Большакова Е.И., Носков А.А.

Программные средства анализа текстов на основе лексико-синтаксических шаблонов языка LSPL

Введение

Information Extraction (IE) – быстро развивающееся прикладное направление в области компьютерной лингвистики, изучающее задачи извлечения из текстов на естественном языке (ЕЯ) определенной информации – терминов и их отношений, событий и именованных сущностей (имен, персоналий, географических названий) и др. [3, 12]. Решение подобных задач требует распознавания в тексте определенных языковых конструкций (например, именных групп), что реализуется обычно на базе частичного синтаксического анализа, без полного синтаксического разбора предложений текста.

Построение конкретных IE-приложений чаще всего выполняется с использованием специализированных инструментальных систем [1, 4, 7, 9], наиболее известной из которых является система GATE [1]. При этом характерной особенностью является использование особых формальных языков (в системе GATE – язык Jape) для задания информации о составе и грамматических свойствах распознаваемых конструкций, обычно в форме специальных *шаблонов*. С помощью шаблонов уже готовые модули по анализу ЕЯ-текста могут быть настроены для решения нужной задачи обработки текстов в определенной предметной области.

Языки записи шаблонов конструкций, используемые в известных инструментальных системах построения ЕЯ-приложений, имеют ряд недостатков, затрудняющих их эффективное применение. Язык Jape системы GATE универсален, в нем нет встроенных средств задания лингвистической информации и он не позволяет описать важное свойство грамматического согласования элементов ЕЯ-конструкции. В других языках, например, в языке лексических шаблонов отечественной системы Alex [9], возможности описания лингвистических свойств языковых конструкций также ограничены.

Предложенный в работах [5, 6] язык LSPL создавался как декларативный язык спецификации лексических и грамматических свойств выделяемых в текстах конструкций, ориентированный в первую очередь на описание конструкций русского языка. Язык позволяет описывать конструкции в виде *лексико-синтаксических шаблонов*, определяющих входящие в конструкцию слова с учетом их морфологических характеристик и условий их грамматического согласования. Шаблоны подготавливаются лингвистом или специалистом по предметной области анализируемых ЕЯ-текстов;

модификация и настройка шаблонов также может быть выполнена без участия программиста.

Для языка LSPL был разработан метод автоматического выделения в тексте конструкций по их шаблонам [10] и реализованы поддерживающие язык программные средства, включающие средства его интеграции в приложения по автоматической обработке ЕЯ-текстов, а также среду для просмотра и анализа текстов по LSPL-шаблонам.

В данной работе дается краткий обзор известных средств описания и распознавания языковых конструкций в текстах на ЕЯ. Характеризуются выразительные возможности языка LSPL и особенности разработанного метода выделения конструкций по их LSPL-шаблонам. Рассматриваются реализованные для языка программные средства и кратко описывается опыт их применения для создания приложений, требующих анализа ЕЯ-текстов.

Средства описания и выделения языковых конструкций

Широкое распространение инструментальной системы GATE [1] было обусловлено предложенным в ней компонентным подходом к построению приложений по автоматической обработке ЕЯ-текстов, позволяющим «собирать» приложения для конкретных задач из готовых программных модулей анализа текста, а также наличием обширной библиотеки этих модулей и использованием простой модели обрабатываемых данных на базе аннотаций. Аннотации имеют тип, они создаются и перерабатываются программными модулями GATE и представляют собой набор атрибутов, приписываемых к непрерывным фрагментам обрабатываемого текста.

Немалую роль в распространении GATE сыграл язык Jape [2]. Несмотря на применение для обработки ЕЯ-текстов, Jape не содержит лингвистической специфики и представляет из себя язык правил для преобразования произвольной текстовой разметки в форме аннотаций. Правила Jape состоят из двух частей – левой, задающей шаблон, используемый для выделения в тексте некоторого фрагмента текста (языковой конструкции), и правой, описывающей действия, которые должны быть произведены в случае успешного выделения.

В общем случае левая часть содержит типы и атрибуты нужных аннотаций, ограничения на возможные значения атрибутов, а также операторы регулярных выражений (`?`, `*`, `+`, `|`). К примеру, простой шаблон для выделения лексем-чисел `{Token.kind==number}` содержит указание типа аннотации (`Token` – лексема), ее атрибута (`kind` – вид лексемы) и его значения (`number` – число). С помощью метки в шаблоне может быть указана та часть выделяемого фрагмента текста, к которой применяются действия из правой части Jape-правила, обычно они включают добавление новых аннотаций, с заданием их конкретных атрибутов. Пример правила, осуществляющего выделение

фрагмента текста с аннотацией `Lookup` (значение ее атрибута `majorType` равно `jobtitle`) и добавляющего новую аннотацию `Jobtitle` с атрибутом `rule` и со значением `PersonJobTitle`:

```
({Lookup.majorType == jobtitle}) : jobtitle -->
 :jobtitle.Jobtitle = {rule= “PersonJobTitle”}
```

К достоинствам языка Jape относится его универсальность, однако ее следствием является отсутствие встроенных лингвистических атрибутов (лексических, морфологических, синтаксических). Другой его недостаток – невозможность сравнения атрибутов различных аннотаций одной конструкции, что существенно затрудняет запись условия грамматического согласования слов описываемой языковой конструкции. В целом, использование языка Jape для описания выделяемых в ЕЯ-тексте конструкций требует определенной квалификации и опыта, а записанные на этом языке шаблоны языковых конструкций громоздки и плохо читабельны.

В системе RCO Pattern Extractor [7], предназначенной для извлечения информации из текстов на русском языке, язык Jape был расширен средствами задания морфологических характеристик слов выделяемых конструкций (на базе модулей анализа русской морфологии). В шаблонах можно, в частности, описывать конструкции с конкретными лексемами и словами в произвольной форме (например, элемент шаблона для выделения всех словоформ глагола *говорить*: `{Token.text=|“говорить”}`). Тем не менее язык шаблонов системы RCO сохраняет основные недостатки исходного языка Jape, такие как громоздкость записи шаблонов и невозможность прямого задания в них условия синтаксического согласования.

Другим средством описания языковых конструкций являются лексические шаблоны системы Alex [9], разработанной для целей автоматизации лингвистических исследований, а также разнообразных работ с отдельным текстом (составление индекса статьи, проверка терминологии и др.).

В простейшем случае шаблон системы Alex состоит из нескольких лексем (при записи которых окончания заменяются многоточием). В более сложных случаях шаблон описывает несколько вариантов выделяемой конструкции с использованием опциональных лексем (записываются в круглых скобках) и уже определенных вспомогательных шаблонов конструкций (их имена записываются в квадратных скобках). Приведем пример шаблона словосочетания, выражающего в тексте понятие *лесные ресурсы*:

```
[лесные ресурсы]=[лес]V([земля])_лесн…_фонд…
                  V[лесные земли] V лесн…_[ресурс]
```

Как в языке Jape, в шаблонах системы Alex могут быть использованы метки для выделения составных частей распознанных конструкций (например, адресата при выделении полного адреса).

В целом, язык лексических шаблонов системы Alex достаточно лаконичен и нагляден, однако существенным его недостатком является отсутствие средств задания как морфологических характеристик слов (часть речи, падеж, число и др.), так и связи их грамматического согласования.

Рассматриваемый в данной работе язык LSPL позволяет описывать в шаблоне ЕЯ-конструкции как лексические, так и грамматические ее свойства. Важной особенностью языка является возможность явного задания в шаблоне связи грамматического согласования, характерной для многих конструкций русского языка (в первую очередь – именных словосочетаний). В язык были включены выразительные средства, позволяющие достаточно гибко записывать как лексикографические единицы (строки, словоформы, лексемы), так и их основные морфологические характеристики с возможностью их согласования.

Лексико-синтаксические шаблоны языка LSPL

Шаблон языка LSPL задает последовательность элементов-слов, из которых состоит описываемая языковая конструкция, с их полностью или частично конкретизированными морфологическими характеристиками (часть речи, падеж, род, число и т.п.), и позволяет задавать условия их грамматического согласования. К примеру, шаблон `N V<t=pres> Av <N=V>` описывает конструкцию, состоящую из существительного (`N`), следующего за ним глагола (`V`) в прошедшем времени (`t=pres`) и наречия (`Av`), причем существительное и глагол грамматически согласованы (например: *машины работают громко*).

В общем случае для входящего в шаблон элемента-слова могут быть указаны часть речи, конкретная лексема и значения морфологических характеристик: запись `A<простой,n=plur>` задает прилагательное (`A`) *простой* в любой из возможных форм множественного числа: *простые*, *простых*, *простым* и т.п.

Условия согласования задают равенство либо всех, либо конкретных морфологических признаков определенных элементов-слов, например, согласование по числу, как в шаблоне `N V<t=pres> Av <N.n=V.n> .`

Для описания конкретных строк, встречающихся в описываемых конструкциях, например, знаков пунктуации, в шаблоне может быть использована запись вида `";"`.

В шаблон могут включаться и более сложные элементы – повторения и опциональные элементы, которые записываются соответственно в фигурных и квадратных скобках. К примеру, элемент `{N<c=gen>}` обозначает цепочку из нескольких существительных в родительном падеже (вывод записи файла и т.п.), а запись `["не"]`

задает необязательность вхождения частицы *не* в описываемую конструкцию.

Язык LSPL позволяет также записывать в шаблоне несколько альтернатив, соответствующих различным вариантам описываемой языковой конструкции. Например, шаблон `AP = A|Pa` описывает адъектив – прилагательное или причастие.

LSPL-шаблон может иметь имя и параметры; параметры записываются в круглых скобках и фиксируют некоторые морфологические характеристики описываемой конструкции. Например, шаблон `NP = {A} N1 <A=N1> [N2<c=gen>] (N1)` определяет именную группу `NP` из нескольких прилагательных, согласованного с ними существительного и опционального существительного в родительном падеже (*верный ответ*, *короткий интервал времени*, *удаленный банковский терминал*). В этот шаблон входят два существительных (`N1` и `N2`), для их различения используются числовые индексы. Параметрами шаблона являются морфологические характеристики первого существительного.

Язык позволяет использовать уже определенные шаблоны для задания шаблонов более сложных языковых конструкций, и при этом можно использовать параметры известных шаблонов для конкретизации их элементов или для согласования элементов конструкции. К примеру, указанный выше шаблон именной группы `NP` можно применить для задания частных случаев именной группы, когда первое ее существительное употребляется во множественном числе: `NP <n=plur>`. Другой пример – описание конструкции, включающей эту группу и согласованный с ней глагол в прошедшем времени: `NP V<t=past> <NP=V>` (*внешнее условие проверялось*).

В целом, язык LSPL является достаточно гибким средством описания лексических и грамматических свойств конструкций русского языка, для целей их распознавания в тексте.

Выделение языковых конструкций по LSPL-шаблонам

При выделении конструкций заданные шаблоны последовательно накладываются на текст, образуя так называемые *варианты наложения*, соответствующие различным случаям употребления в тексте конструкций, описанных этими шаблонами.

Метод выделения использует представление текста в виде графа, в котором ребра соответствуют различным *синтаксическим интерпретациям* фрагментов текста, включающих слова, а вершины соответствуют фрагментам, не содержащим слов (в частности, пробельным символам).

При построении внутреннего представления текста сначала осуществляется его разбиение на фрагменты — слова, знаки препинания и пробельные символы, и выполняется морфологический

анализ слов, морфологические (синтаксические) интерпретации которых образуют ребра графа. При этом в общем случае пара соседних вершин графа соединена несколькими ребрами вследствие морфологической омонимии (например, именительного и винительного падежа существительных). Пример такой ситуации приведен на Рис. 1: у слова *маленький* и *домишко* по две морфологических интерпретации, соответствующих именительному и винительному падежам (ребра a, b и c, d), а у слова *высокого* – три: именительный или винительный падеж мужского рода, родительный падеж среднего рода (ребра h, i , j).

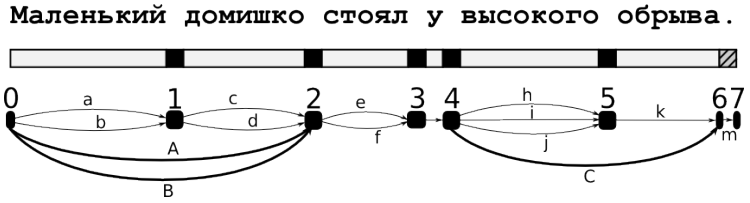

Рис. 1. Граф текста с вариантами наложения шаблона A N <A=N>(N)

Для любой выделенной в тексте конструкции ее синтаксическая интерпретация представляется в графе текста дополнительным ребром. На Рис. 1 ребра A, B и C представляют интерпретации фрагментов текста, выделенных по шаблону согласованной именной группы из прилагательного и существительного A N <A=N> (N). Опять же, одна и та же пара вершин может быть соединена несколькими ребрами (например, A и B), представляющих разные синтаксические интерпретации одного и того же фрагмента *маленький домишко* (они отличаются значениями морфологических характеристик входящих в него слов). Таким образом, из-за морфологической омонимии возможно несколько вариантов наложения шаблона на один и тот же отрезок текста.

Рассмотренное графовое представление текста удобно для учета различных сочетаний морфологических интерпретаций входящих в текст слов и позволяет при наложении шаблонов единообразно обрабатывать как элементы-слова, так и вспомогательные шаблоны.

Выделение конструкции по LSPL-шаблону представляет из себя поиск в графе текста, который включает три основных этапа:

1. Определяется множество ребер, с которых может быть начат поиск, для этого применяются индексы слов, шаблонов и частей речи (эти индексы строятся одновременно с построением графа текста).
2. В графе ищутся пути, начинающиеся с найденных ребер и соответствующие последовательности элементов шаблона. Поиск этих путей представляет обход графа в глубину с откатом назад,

при этом на каждом шаге рассматриваются все допустимые продолжения пути, соответствующие текущему элементу шаблона и заданным для него ограничениям на морфологические характеристики. Условия согласования элементов проверяются сразу после конкретизации морфологических характеристик, что позволяет сократить множество рассматриваемых путей в графе.

3. Найденные пути группируются и образуют варианты наложения, добавляемые в граф в виде новых ребер. Группировка путей проводится для более эффективной работы метода.

Свойственная естественному языку множественность синтаксических интерпретаций одного отрезка текста неизбежно увеличивает пространство поиска и объем памяти, необходимой для хранения графа текста. Это особенно заметно, когда шаблон содержит большое число элементов, не участвующих в условиях согласования, а также включает повторения элементов без их согласования.

Для сокращения поиска предполагается, что при анализе по шаблонам важны только те морфологические характеристики элементов, которые будут использованы в объемлющей конструкции или вошли в число параметров самого шаблона. Все синтаксические интерпретации одного фрагмента текста, различающиеся только характеристиками, не входящими в число выходных параметров, группируются и не различаются в ходе дальнейшего поиска на графе, что позволяет значительно уменьшить количество рассматриваемых вариантов наложения. Такой подход позволяет гибко управлять количеством возможных вариантов наложения шаблона на текст путем изменения набора его выходных параметров.

Состав и архитектура программных средств

Основное назначение реализованных программных средств – автоматический поиск в заданном тексте на русском языке фрагментов, представляющих варианты наложения на текст заданных LSPL-шаблонов (т.е. выделение всех вхождений в текст конструкций, описанных этими шаблонами). Поскольку выделенные языковые конструкции часто требуют некоторой дальнейшей обработки для решения конкретных прикладных задач, дополнительно был реализован механизм задания *преобразований* найденных вариантов наложения. Такие преобразования могут включать нормализацию слов выделенной конструкции, генерацию новых шаблонов из элементов выделенной конструкции, перевод выделенных конструкций в структуру данных, требуемую прикладной задачей.

Разработанные программные средства включают:

- Ядро, реализованное на языке C++ и осуществляющее выделение языковых конструкций по LSPL-шаблонам и их преобразование;
- Консольные утилиты, предоставляющие доступ к функциям ядра;

- Адаптер для использования ядра в языке программирования Java, реализованный на основе технологии JNI;
- Среда просмотра и анализа текстов, реализованная в виде приложения на базе Eclipse RCP с использованием Java-адаптера;
- Средства интеграции для IDE Eclipse, служащие для быстрого прототипирования Java-приложений с использованием языка LSPL.

Для выполнениияния графематического и морфологического анализа в ядре предусмотрены подключаемые модули (в настоящий момент используются анализаторы АОТ [11]). Допускается также подключение внешних словарей (например, словаря синонимов).

Главная функция **среды просмотра и анализа текстов** – задание LSPL-шаблонов конструкций и визуализация результатов поиска по заданным шаблонам в различных текстах. Основными пользователями среды являются специалисты, участвующие в разработке LSPL-шаблонов, которым необходимо анализировать тексты и тестировать создаваемые шаблоны. Возможности среды охватывают:

- Загрузку текстов различных форматов (форматы Microsoft Office, XML, HTML, а также простой неразмеченный текст);
- Ввод новых шаблонов или загрузка уже существующих из файла, при этом происходит проверка синтаксической корректности шаблонов и вывод сообщений о найденных ошибках;
- Поиск и выделение в загруженном тексте конструкций по заданным LSPL-шаблонам; подсчет статистики употребления конструкций;
- Просмотр информации о выделенных конструкциях и найденных при этом вариантах наложения шаблонов.

Пользовательский интерфейс среды – см. Рис. 2 – представляет собой окно с тремя рабочими областями: для просмотра анализируемого текста, для ввода LSPL-шаблонов и для просмотра выделенных по шаблонам конструкций.

В области просмотра текста подсвечиваются фрагменты, соответствующие выделенным по шаблонам конструкциям; при наведении курсора на такой фрагмент показывается информация о его синтаксических интерпретациях.

В области шаблонов визуализируются шаблоны, применяемые для выделения конструкций; внизу области содержится поле ввода нового шаблона; для каждого шаблона может быть выведена статистика его наложения на текст.

В области просмотра выделенных конструкций отображаются синтаксические интерпретации всех найденных вариантов наложения шаблонов, сгруппированные по фрагментам текста; также может быть выведен список всех слов текста с фильтрацией по частям речи.

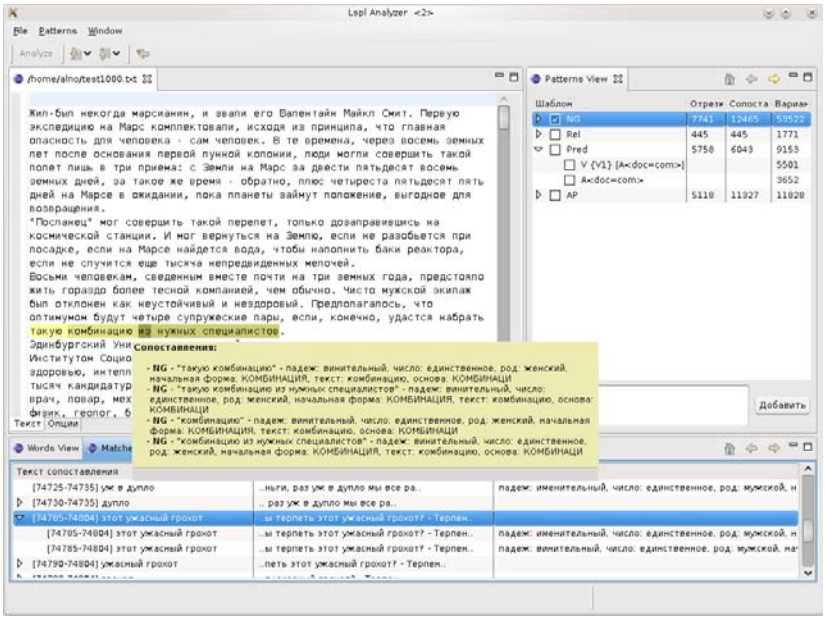

Рис. 2. Среда просмотра и анализа текстов

Для ускорения прототипирования и разработки приложений с использованием языка шаблонов LSPL и языка программирования Java были дополнительно разработаны **средства интеграции** для популярной среды разработки на языке Java – Eclipse. Эти средства включают автоматическую загрузку динамических библиотек (зависящих от операционной системы) и архивов со словарями (требуемых для морфологического и графематического анализа), а также их автоматизированное подключение к разрабатываемому приложению путем выбора соответствующего пункта в опциях проекта.

К разработанным средствам интеграции относятся и средства редактирования и отладки LSPL-шаблонов непосредственно в IDE Eclipse. Обычно отладка шаблонов производится с использованием консольных утилит или среды просмотра и анализа текстов, однако это требует их постоянного копирования между средой анализа и IDE, что приводит к потерям во времени. Разработанные средства позволяют проводить анализ результатов наложения шаблонов на различные тексты непосредственно в редакторе с использованием интерфейса, схожего со средой просмотра и анализа текстов. Среди особенностей этого интерфейса стоит отметить анализ текста, производящийся автоматически при его изменениях или же изменениях шаблонов.

Разработанные приложения

Реализованные программные средства были опробованы при создании нескольких приложений, связанных с автоматической обработкой ЕЯ-текста. Первым приложением был автоматический терминологический анализ русскоязычных научно-технических текстов [6, 8]. С помощью языка LSPL были формально описаны:

- синтаксические образцы терминологических словосочетаний, употребляемых в научно-технических текстах (именные группы разной структуры);
- словарные термины и их синонимы (из словарей по физике и информатике (*вектор намагниченности*, *вектор Умова* и т.п.);
- регулярно используемые в научно-технических текстах фразы-определения новых (авторских) терминов и фразы введения терминологических синонимов (*Эту проблему назовем проблемой скрытого состояния* и т.п.);
- правила образования лексико-синтаксических вариантов терминов (*библиотека стандартных программ – библиотека программ*, *коллекция текстов – текстовая коллекция* и т.п.);
- правила образования в тексте соединений нескольких терминологических словосочетаний (*шина адреса*, *шина данных*, *шина управления – шина адреса, данных и управления* и т.п.).

Для каждой группы полученных шаблонов на базе реализованных средств поддержки языка LSPL были построены и экспериментально исследованы процедуры выявления в научно-техническом тесте терминов разного вида. На этой основе была предложена стратегия объединения их результатов, улучшающая показатели точности и полноты распознавания терминов в тесте.

Другим приложением, разработанным с использованием языка LSPL, была вопросно-ответная система, применяющая для поиска ответа логический вывод (метод резолюций). Исходные утверждения о сущностях предметной области, как и сами вопросы о них формулируются пользователем в виде предложений русского языка, система переводит их в формулы логики первого порядка. Примеры утверждений и вопросов: *Если книга по искусству большая, она дорогая. Есть ли большие недорогие книги?*

Синтаксический анализ входных предложений проводится на основе набора LSPL-шаблонов, в который входят шаблоны именований сущностей, шаблоны описания их свойств, шаблоны утверждений и шаблоны вопросов. Для перевода ЕЯ-предложений в формулы применяется реализованный (в составе программных средств поддержки языка LSPL) механизм преобразований найденных вариантов наложений шаблонов, который позволяет описать правила

преобразований непосредственно в шаблонах и тем самым отделить их от программного кода приложения (вопросно-ответной системы).

Еще одним примером приложения, разработанного с помощью LSPL-шаблонов, является модуль автоматической генерации тестов для программного кода на языке Java с использованием комментариев к коду, написанных на естественном языке. Генерация основана на предположении, что комментарии к программному коду часто содержат полуформальное описание требуемого поведения соответствующих элементов кода (переменных, функций, методов, классов и др.), которое может быть использовано для построения программных тестов.

Реализованный (в виде плагина для IDE Eclipse) модуль выполняет генерацию JUnit-тестов (процедур автоматизированного тестирования) по комментариям, написанным для системы автоматической генерации документации Javadoc. Укажем основные этапы генерации процедур тестирования:

1. С помощью LSPL-шаблонов именных групп из Javadoc-комментариев извлекаются возможные именования элементов программного кода, (например: *первое число*, *результат*, *значение факториала*).
2. Для каждого извлеченного именования формируется набор LSPL-шаблонов для выделения его употреблений в тексте (применяется механизм преобразований конструкций, выделенных на этапе 1).
3. На основе сформированных шаблонов именований строятся шаблоны конструкций, описывающих различные аспекты поведения кода (взаимосвязь параметров функции и ее результата, условия возникновения исключений и т.п.).
4. Построенные шаблоны используются для выделения из текста комментариев набора ограничений, описывающих поведение кода (например, ограничение на значение функции), а из них – набора проверяемых в тестах условий, по которым генерируется код процедур тестирования.

Заключение

В работе описаны программные средства, разработанные в поддержку языка лексико-синтаксических шаблонов LSPL, служащего для описания языковых конструкций в системах автоматического извлечения и анализа информации из текстов на русском языке. Средства были успешно опробованы при создании трех различных приложений, требующих анализа текста на естественном языке.

Литература

1. Bontcheva K., et al. GATE: A Unicode-based infrastructure supporting multilingual information extraction. In: Proceedings of Workshop on Information Extraction for Slavonic and Other Central and Eastern European Languages (IESL'03), Borovets, 2003.

2. Cunningham H., et al. JAPE: a Java Annotation Patterns Engine. (Second Edition). Technical report CS--00--10, University of Sheffield, Department of Computer Science, 2000.
3. Grishman R. Information extraction. In: The Oxford Handbook of Computational Linguistics. Mitkov R. (ed.). Oxford University Press, 2003. p. 545-59.
4. Petasis G., et al. Ellogon: A New Text Engineering Platform // Proc. of the Third International Conference on Language Resources and Evaluation (LREC 2002). Las Palmas, 2002, p. 72-78.
5. Большакова Е.И., Баева Н.В., Бордаченкова Е.А., Васильева Н.Э., Морозов С.С. Лексико-синтаксические шаблоны в задачах автоматической обработки текстов // Компьютерная лингвистика и интеллектуальные технологии: Труды Межд. конференции Диалог ‘2007. – М.: Издательский центр РГГУ, 2007, с.70-75.
6. Большакова Е.И., Васильева Н.Э. Формализация лексико-синтаксической информации для распознавания регулярных конструкций естественного языка. // Программные продукты и системы, 2008, № 4 , с. 103-106.
7. Ермаков А.Е. и др. RCO Pattern Extractor: компонент выделения особых объектов в тексте // Информатизация и информационная безопасность правоохранительных органов: XI Межд. научная конференция. Сборник трудов – Москва, 2003. с. 312-317.
8. Ефремова Н.Э., Большакова Е.И., Носков А.А., Антонов В.Ю. Терминологический анализ текста на основе лексико-синтаксических шаблонов // Комп. лингвистика и интеллектуальные технологии: По материалам Междунар. конф. «Диалог» (Бекасово, 26-30 мая 2010 г.) Вып. 9 (16). – М.: Изд-во РГГУ, 2010, с. 124-129.
9. Жигалов В.А. и др. Система Alex как средство для многоцелевой автоматизированной обработки текстов // Труды Международного семинара Диалог’2002: "Компьютерная лингвистика и интеллектуальные технологии". М.: Наука, 2002. Т.2, С.192-208.
10. Носков А.А. Метод выделения в тексте конструкций по их лексико-синтаксическим шаблонам // Сборник статей молодых ученых факта ВМиК МГУ – М.: МАКС Пресс, 2009, Выпуск 6, с. 136-145.
11. Сокирко А. В. Морфологические модули на сайте www.aot.ru // Труды международной конференции Диалог’2004: Компьютерная лингвистика и интеллект. технологии. – М.: Наука, 2004, с. 559.
12. Хорошевский В.Ф. OntosMiner: семейство систем извлечения информации из мультиязычных коллекций документов // Девятая Национальная конференция по искусственному интеллекту КИИ-2004. Т. 2. – М.: Физматлит, 2004, с. 573-581.