТ).

Мы написали правила для выделения отдельных именных, глагольных, предложных групп, групп с числительными и наречиями, некоторых групп, основанных на сочинении. Правила более высокого уровня описывают, как эти группы соединяются между собой. В общем случае, при формулировании правила нужно было принять решение относительно того, с какими признаками мы работаем, какие значения они могут принимать и какие необходимо установить ограничения.

Помимо того, чтобы задавать признак строкой (CASE=accs) или числом (PERS=3), мы можем также приписывать группе некоторый параметр, имеющей булево значение. Мы используем эту возможность для добавления к основной категории некоторой подкатегории.

NP[+adjf, CASE=?c, GENDER=?g, NUMBER=?n] -> AdjP[CASE=?c, GENDER=None, NUMBER=?n] NP[CASE=?c, GENDER=?g, NUMBER=?n]\n")

NP[+gent, CASE=?c, GENDER=?g, NUMBER=?n] -> NP[CASE=?c, GENDER=?g, NUMBER=?n] NP[CASE=gent]

Первое правило представляет собой объединение в именную группу согласованных прилагательного и существительного. В левой части правила мы имеем обычную именную группу — NP, однако сохраняем информацию о том, что она содержит в себе прилагательное +adjf, то есть параметр adjf получает значение истины. Аналогично, во втором примере сохраняется информация о

присоединении генитивной формы (+gent). Так мы можем сколько угодно раз усложнять нашу NP, и, в итоге, она останется NP и сможет играть роль субъекта, объекта или генитива в более крупной группе — то есть обеспечивается необходимая рекурсия. Подобным образом можно задавать и ограничения на применение правил. Сохранение информации о подкатегории позволяет, например, запретить объединять глагольную группу переходного глагола, у которого уже есть прямое дополнение, с ещё одним объектом следующим образом:

VP[+objt, NUMBER=?n, PERS=?p, GENDER=?g, TRANSITIVITY=tran] -> VP[-objt, NUMBER=?n, PERS=?p, GENDER=?g, TRANSITIVITY=tran] NP[CASE=accs]

Например, при разборе фразы «вижу хвост кота» группа «вижу хвост», объединившись по этому правилу, уже не сможет присоединить к себе следующее линейно за ним слово «кота» в качестве объекта, так как в правой части правила указано, что значение objt должно быть ложью.

Для удобства составления правил в такой системе, где важным является сравнение значений признаков категорий, мы решили обозначать отсутствующие значения признаков, такие как лицо у глаголов прошедшего времени и род у глаголов настоящего и будущего времени, нулем. Это позволяет накладывать необходимые ограничения на правила. Формы инфинитива записываются в грамматику с признаками глагола, имеющими значение 0.

Правила объединения именной и глагольной группы в предложение (без инверсии) выглядят следующим образом:

XP[-inv] -> NP[CASE=nomn, NUMBER=?n, PERS=?p, GENDER=?g] VP[NUMBER=?n, PERS=0, GENDER=?g]

XP[-inv] -> NP[CASE=nomn, NUMBER=?n, PERS=?p, GENDER=?g] VP[NUMBER=?n, PERS=?p, GENDER=0]

Мы добавили всем существительным фиктивный признак 3 лица, что позволяет ограничиться двумя приведенными выше правилами для согласования всех существительных и местоимений с глаголами настоящего, прошедшего и будущего времен. Согласование происходит по числу и роду, если признак лица равен нулю, либо по числу и лицу, если отсутствует признак рода.

2.4. Программная реализация синтаксического анализатора

Программа, реализующая наш парсер, написана на языке Python (версия 3.4.3), использует инструменты пакета NLTK и морфологический анализатор PyMorphy2 и, соответственно, предполагает предустановку этих трех компонентов.

При запуске программы в файл записывается сама грамматика, то есть разработанные нами правила. Эта часть грамматики обозначается как «grammar productions». После того, как пользователь вводит предложение для разбора, оно разбивается на токены. Далее, для полученного списка вызывается функция, переводящая морфологическую информацию о словоформах в вид правил категориальной грамматики и записывающая их в файл с грамматикой. Такие правила попадают в раздел «lexical productions». Теперь у нас есть

полноценная категориальная грамматика с грамматическими и лексическими продукциями, на основе которой может проводиться синтаксический анализ. Тогда вызывается встроенный в NLTK парсер, который производит разбор предложения на основе нашей грамматики.

Допустим, пользователь вводит предложение «Человек видит лапу кота». Тогда на выходе будет получен следующий разбор предложения (см. рис. 1). Здесь можно наглядно увидеть, каким образом категории (группы) объединяются в составляющие и какие характеристики сохраняются для какой группы. При наличии синтаксической неоднозначности будет предъявлено несколько вариантов разбора.

Так выглядят некоторые из правил, применившихся при разборе данного предложения:

XP[-inv] -> NP[CASE=nomn, NUMBER=?n, PERS=?p, GENDER=?g] VP[NUMBER=?n, PERS=?p, GENDER=0]

NP[CASE=?c, GENDER=?g, NUMBER=?n] -> NOUN[CASE=?c, GENDER=?g, NUMBER=?n] человек

VP[+objt, NUMBER=?n, PERS=?p, GENDER=?g] -> VP[-objt, NUMBER=?n, PERS=?p, GENDER=?g] NP[CASE=accs] *видит лапу кота*

VP[TENSE=?t, GENDER=?g, NUMBER=?n, PERS=?p] -> VERB[TENSE=?t, GENDER=?g, NUMBER=?n, PERS=?p] *видит*

NP[CASE=?c, GENDER=?g, NUMBER=?n, +gent] -> NP[CASE=?c, GENDER=?g, NUMBER=?n] NP[CASE=gent]\n") *лапу кота*

```
(XP[-inv]
 (NP[CASE='nomn', GENDER='masc', NUMBER='sing']
  (NOUN[CASE='nomn', GENDER='masc', NF='человек', NUMBER='sing', PERS=3]
   человек))
 (VP[GENDER=0, NUMBER='sing', PERS=3, +objt]
  (VP[GENDER=0, NUMBER='sing', PERS=3, TENSE='pres']
   (VERB[GENDER=0, NF='видеть', NUMBER='sing', PERS=3, TENSE='pres']
    видит))
  (NP[CASE='accs', GENDER='femn', NUMBER='sing', +gent]
   (NP[CASE='accs', GENDER='femn', NUMBER='sing']
    (NOUN[CASE='accs', GENDER='femn', NF='лапа', NUMBER='sing', PERS=3]
     лапу))
   (NP[CASE='gent', GENDER='masc', NUMBER='sing']
    (NOUN[CASE='gent', GENDER='masc', NF='кот', NUMBER='sing', PERS=3]
     кота)))))
```

Рис. 1. Разбор предложения «человек видит лапу кота»

3. Заключение

3.1. Обсуждение проблем

На сегодняшний день разрабатываемый парсер успешно справляется с разбором простых предложений. Вместе с тем, переход нашего синтаксического анализатора в рабочее состояние потребовал предварительного решения некоторых проблем.

Первый класс проблем связан с особенностями синтаксической организации русского предложения. В частности, при работе парсера возникают сложности, вызванные относительно свободным порядком слов («мать любит дочь»), омонимией на уровне слов и словоформ (например, слово «его» с высоким индексом неоднозначности), синтаксической неоднозначностью («трость из кости Екатерины Второй»).

Второй класс проблем определяется спецификой функционирования и видом выходных данных морфологического анализатора Pymorphy2. В частности, мы вынуждены были преодолеть следующие трудности.

Некоторые короткие слова (союз «и», предлог «в») морфоанализатор Pymorphy2 размечает как сокращение от более длинного слова («исполняющий»), что порождает несколько вариантов разбора, казалось бы, однозначного предложения. На данный момент было принято решение использовать только первый вариант разбора в Pymorphy2, который как раз и представляет собой нужный предлог или союз.

Pymorphy2 избирательно проводит анализ субстантивированных прилагательных. Так, слово «красный» размечается как существительное, в то же время слово «больной» трактуется исключительно как прилагательное.

Обычным арабским цифрам PyMorphy2 присваивает значение NUMR, но при этом не хранит это значение в той же переменной, куда помещает данные о частях речи и характеристиках других токенов. Это создаёт трудности в извлечении информации для парсера.

Морфологический анализатор не справляется с разбором слов, в которых произведена замена буквы «ё» на «е», хотя в современной русской письменной речи это явление считается достаточно распространённым.

Третья категория проблем вызвана особенностями устройства категориальной грамматики в NLTK. В первую очередь нужно отметить, что грамматика составляющих хорошо зарекомендовала себя в автоматической обработке языков с более строгим порядком слов, таких как английский или немецкий. В русском языке синтаксические группы могут быть разнесены по всему предложению, поэтому их выделение порой представляет собой определённые трудности. Многие современные синтаксические анализаторы во избежание подобных проблем опираются на грамматику зависимостей (например, анализатор Этап-3 [6]), другие же стремятся реализовать гибридный подход (например, метод тринотаций в парсере TreeTon [17]).

Еще одна сложность связана с тем, что контекстно-свободные грамматики, к которым относится используемая нами категориальная грамматика, задаются на основе формальных морфологических параметров и не учитывают лексическое значение слова, что не позволяет разработчикам опираться на семантические

свойства лексем для устранения возникающей синтаксической неоднозначности.

Наконец, категориальная грамматика в NLTK считает свой разбор верным только тогда, когда может построить дерево составляющих полностью. В случае если грамматика не может присвоить категорию хотя бы одному токену в предложении, она считает все остальные составленные группы по умолчанию неверными. Это не представляется нам целесообразным, так как, во-первых, важна любая информация о фрагментах структуры предложения, во-вторых, для некоторых задач автоматической обработки текста не требуется полный разбор предложения.

3.2. Перспективы развития исследования

На данный момент функционирует ядро синтаксического анализатора для русского языка на основе библиотек NLTK. При расширении ядра мы планируем увеличить количество правил: будут добавлены правила для работы с числительными, причастными и деепричастными оборотами, вводными словами. Также будет создан механизм выбора более вероятного разбора предложения. В основе этого механизма будет лежать концепция силы связей между членами предложения, принятая в теории конструктивного синтаксиса Н.Ю. Шведовой [18]. Мы планируем ввести в парсер дополнительный механизм задания приоритета для правил. Так, например, более предпочтительным будет считаться разбор с немаркированным порядком слов («субъект-глагол-предикат»). Кроме того, с помощью дополнительного набора правил мы введем возможность разбирать фрагмент предложения в том случае, если оно не может быть разобрано целиком. Наконец, мы планируем синхронизировать наш синтаксический анализатор с ранее созданным и протестированным гибридным морфологическим анализатором, использующим теггеры NLTK и PyMorphy2 [19].

По итогам проведенного исследования можно утверждать, что идея создания синтаксического анализатора на основе категориальной грамматики является состоятельной. На данный момент мы продолжаем работу над расширением списка правил и устранением ошибок анализа, чтобы перейти к тестированию парсера на русскоязычном корпусе текстов и оценке результатов его разборов.

3.3. Благодарности

Исследование поддержано грантом РФФИ № 16-06-00529 «Разработка лингвистического комплекса для автоматического семантического анализа русскоязычных корпусов текстов с применением статистических методов».

Литература

[1] Гладкий А.В. Синтаксические структуры естественного языка в автоматизированных системах общения. М., 1985.

[2] Мельчук И.А. Опыт теории лингвистических моделей «Смысл <=> Текст». М., 1974/1999.

[3] Мельчук И.А. Автоматический синтаксический анализ. Т. 1. Общие принципы. Внутрисегментный синтаксический анализ. Новосибирск, 1964.

[4] Иорданская Л.Н. Автоматический синтаксический анализ. Том 2. Межсегментный синтаксический анализ. Новосибирск, 1967.
[5] Фитиалов С.Я. Формальные грамматики. Л., 1984.
[6] Iomdin L., Petrochenkov V., Sizov V., Tsinman L. ETAP parser: state of the art // Компьютерная лингвистика и интеллектуальные технологии: По материалам ежегодной Международной конференции «Диалог» (Бекасово, 30 мая — 3 июня 2012г.). М., 2012.
[7] Апресян Ю.Д. и др. Синтаксически и семантически аннотированный корпус русского языка: современное состояние и перспективы // Национальный корпус русского языка 2003–2005 г. (результаты и перспективы). М., 2005. С. 193–214.
[8] AOT: Синтаксический анализ. Построение дерева зависимостей всего предложения. URL: http://www.aot.ru/docs/synan.html.
[9] Antonova A.A., Misyurev A.V.Russian dependency parser SyntAutom at the DIALOGUE-2012 parser evaluation task // Компьютерная лингвистика и интеллектуальные технологии: По материалам ежегодной Международной конференции «Диалог» (Бекасово, 30 мая — 3 июня 2012г.). М., 2012.
[10] Каневский Е.А., Боярский К.К. Семантико-синтаксический анализатор SEMSIN // Компьютерная лингвистика и интеллектуальные технологии: По материалам ежегодной Международной конференции «Диалог» (Бекасово, 30 мая — 3 июня 2012г.). М., 2012.
[11] Anisimovich K. V., Druzhkin K. Ju., Minlos F. R., Petrova M. A., Selegey V. P., Zuev K. A. Syntactic and semantic parser based on ABBYY Compreno linguistic technologies // Компьютерная лингвистика и интеллектуальные технологии: По материалам ежегодной Международной конференции «Диалог» (Бекасово, 30 мая — 3 июня 2012г.). М., 2012.
[12] Протасов С.В. Преимущества грамматики связей для русского языка // Труды международной конференции «Диалог 2005». М., 2005.
[13] Толдова С.Ю., Соколова Е.Г., Астафьева И., Гарейшина А., Королева А., Привознов Д., Сидорова Е., Тупикина Л., Ляшевская О.Н. Оценка методов автоматического анализа текста 2011–2012: синтаксические парсеры русского языка // Компьютерная лингвистика и интеллектуальные технологии: По материалам ежегодной Международной конференции «Диалог» (Бекасово, 30 мая — 3 июня 2012г.). М., 2012.
[14] Sharoff S., Nivre J. The proper place of men and machines in language technology: Processing Russian without any linguistic knowledge // Компьютерная лингвистика и интеллектуальные технологии: По материалам ежегодной Международной конференции .Диалог. (Бекасово, 25–29 мая 2011 г.). Вып. 10 (17). М., 2011.
[15] Bird S., Klein E., Loper E. Natural Language Processing with Python: Analyzing Text with the Natural Language Toolkit. Beijing, 2009.
[16] Korobov M. Morphological Analyzer and Generator for Russian and Ukrainian Languages. Analysis of Images, Social Networks and Texts: 4th International Conference, AIST 2015, Yekaterinburg, Russia, April 9–11, 2015. Communications in Computer and Information Science, Springer, 2015.

[17] Мальковский М.Г., Старостин А.С. Алгоритм синтаксического анализа, используемый в системе морфо-синтаксического анализа «TREETON» // Труды международной конференции Диалог-2007. М.: изд-во РГГУ, 2007.
[18] Русская грамматика. Т. 2: Синтаксис / Н. Ю. Шведова (гл. ред.). М.: Наука, 1980.
[19] Паничева П.В., Протопопова Е.В., Митрофанова О.А., Мирзагитова А.Р. Разработка лингвистического комплекса для морфологического анализа русскоязычных корпусов текстов на основе Pymorphy и NLTK // Труды международной конференции «Корпусная лингвистика — 2015». СПб., 2015.

Development of the Core for Syntactic Parser for Russian based on NLTK libraries

A. Moskvina, D. Orlova, P. Panicheva, O. Mitrofanova
Saint Petersburg State University

Our project is aimed at the development of the syntactic parser for Russian based on NLTK toolkit for Python. NLTK provides linguistic environment for building formal grammars. We describe a feature-based grammar which allows to analyze the most important syntactic groups within clauses occurring in Russian texts. Our parser operates with rules which include morphological information from the input sentences. The rules are based on the tagset accepted in PyMorphy2 morphological tagger. In the nearest future we plan to enrich our parser so that it could process any well-formed Russian sentences.

Keywords: Natural Language Processing, Syntactic Analysis, Russian, NLTK.