УДК 004.4'413

СИНТАКСИЧЕСКИЙ АНАЛИЗАТОР ДЛЯ ВОПРОСНО-ОТВЕТНОЙ СИСТЕМЫ

К.Х. Ким, А.П. Савинов

Томский политехнический университет
E-mail: kimkir@sibmail.com

Рассмотрена программная реализация в среде Borland Delphi 7 алгоритма синтаксического анализатора текстов на естественном русском языке. За основу взята лингвистическая методология проведения глубокого синтаксического анализа И.А. Мельчука. Приведен анализ достоинств и недостатков этого подхода с точки зрения программирования. Показано, что внедрение программного блока глубокого синтаксического анализа расширит возможности вопросно-ответной системы.

Ключевые слова:
Лингвистический процессор, автоматический синтаксический анализ, вопросно-ответные системы, интерфейс на естественном языке, синтаксическая структура предложения.

Key words:
Linguistic processor, automatic syntax analysis, question-answer systems, natural language interface, syntax structure of the sentence, generative grammars, syntax, parser.

Введение

Проблема создания интеллектуального человеко-машинного интерфейса остается актуальной и по сей день. Между собой люди общаются с помощью речи на естественном языке или письменного текста. Для обеспечения массового использования компьютеров населением необходимо, чтобы взаимодействие неподготовленного пользователя с ЭВМ также происходило на естественном языке в виде письменного текста или речи [1–3].

Вопросно-ответные системы, воспринимающие текст на естественном языке, являются одним из направлений развития искусственного интеллекта.

Под вопросно-ответными системами понимаются системы, способные принять запрос пользователя на естественном русском языке, организовать поиск требуемой информации в базе знаний и выдать ответ пользователю на том же языке.

Чтобы понять смысл запроса, компьютер должен осуществить обработку текста:

- графематическую;
- морфологическую;
- синтаксическую;
- семантическую.

Графематический анализ является подготовительным этапом для морфологического анализа. Входной текст разбивается на отдельные словоформы, программа запоминает расположения словоформ в тексте, распознает косвенные элементы текста (номера телефонов, наименования организаций, Ф.И.О., даты и т. д.).

Целью морфологического анализа является определение всех вариантов грамматических характеристик (род, число, падеж, часть речи) к каждой словоформе входного текста.

Синтаксический анализ определяет наличие и характер синтаксических связей между словоформами, создает синтаксическую структуру текста.

Семантический анализ создает структуру входного текста.

На сегодняшний день графематический и морфологический этапы реализованы в ряде систем и успешно работают. Основное внимание сейчас уделяется синтаксическому и семантическому анализу. Необходимо реализовать глубокий синтаксический анализ, способный обрабатывать тексты на естественном языке без ограничений в предметной области.

Исследования в формальной лингвистике можно условно разделить на два подхода: построение универсальной грамматики, верной для всех существующих языков мира, и построение формальной модели, наиболее полно охватывающей все множество грамматических явлений конкретного языка.

Н. Хомский стал родоначальником первого подхода и основателем школы генеративистов. Самым ярким представителем второго подхода является И.А. Мельчук, автор модели «Смысл – Текст» [1, 2].

На сегодняшний день наибольшие успехи наблюдаются в английском языке, в котором уже существует ряд систем, способных успешно производить синтаксический разбор предложений. Поскольку для английского языка характерен прямой порядок слов в предложении, для осуществления синтаксического анализа наиболее подходят грамматики составляющих, разработанные американским ученым Н. Хомским [1].

Для русского языка характерна независимость состава слов и порядка их следования в предложении, поэтому для него наиболее приемлемо использование на синтаксическом уровне грамматики зависимостей.

Лингвистический алгоритм автоматического синтаксического анализа текста на естественном языке, реализующий грамматики зависимостей, был детально разработан И.А. Мельчуком в середине 60-х гг. прошлого века [2]. Особенностью предложенного подхода является стремление сделать полный синтаксический разбор предложения. Однако вычислительная мощность компьютеров не позволяла тогда реализовать сложные алгоритмы анализа в полном объеме. Упрощение алгоритмов и отказ от перебора омонимичных вариантов – компромисс, который приводил к низкой точности синтаксического анализа предложения.

Наиболее популярным стало применение статистических (вероятностных) алгоритмов синтаксического анализа. Главная особенность подобных методов анализа – отсутствие жесткой системы синтаксических правил, для создания которой, собственно, и требовалось участие лингвиста.

Вместо системы синтаксических правил используется обширный набор примеров-предложений, разобранных человеком вручную (или полуавтоматически, но с обязательным контролем человека-оператора). Этот набор примеров используется для «обучения» статистического распознавателя, опирающегося на известный метод дерева принятия решений [3].

Для создания набора примеров не требуется высокой квалификации в области лингвистики. Проблема только в количестве этих примеров – для достижения приемлемой точности анализатора их может потребоваться тысячи.

Разновидностью статистических систем синтаксического анализа являются анализаторы, которые используют описание языка в виде моделей управления. Они настроены на работу в требуемой предметной области и полученные в результате предварительного анализа корпуса текстов этой предметной области. Каждой модели управления приписывается частотность, характеризующая вероятность использования этой модели управления для новых текстов данной области [4].

Наблюдаемый сегодня технический прогресс в развитии вычислительной техники, а также необходимость создания универсального лингвистического процессора, позволяющего работать с достаточно широкой областью знаний, привели к необходимости вернуться к концепции создания синтаксических процессоров на основе применения лингвистических правил грамматики, описывающих предметную область.

Конечной целью работы является программная реализация вопросно-ответной системы, способной строить синтаксическую структуру входного текста, и на основании этой структуры отвечать на запросы пользователя, сформулированные на естественном русском языке. В отличие от большинства систем по анализу текстов на естественном языке, вопросно-ответная система с глубоким синтаксическим анализом не будет ограничиваться одной предметной областью (медициной, юриспруденцией, бухгалтерией и т. д.) или тематикой входных текстов и не будет привязана к узкоспециализированной терминологии, понятиям и терминам. Это является существенным достоинством системы, которое расширяет области ее применения для обработки текстов на естественном русском языке.

1. Общие принципы внутрисегментного синтаксического анализа

В основу работы синтаксического анализатора был положен лингвистический алгоритм синтаксического внутрисегментного анализа текста на естественном языке И.А. Мельчука.

Внутрисегментный автоматический синтаксический анализ занимается определением синтаксических связей между словами одного простого предложения (сегмента, фрагмента) или предложения в составе сложносочиненного или сложноподчиненного предложения.

Все словосочетания рассматриваются как сочетания из двух слов (синтагм), где одно слово – «главное», другое – «зависимое». «Главное» и «зависимое» слова связаны друг с другом определенным видом связи. Существуют 33 типа синтаксических связей (ОНД – Отношения Непосредственной Доминации).

Слова в предложении согласованы между собой по средством морфологических признаков (род, число, падеж). Если рассматривать значения грамматических признаков слов, как некоторую конфигурацию, то каждое словосочетание имеет определенную конфигурацию грамматических признаков. Чтобы не приходилось каждый раз вести поиск по всем типам синтаксических связей и сопоставлять грамматические признаки для определения типа связи, была создана специальная таблица – таблица конфигураций. Каждая строка таблицы конфигураций представляет собой отдельное правило синтаксиса.

Алгоритм анализа построен таким образом, что как только начинается работа с конкретной конфигурацией, то он прорабатывается максимально подробно. Все алгоритмы модели синтаксиса формализованы для реализации в виде программы.

До начала основной обработки текста имеет место подготовительный этап. Слова и их морфологические признаки переносятся в специальную таблицу информаций, которую будет использовать в своей работе алгоритм синтаксического анализа. Кроме полей с морфологическими признаками, в ней предусмотрены поля для записи синтаксической связи и типа этой связи. Формально, для фиксирования синтаксической связи между словами алгоритм записывает «главному» слову, какие слова зависят от него, а «зависимым» указывается «главное».

Когда этап предварительной обработки завершен, алгоритм переходит к синтаксической обработке входных данных и начинает рассматривать слова первого простого предложения (сегмента, фрагмента). Для того, чтобы определить, какое из слов «главное», а какое «зависимое», алгоритм поочередно берет каждое слово и рассматривает его как «главное», далее ищет в пределах простого предложения «зависимое» слово.

Как только задано «главное» слово, остальные слова в предложении становятся потенциальными «зависимыми». Параметры «зависимого слова» алгоритм смотрит в таблице конфигураций. Записи в таблице конфигураций сортированы по «главным» словам. Как только все синтаксические конструкции с данным «главным» словом рассмотрены, алгоритм переходит к следующему по порядку слову и так, пока не закончится сегмент.

Некоторые синтаксические связи могут быть однозначно установлены только после того, как установлены более простые или очевидные связи. Для этого синтаксический анализ сегмента разделен на 5 циклов. На каждом цикле рассматриваются только те конфигурации, которые предписаны соответствующему циклу так, как будто других конфигураций не существует.

Принцип распределения конфигураций по циклам таков. Группа конфигураций, которые рассматриваются на первом цикле, – это конфигурации, которые не зависят от других конфигураций. Конфигурации, которые рассматриваются на втором цикле – это такие конфигурации, которые зависят хотя бы от одной конфигурации первого цикла и т. д.

2. Описание методов реализации внутрисегментного синтаксического анализа в среде Delphi 7

Если рассматривать внутрисегментный автоматический синтаксический анализ текста с точки зрения его реализации в виде программы, то можно сказать, что система использует 2 типа данных:

Статические данные и алгоритмы или данные в виде таблиц – это морфологические признаки словоформ (в таблице информаций) и словарь правил синтаксиса (в виде таблицы конфигураций). Эти данные реализованы в виде отдельных таблиц.

Часть полей в таблице информаций являются пустыми. Они заполняются в процессе анализа. Поля таблицы содержат следующую информацию: морфологические признаки слов, наличие подчиненных слов, наличие главного слова, тип синтаксической связи, вхождение в именную или предложную группу и т. д.

Каждая запись в таблице конфигураций содержит информацию о морфологических признаках главного и зависимого слов, о типе синтаксической связи между ними и об указаниях на обработку в других программных блоках.

Алгоритмы для обработки данных – это программные блоки, на которые разделена вся система автоматического синтаксического анализа. Эти алгоритмы реализованы в виде программных процедур. Алгоритмы используют данные из таблицы конфигураций как параметры для процедур (место поиска, предмет поиска, параметры поиска, диапазон поиска). Процедуры разделяются согласно подходу внутрисегментного анализа на 5 блоков:

- основной алгоритм;
- проверка на дополнительные условия;
- дополнительные обработки;
- обработка исключений (действия, если не найден второй член конфигурации);
- основная обработка.

В описании автоматического внутрисегментного синтаксического анализа И.А. Мельчука каждый блок представлен в виде списка операций. Все операции в блоке сгруппированы под заголовками, которые рассматривают конкретный случай в синтаксической грамматике (составные предлоги «...так, как...», «последовательности числительных», и т. д.). Каждая операция описывается текстом и примерами (рис. 1).

В среде *Delphi* 7 была сохранена структура представления алгоритма. Заголовки таких операций были реализованы в виде названий классов, а операции внутри них упорядочены по шагам. Краткие описания операций сохранены в виде комментариев в коде программы. Все 5 блоков автоматического внутрисегментного анализа имеют одинаковую структуру. Основной алгоритм взаимодействует с остальными блоками по принципу «ссылок», блок отправляет данные на дальнейшую обработку, всегда с указанием номера процедуры и блока.

Рассмотрим на примере реализации Блока исключений (действия, если не найден второй член конфигурации). Блок состоит из 58 классов, каждый из которых является наследником ***TProcClass***. Каждый класс-наследник состоит из набора последовательных процедур, предназначенных для разрешения определенной исключительной ситуации. Это может быть проверкой или записью значений в таблицу информаций или сегментов. Класс-родитель содержит вспомогательные процедуры, общие для классов-наследников, такие как изменение индексов в таблице морфологических признаков при добавлении/удалении строки.

Для того, чтобы остальным блокам было «понятно», от какого класса необходимо создать наследника, названия классов соответствуют номеру, указанному в таблице конфигураций.

Например, если в таблице конфигурации требуется обращение к процедуре номер 5 блока исключений, класс будет иметь название *TIfNotSecMem5* (*If Not Second Member* – если не найден второй член конфигурации). Каждый класс содержит определенный набор последовательных процедур, назы-

§ 5. ЧАСТЬ Д – ДЕЙСТВИЯ В СЛУЧАЕ, ЕСЛИ НЕ НАЙДЕН ВТОРОЙ ЧЛЕН КОНФИГУРАЦИИ

Для составных предлогов, последним компонентом которых является предлог (КНФ 15–20)

1. Проверить, является ли словоформа первая вправо от ***i0***, словоформой *друг*.

2. Обозначить словоформу, первую вправо после друг, через ***i1***.

Имеются в виду случаи, типа *в зависимости друг от друга* // *по отношению друг к другу* и т. д., где словоформа *друг* отделяет последний компонент составного предлога.

Рис. 1. *Пример описания алгоритма автоматического синтаксического анализа из книги И. А. Мельчука. i_0 – главное слово; i_1 – зависимое слово; КНФ – конфигурация*

ваемых шагами: *Step*1, *Step*2 и т. д. Такая структура объясняется наличием случаев, когда из одного класса происходит обращение к определенному шагу другого класса. На рис. 2 приводится пример структуры одного класса.

В качестве параметров процедурам передается номер текущего слова (индекс *i*0), и два индекса *i*1 и *i*2, связанные с данным словом, а также номер текущей конфигурации. Эти переменные не являются глобальными в связи с тем, что передача их как параметров по ссылке исключит возможные ошибки и поможет отследить места их изменения.

Для отслеживания передаваемых параметров и вызываемых блоков в процессе работы системы создается файл «*journal.txt*». Каждая вызываемая процедура записывает в него свои координаты и значения полученных параметров. Этот способ контроля над ходом работы блока позволяет наглядно отображать, как обрабатывалась конфигурация, на какой шаг она перешла, на каком этапе работы возникла ошибка.

Таблицы морфологических характеристик, конфигураций и сегментов реализованы в виде отдельных таблиц базы данных. Такой способ представления оказался наиболее приемлемым, т. к. он обеспечивает быстрый доступ к данным и их редактированию. Чтобы работа с базой данных не влияла на скорость работы системы, количество таблиц было минимизировано до трех. Для обеспечения наглядности таблиц и их содержимого, было решено заменить числовые значения полей таблицы на символьные. Это позволило решить проблему индексации таблиц в базе данных. В противном случае произошло бы значительное замедление работы программы.

Создание таблиц БД осуществляется с помощью утилиты Database Desktop (DBD), входящей в комплект поставки Delphi. Утилита DBD решает целый ряд задач, связанных с таблицами. С ее помощью можно создавать или изменять структуру таблицы, строить ее первичные ключи и индексы, создавать и изменять записи, просматривая их и т. д. Для хранения структуры таблиц была выбрана база данных Paradox 7.

Для интерпретации данных из базы используется компонент TTable из панели BDE. В среде Borland Delphi существует набор встроенных функций для обращения к данным в таблице TTable. Использование этих функций привело к уменьшению громоздкости кода и к исключению ошибок, связанных с написанием запросов к таблицам.

Интерфейс построения дерева зависимостей

Для наглядного отображения результатов анализа спроектировано отдельное приложение. Результат работы синтаксического анализа в виде дерева зависимостей предложения выводится на отдельную форму. Словоформы предложения с пронумерованными стрелками от главной словоформы к зависимой словоформе отображаются в виде графика с расшифровкой номеров стрелок, в виде наименований типов синтаксической связи.

Дополнительная форма, с информацией о ходе анализа в реальном масштабе времени. На этой форме отображается информация о переходе процесса обработки с одного блока в другой, сообщения об ошибках, отображение обработанных информаций и их синтаксических связей, ход выполнения анализа, время, затраченное на обработку,

```
Procedure TIfNotSecMem5 (var i0, i1, i2 knf: integer)
    Begin
     If (некоторое условие) then TIfNotSecMem35.Step3 (var i0, i1, i2 knf: integer);
    End;
```

Рис. 2. *Структура программной реализации одной из обработок алгоритма синтаксического анализа*

```
//-------------------------------------Класс1 TKNF15_20-----------------------------------

 1. Проверить, является ли словоформа, первая вправо от I0, словоформой "друг"
 2. Если да, то обозначить словоформу, первую вправо после "друг" через I1
-------------------------------------------------------------------------------------------------}

 procedure TIfNotSecMem1.Step1();
 begin
        ShowMessage ('блок IfNotSecMem1 Step1');
// ShowMessage('FormSystem.TableInformation.FieldByName('SymbolsWord').AsString');
         FormSystem.TableInformation.Next;
         if (FormSytem.TableInformation.FieldByName('SymbolsWord').AsString= 'друг')
        then Step2
        else begin
        I0 := I0+1; // следующая словоформа, переход к A.1.4
        ShowMessage ('Если не найден 2-ой член 15_20.Step1->переход к A.1.4');
 end;
 end;

 procedure TIfNotSecMem1.Step2();
 begin
        ShowMessage ('Если не найден 2-ой член 15_20.Step2');
        FormSystem.TableInformation.Next;
         i1:= FormSystem.TableInformation.FieldByName('WordID').AsInteger;
 // переход к блоку G
        ShowMessage('Если не найден 2-ой член 15_20.Step1->переход к G9');
 BusyComb_V.Step1;
 BusyComb_V.Free;
 end;
//-------------------------------------Конец Класс1 TKNF15  20----------------------------
```

Рис. 3. *Пример программирования одного из блоков алгоритма синтаксического анализа*

время, оставшееся до завершения обработки, номер текущего цикла анализа.

Программный блок ведения статистики

Процедуры блока статистики осуществляют сбор статистических данных. Какие типы синтаксических конструкций часто встречаются, при каких условиях (контекст), в каких программных блоках возникают ошибки – всю эту информацию упорядочивает, сортирует и классифицирует блок ведения статистики. Целью работы блока является оптимизация работы блока автоматического синтаксического анализа и сокращение времени обработки часто встречаемых синтаксических структур.

Заключение

Рассмотрена программная реализация в среде Borland Delphi 7 алгоритма синтаксического анализатора текстов на естественном русском языке. Программно реализованы блоки для дополнительных обработок и исключений. Программные блоки проверки на дополнительное условие, основных обработок и основной алгоритм реализованы примерно на 50 %. В алгоритм И.А. Мельчука и структуру данных внесены существенные доработки: полностью исключен принцип «переадресации»; в структуру таблиц морфологических признаков (информаций к словоформам) и таблицы конфигураций добавлены новые поля. Все поля, связанные с переадресацией, исключены. Изменения были необходимы из-за

различия в принципе построения морфологической информации к словам. В алгоритме И.А. Мельчука используется словарь основ и флексий, а в разрабатываемом синтаксическом анализаторе используется словарь с восстановленной парадигмой слов.

Разработан и программно реализован журнал операций для тестирования работоспособности блоков автоматического синтаксического анализа, так как осуществление тестирования программных блоков по отдельности очень сложно. Выбраны методы программирования алгоритмов и способы хранения данных, задействованных в работе программы.

СПИСОК ЛИТЕРАТУРЫ

1. Большаков И.А., Гельбух А.Ф. Модель «Смысл – Текст» тридцать лет спустя // Диалог 99: Труды Междунар. семинара. – М., 1999 – Т. 1. – С. 15–24.
2. Мельчук И.А. Автоматический синтаксический анализ. – Новосибирск: Изд-во АН СССР, 1965. – 358 с.
3. Андреев А.М., Березкин Д.В., Брик А.В., Кантонистов Ю.А. Вероятностный синтаксический анализатор для информационно-поисковой системы [Электронный ресурс]. – режим доступа: http://www.inteltec.ru/publish/articles/textan/1kx5_9.shtml. – 16.03.2006.
4. Волкова И.А., Мальковский М.Г., Одинцев Н.В. Адаптивный Синтаксический анализатор // Диалог 2003: Труды Междунар. семинара. – М., 2003. – Т. 1. – С. 401–406.

Поступила 09.04.2009 г.

УДК 681.3.068+681.5

ВЫЯВЛЕНИЕ СКРЫТЫХ ЗАКОНОМЕРНОСТЕЙ В СЛОЖНЫХ СИСТЕМАХ

О.Г. Берестнева, Я.С. Пеккер

Томский политехнический университет
Сибирский государственный медицинский университет
E-mail: ogb@tpu.ru; pekker@ssmu.ru

Рассмотрены основные подходы к решению задачи выявления скрытых закономерностей в сложных медико-биологических системах. Показаны особенности решения такой задачи в случае количественных и качественных экспериментальных данных. Представлена технология выявления скрытых закономерностей на основе методов Data Mining.

Ключевые слова:
Компьютерный анализ данных, скрытые закономерности, биосистемы, деревья решений.

Key words:
Computer analysis of data, hidden patterns, biological systems, decision trees.

Введение

Центральной проблемой медико-биологических исследований, независимо от их специфики, является оценка состояния биологической системы. При этом под термином «биологическая система» мы понимаем организм в целом или его часть любой степени сложности на клеточном, органном уровне или уровне функциональной системы [1]. Эти же особенности относятся к анализу как биотехнических систем, так и любых сложных систем, имеющих большое количество каналов взаимодействия с внешней средой и характеризующихся существенной стохастической составляющей функционирования.

На современном уровне исследования биологических систем имеет смысл говорить именно о системном подходе, который получил развитие в трудах И.Р. Пригожина, П.К. Анохина, К.В. Судакова, У. Эшби и других выдающихся ученых.

Начиная со второй половины XX в., биология и медицина стремительно отходят от вербального описания и все больше тяготеют к формализации процессов, происходящих в биосистемах, использованию математических моделей и технических средств, а позднее и компьютерных технологий [2, 3]. Однако, это сопряжено с колоссальными сложностями, особенно при оценке динамических характеристик поведения биосистем и процесса их адаптации.

Значительные трудности изучения количественных характеристик биологических систем предопределяются особенностями и свойствами последних, и, прежде всего [1]: структурной и функциональной сложностью; вариабельностью параметров для одного состояния; нелинейностью характеристик; невозможностью полного описания системы. Последняя особенность («ущербность» описания) существенно осложняет постро-