На правах рукописи

Манушкин Евгений Сергеевич

Метод автоматического предсинтаксического анализа проектной документации с использованием КС-грамматик

Специальность 05.13.12
Системы автоматизации проектирования (информатика)
(технические науки)

Автореферат
диссертации на соискание ученой степени
кандидата технических наук

Москва 2012

Работа выполнена на кафедре «Информационные технологии и автоматизированные системы» Московского государственного института электроники и математики (технический университет)

Научный руководитель: кандидат технических наук, доцент
Клышинский Эдуард Станиславович

Официальные оппоненты: доктор технических наук, профессор
Майков Константин Анатольевич

кандидат технических наук, старший преподаватель
Лебедев Андрей Сергеевич

Ведущая организация: Учреждение Российской Академии Наук
Институт прикладной математики
им. М.В. Келдыша РАН

Защита состоится «22» мая 2012 г. в «14» часов на заседании диссертационного совета Д 212.133.03 при Московском государственном институте электроники и математики (техническом университете) по адресу: 109028, г. Москва, Б. Трехсвятительский пер., д.3.

С диссертацией можно ознакомиться в библиотеке Московского государственного института электроники и математики (технический университет)

Автореферат разослан: «21» 04 2012 г.

Ученый секретарь
диссертационного совета
д.т.н., доцент Леохин Ю.Л.

ОБЩАЯ ХАРАКТЕРИСТИКА РАБОТЫ

Актуальность темы. В ходе своего существования предприятия формируют огромные архивы документации. В этих архивах содержатся не только результаты официального документооборота (приказы, распоряжения и пр.), но и техническая документация по выполняемым и выполненным проектам: технические отчеты, проектная документация, планы и т.д. Значительная часть этих документов оформлена в формате текстового описания.

Одной из задач систем автоматизированного проектирования является систематизация хранения данных об изделии и приведение всей документации к единому стандарту. В этой области широко используются CALS-технологии. Составными частями CALS являются такие технологии, как ILM (Information Lifecycle Management) и PDM (Product Data Management). Основной задачей ILM-систем является хранение документации на изделие. Кроме того, ILM-системы отвечают за процессы хранения, распределения, миграции, архивировании и удаления данных в инфраструктуре предприятия. PDM системы позволяют управлять данными об изделии и информационными процессами жизненного цикла изделия, которые создают и используют эти данные.

Использование технологий ILM и PDM позволяет перейти к безбумажной обработке проектной документации, что позволяет значительно сократить время разработки изделия. Однако подобные технологии не производят интеллектуальную обработку данных, которая могла бы, с одной стороны, еще больше сократить время разработки и, с другой стороны, значительно упростить сам процесс разработки. Современное развитие науки и компьютерных технологий позволяют перейти на качественно иной уровень работы с документацией. На данный момент предприятия завершают переход от электронного хранилища к автоматической обработке документации. Автоматическая обработка документации (АОД) позволяет выполнять такие задачи, как, например, поддержка документации на нескольких языках и автоматическое исправление ошибок в тексте, информационный поиск и составление баз знаний о проектах. Для выполнения этих и многих других задач АОД требуется использование методов компьютерной лингвистики, которые занимаются непосредственно обработкой текстов на естественном языке.

Для автоматической обработки текстовой документации зачастую требуется проводить полный анализ текста, который требует существенных временных затрат. Среди прочих этапов полного анализа, этап синтаксического анализа является самым ресурсоемким этапом автоматического анализа текста. Грамматические методы

синтаксического анализа, принадлежащие к американской школе, основателем которой является Н. Хомский, являются наиболее изученными и по точности превосходят статистические методы синтаксического анализа. Однако скорость анализа при использовании методов американской школы остается относительно невысокой.

Для ускорения работы синтаксического анализа часто используют дополнительные этапы анализа. Одним из таких этапов является этап синтаксической сегментации, который выделяет априорную информацию о структуре предложения на основе выделения его фрагментов или составных конструкций. Синтаксический анализатор проводит разбор исходя из того, что найденные слова принадлежат той или иной синтаксической категории, и не будет предпринимать заведомо ложные попытки разобрать предложение по-другому. За счет этого количество итераций разбора заметно сокращается и, как результат, уменьшаются вычислительные затраты.

Правила для этапа синтаксической сегментации зачастую зависят от правил синтаксического анализа. Составление таких правил вручную требует тщательного изучения грамматики, к тому же правила, составленные человеком, требуют проверки и отладки. Мало того, что проверка и отладка являются достаточно долгим и трудоемким процессом, к тому же тщательно отлаженные правила могут не обеспечивать корректности для всех возможных видов предложений.

В связи с этим, разработка метода автоматического предсинтаксического анализа текстов проектной документации на изделие, является актуальной, так как предлагаемые теоретические положения позволяют автоматизировать процесс проектирования этапа синтаксической сегментации. Новые разработки и методы в области компьютерной лингвистики позволят внедрять новые технологии обработки проектной документации в САПР. Технические решения, связанные с автоматической обработкой текстов проектной документации, являются хорошим дополнением существующих CALS-технологий.

Цель работы. Целью диссертационной работы является повышение эффективности построения систем анализа проектной документации за счет автоматизации труда разработчиков таких систем.

Задачи исследования:

1. Анализ существующих методов синтаксического анализа и систем, использующих этап синтаксической сегментации.
2. Разработка алгоритма преобразования правил в формате расширенных БНФ в правила в формате расширенных сетей переходов (ATN).
3. Разработка формального метода вычисления терминальных множеств, использующихся на этапе предсинаксического анализа проектной документации.

4. Разработка метода автоматического предсинтаксического анализа проектной документации с использованием КС-грамматик.
5. Проектирование и разработка программного комплекса для предсинтаксического анализа текстов проектной документации, а также проведения вычислительных экспериментов для тестирования разработанного метода.

Методы исследования. При решении поставленных задач использовались основные методы теории компиляторов, компьютерной лингвистики, дискретной математики, в частности теории графов, а также методы объектно-ориентированного программирования.

На защиту выносятся следующие основные положения:

- Формальный метод вычисления терминальных множеств, необходимых для метода автоматического предсинтаксического анализа проектной документации.
- Метод автоматического предсинтаксического анализа проектной документации с использованием КС-грамматик

Научная новизна выполненной работы.

1. Предложен алгоритм эквивалентного преобразования грамматики расширенных БНФ в грамматику ATN.
2. Предложен новый метод вычисления терминальных множеств, необходимых для решения задачи проведения предсинтаксического анализа проектной документации.
3. Предложен новый метод, позволяющий проводить этап предсинтаксического анализа в системах анализа проектной документации, использующих грамматики расширенных БНФ и ATN.
4. Разработан метод тестирования качества и производительности результатов, полученных в диссертационной работе.

Практическая значимость результатов. Предложено новое техническое решение, позволяющее ускорить работу этапа синтаксического анализа текстов проектной документации. Решение использует только информацию о правилах синтаксического анализа, записанных в форме БНФ или ATN. Это позволяет разработчикам систем синтаксических анализаторов реализовывать этап синтаксической сегментации текста практически без затрат времени на разработку правил сегментации и без обязательного привлечения профессионалов в области компьютерной лингвистики, что в результате сокращает финансовые затраты и время, необходимые на разработку системы. В свою очередь проведение этапа синтаксической сегментации позволяет значительно увеличить производительность систем анализа проектной документации.

Автором разработано программное обеспечение, выполняющее предсинтаксический анализ предложений на основе предложенного метода. В результате проведенных экспериментов было установлено, что сокращение времени разбора отдельных предложений в результате применения разработанного метода превышает 80%, тогда как среднее ускорение находится на уровне 10%.

Достоверность и обоснованность полученных результатов подтверждается:

- корректностью использования математического аппарата и методов испытаний;
- апробацией основных результатов исследований на научных семинарах и публикацией результатов в научных журналах;
- результатами внедрения разработанных методов и рекомендаций в практику.

Реализация и внедрение результатов. Алгоритмы и методы, описанные в данной работе, реализованы автором в компьютерной программе. Программа создавалась как с целью апробации и совершенствования разрабатываемых методов и алгоритмов, так и с целью практического использования в машинном переводчике "Кросслейтор", разрабатываемом в ИПМ им. М.В. Келдыша РАН и при выполнении гос. контракта П-261 в рамках ФЦП "Научные и научно-педагогические кадры инновационной России" на 2009-2013 гг., заключенного между Министерством образования и науки и МИЭМ. Результаты работы обсуждались на научно-практическом семинаре «Новые информационные технологии в автоматизированных системах» в 2009 и 2010 гг.

Публикации. Всего автором опубликовано 8 научных работ из них 2 в журналах из перечня ВАК.

СОДЕРЖАНИЕ РАБОТЫ

Во введении дается обоснование актуальности темы диссертационной работы, определяется направленность исследования, формулируются цели и задачи исследования, определяется научная новизна и практическая значимость результатов.

В первой главе рассмотрены некоторые задачи, в которых необходим анализ текстов проектной документации, а также роль и задачи синтаксического анализа, без которого не может обойтись практически ни одна система полного анализа текста на естественном языке. Также в первой главе рассмотрены методы и подходы к синтаксическому анализу, в том числе в системах, выполняющих синтаксическую сегментацию. Для внедрения новых способов интеллектуальной обработки данных в рамках CALS-идеологии необходимо использовать методы автоматической обработки текстов (АОТ) на естественном языке. Это в первую очередь связано с тем, что большая часть документации на изделие состоит из текстового описания проекта. Применение таких методов позволяет переработать текстовую информацию, содержащуюся в

документации, в машинное представление, которое может быть использовано в целом ряде задач. Например, при разработке нового продукта проектировщикам может потребоваться документация на уже изготовленные продукты, схожие по назначению. В результате интеллектуального поиска в имеющихся архивах документации может выясниться, компания уже выпускала подобную продукцию, и нет необходимости в ее проектировании с нуля. Еще одним примером использования АОТ для документации является система "ЛоТА", которая анализирует документацию, связанную с логикой работы систем в авиационной промышленности, и определяет правильность описания алгоритмов действия экипажа воздушного судна в различных ситуациях. Одним из наиболее сложных приложений автоматической обработки проектной документации, содержащее огромное количество подводных камней, является поддержка документации на нескольких языках и автоматический перевод документации.

Приведенные выше примеры отнюдь не составляют полный список приложений АОТ проектной документации. Однако, несмотря на большие перспективы и явную необходимость автоматизации обработки проектной документации, разработки в данной области используются не слишком активно. Дело в том, что в данной области существует множество нерешенных проблем. В частности одной из таких проблем является соблюдение баланса качества и производительности систем анализа проектной документации.

Одним из самых ресурсоемких этапов анализа текста на естественном языке является синтаксический анализ. По сути, от качества синтаксического анализа зависит качество всего анализа текста. Системы, использующие базы правил, работают на уровне систем, использующих статистические методы для проведения синтаксического анализа. Качество таких систем сильно зависит от количества правил. Однако чем больше база правил, тем труднее ее пополнять и редактировать, к тому же большое число правил ведет к существенному снижению скорости анализа. В связи с этим последнее время наблюдается тенденция к слиянию двух подходов. На основании статистической или иной обработки больших объемов информации проводится генерация правил анализа текста.

Существуют различные методы повышения производительности синтаксического анализа. Одним из наиболее распространенных подходов, пользующийся популярностью в последнее время, является переход к поверхностному синтаксическому анализу. При проведении поверхностного синтаксического анализа перед анализатором не стоит задачи связать все слова в предложении. Увеличение производительности достигается за счет использования сравнительно простых правил, оперирующих, например, глагольными группами, предлогами и знаками препинания. Такой подход обладает высокой

производительностью, однако, к сожалению, не обладает высокой точностью, которой обладает полный синтаксический анализ.

Среди методов, позволяющих проводить полный синтаксический анализ, методы классической школы, основанные на порождающей теории Н.Хомского, являются наиболее изученными и широко применяются в ЕЯ-системах. Качество анализа, проводимого с помощью таких методов, напрямую зависит от размера базы правил анализа. Однако этот фактор также сильно влияет на производительность синтаксического анализа. Без применения дополнительных методов оптимизации анализа, его скорость будет экспоненциально зависеть от длины входного предложения. Примерами методов оптимизации полного анализа являются использование методов LL(1) грамматик и кэширование промежуточных результатов анализа.

Некоторые системы для ускорения синтаксического анализа наряду с оптимизацией парсинга используют дополнительные этапы обработки текста. Одним из таких этапов является синтаксическая сегментация предложения, которая применяется, например, в текстовом процессоре STP. На данном этапе во входном предложении выделяются границы синтаксических единиц (сегментов) для определения априорной информации о структуре предложения. Располагая такой информацией, парсер не будет предпринимать неуспешных попыток разбора того или иного фрагмента предложения исходя из того, что слова фрагмента принадлежат заданной синтаксической группе. Для выделения информации о сегментах используются вычислительно простые алгоритмы, за счет чего достигается выигрыш в скорости синтаксического анализа.

Правила для этапа синтаксической сегментации зачастую зависят от формата правил, используемых в системе. Составление подобных правил вручную может потребовать досконального изучения грамматики, используемой в системе, что является отталкивающим фактором для многих разработчиков. К тому же такие правила следует тщательно проверять и отлаживать, что на самом деле не может дать стопроцентную гарантию достоверности таких правил. Для гарантии достоверности требуется вручную проанализировать огромные массивы текстов, выделяя примеры и контрпримеры для правил. На сегодняшний день среди систем, выполняющих синтаксическую сегментацию, нет ни одной системы, в которой был бы решен вопрос автоматического пополнения базы правил сегментации. Как следствие, такие правила создаются лингвистами вручную.

В связи с этим задача разработки метода, который позволил бы автоматизировать процесс проектирования этапа синтаксической сегментации без привлечения специалистов, становится весьма актуальной.

На данный момент системы, позволяющие проводить полный синтаксический анализ, можно условно разделить на две категории: системы, использующие традиционную технологию, в основе которой лежит порождающая теория Н. Хомского, и системы, использующие правила, основанные на взаимоотношениях слов. Предложенный метод автоматического предсинтаксического анализа может быть применен лишь в системах, использующих традиционную технологию синтаксического анализа, поскольку в основе метода лежит использование элементов LL(1) и LL(2)-разборов, которые могут быть применены только в контекстно-свободных грамматиках.

Во второй главе определяется формальная основа для разработанного метода автоматического предсинтаксического анализа проектной документации с использованием КС-грамматик.

Предполагается, что, перед выполнением синтаксической сегментации, над входным предложением проводились несколько этапов обработки (например, графематический и морфологический анализ). Слово w в предложении на естественном языке (лексема) рассматривается как множество словоформ (омонимов): $w=\{d_1,d_2,...,d_n\}$, где d_i – словоформа слова w. Каждая словоформа d некоторого слова представлена в виде набора атрибутов $d=\{a_1,a_2,...,a_k\}$, где каждый атрибут a представляет собой пару $a=<n_a,v_a>$, в которой n_a– имя атрибута (например, часть речи, род, число, падеж и т.д.), а v_a – значение атрибута. Предполагается, что у любой словоформы не бывает двух атрибутов с одинаковыми именами.

В качестве входных данных предложенный метод использует грамматику, записанную в формате расширенных БНФ либо ATN. Для начала рассмотрим спецификацию грамматики расширенных БНФ.

Любой символ грамматики расширенных БНФ содержит набор параметров, которые являются, по сути, шаблонами атрибутов словоформы. Набор параметров некоторого символа грамматики s будем обозначать $P^s=\{p_1,p_2,...,p_k\}$, где каждый параметр p представляет собой четверку $p=<n_p,r_p,g_p,v_p>$ в которой:

n_p – имя параметра (например, род, число, падеж и т.д.);

r_p – отношение, которое может принимать значение «=», «!=» и «+» (равно, не равно, совпадает, соответственно);

g_p – номер (идентификатор) группы;

v_p – значение параметра.

Каждое значение r_p имеет приоритет (в данном определении значения перечислены в порядке уменьшения приоритета). Предполагается, что во множестве P^s не может быть двух параметров с одинаковыми именами.

Терминальный символ грамматики представляет собой набор параметров, нетерминальный символ грамматики представляет собой ссылку на правило грамматики и набор параметров. Для построения дерева зависимости словоформ каждый символ грамматики имеет свой идентификатор, уникальный внутри правила, и идентификатор символа, которому он подчиняется. Чтобы производить проверку согласования слов и синтаксических групп требуется вычислять атрибуты для разобранных фрагментов предложения и хранить их как промежуточный результат анализа. Для этой цели будем использовать словоформу.

Далее будет описана операция проверки соответствия между терминалом и словоформой, использующая историю разбора, которая хранится в памяти синтаксического анализатора. Будем рассматривать синтаксический анализатор, как некую виртуальную машину (парсер), которая имеет собственный стек, в котором хранятся номера правил и продукций, а также номера нетерминальных символов этих продукций, в которые был совершен рекурсивный спуск. Также парсер хранит номер текущего слова предложения, номер текущего правила, номер текущей продукции в правиле, а также номер текущего символа продукции. Для хранения промежуточных результатов разбора предложения, для каждой продукции предусмотрена своя ячейка памяти. После проверки на соответствие очередного символа продукции, в ячейку памяти продукции заносится данный символ и словоформа, с которой он успешно сравнился.

Определим операцию соответствия. Словоформа d соответствует (удовлетворяет) терминалу t (и мы будем обозначать этот факт следующим образом: $d \sim t$), если для любого параметра $p=<n_p,r_p,g_p,v_p>$ терминала t найдется атрибут $a=<n_a,v_a>$ словоформы d с таким же именем ($n_a=n_p$), при этом:

- если $r_p=$ '=', то $v_a=v_p$;
- если $r_p=$ '!=', то $v_a \neq v_p$;
- если $r_p=$ '+', то для каждой пары $d' \sim s'$ из ячейки памяти текущей продукции, где у символа s' есть параметр p' с тем же именем и номером группы что и у p (то есть $p'=<n_p,r_{p'},g_p,v_{p'}>$), должно выполняться равенство: $v_a=v_{a'}$, где $a'=<n_p,v_{a'}>$ является атрибутом словоформы d'.

В случае, когда в продукции встречается нетерминал, синтаксический анализатор пытается применить правило, на которое ссылается нетерминал. В результате применения правила для разобранного фрагмента текста вычисляется словоформа, которая должна пройти проверку соответствия с нетерминалом, выполняющуюся аналогичным способом как и для терминала.

Для вычисления словоформы, которая получается в результате применения правила, левая часть правила содержит набор пар вида $<n,g>$, где g – идентификатор группы, а n – имя атрибута. Набор атрибутов A^d новой словоформы d формируется следующим образом. Для каждой пары $<n,g>$ данного правила в A^d попадают такие атрибуты $a=<n_a,v_a> \in A^{d'}$, для которых найдется такой параметр $p = <n_p,r_p,g_p,v_p> \in P^s$, что $n_p=n_a=n$ и $g_p=g$, где $d'\sim s$ – пара из ячейки памяти успешно разобранной продукции, $A^{d'}$ – набор атрибутов словоформы d', P^s – набор параметров символа s.

Перейдем к описанию грамматики ATN, которая так же, как грамматика расширенных БНФ, может быть использована в качестве входных данных для предложенного метода.

Будем рассматривать расширенную сеть переходов как набор конечных автоматов, в котором выделен один стартовый автомат, с которого следует начинать разбор предложения. Каждый автомат в свою очередь представлен в виде графа, в дугах которого содержатся терминальные либо нетерминальные символы, описанные выше. Парсер, выполняющий обход сети автоматов, имеет указатель на текущее слово предложения, указатель на текущий граф сети переходов и указатель на текущую вершину графа. Парсер хранит историю разбора, которая организована в виде стека. В данный стек заносятся нетерминальные дуги, в графы которых был произведен рекурсивный спуск. Также для каждого графа организован отдельный стек, в который, как в ячейки памяти продукции расширенных БНФ, заносятся пары словоформ и символов дуг графа. После успешного прохода по некоторой дуге графа в этот стек заносится символ данной дуги и соответствующая ей словоформа. В случае если в дуге находится терминальный символ, рассматривается текущая словоформа входного предложения. Если в дуге находится нетерминал, словоформа вычисляется на основе обхода автомата, на который ссылается этот нетерминал.

Проход по дуге будет успешным только в том случае, если текущая словоформа либо словоформа фрагмента предложения, разобранного по нетерминалу, будет соответствовать параметрам символа, приписанному дуге. Операция соответствия словоформы d и символа грамматики s (будем обозначать это так же, как для расширенных БНФ: $d\sim s$) определяется следующим образом. Словоформа d удовлетворяет символу s, если для любого параметра $p=<n_p,r_p,g_p,v_p>$ символа s найдется атрибут $a=<n_a,v_a>$ словоформы d с таким же именем ($n_a=n_p$), при этом:

если r_p='=', то $v_a=v_p$;

если r_p='!=', то $v_a \neq v_p$;

если r_p='+', то для каждой пары d'~s' из стека графа, в которой у символа s' есть параметр p' с тем же именем и номером группы что и у p (то есть p'=<$n_{p'}$,$r_{p'}$,$g_{p'}$,$v_{p'}$>), должно выполняться равенство: v_a=$v_{a'}$, где a'=<$n_{p'}$,$v_{a'}$> является атрибутом словоформы d'.

Так же, как и правила расширенных БНФ, автоматы сети ATN содержат набор пар <n,g> для вычисления словоформы разобранного фрагмента текста. Набор атрибутов A^d новой словоформы d формируется следующим образом. Для каждой пары <n,g> графа G в A^d попадают такие атрибуты a=<n_a,v_a>$\in A^{d'}$, для которых найдется такой параметр p=<n_p,r_p,g_p,v_p>$\in P^s$, что n_p=n_a=n и g_p=g, где d'~s – пара из стека графа, $A^{d'}$– набор атрибутов словоформы d', P^s – набор параметров символа s.

Помимо спецификаций грамматик расширенных БНФ и ATN во второй главе показана связь между данными грамматиками и описан алгоритм эквивалентного преобразования грамматики расширенных БНФ в грамматику ATN.

В третьей главе изложен метод автоматического предсинтаксического анализа проектной документации с использованием КС-грамматик и алгоритмы вычисления терминальных множеств, необходимых для данного метода. Также в третьей главе предложен алгоритм интерпретации разметки текста, полученной на этапе предсинтаксического анализа. Этот алгоритм может быть использован на этапе синтаксического анализа для повышения производительности анализа текста. Алгоритмы вычисления терминальных множеств изложены для нотации ATN, формат которой изложен во второй главе. Предполагается, что в случае использования грамматики БНФ ее можно преобразовать в эквивалентную расширенную сеть переходов с помощью алгоритма, который также изложен во второй главе.

В данной работе, в отличие от других работ, под фрагментом будет пониматься часть предложения, которая может быть успешно разобрана с использованием некоторого правила грамматики синтаксического анализа. Для определения границ таких фрагментов можно использовать идею выводимости, используемую в теории формальных языков. Например, если слово или пара слов во входном предложении начинает цепочки, выводимые из некоторого правила грамматики, то скорее всего место в предложении, где находится это слово или пара слов, может быть успешно разобрано по данному правилу.

Исходя из этого для определения границ фрагментов можно использовать модифицированные множества *FIRST*, *LAST*, *FIRST2* и *LAST2*. В теории формальных языков данные множества определяются для цепочек терминальных и нетерминальных символов в правилах, описывающих LL(2)-грамматику. Элементами этих множеств являлись терминальные символы (в случае *FIRST* и *LAST*) либо упорядоченные пары

терминальных символов (в случае *FIRST2* и *LAST2*), с которых могут начинаться (*FIRST*, *FIRST2*) либо заканчиваться (*LAST*, *LAST2*) терминальные цепочки символов, выводимые из правил грамматики.

Грамматика, описываемая расширенной сетью переходов относится к расширенным КС-грамматикам, обладающим мощностью контекстно-зависимых грамматик и, следовательно, не является КС-грамматикой в чистом виде. В частности символы грамматики, в которых присутствуют параметры с отношением типа «+», являются зависимыми от других символов графа, внутри которого они записаны. В связи с большим количеством отличий используемой грамматики от КС-грамматики, придется переопределить множества *FIRST*, *LAST*, *FIRST2* и *LAST2* для используемой грамматики (новые множества будут помечены штрихами: *FIRST'*, *LAST'*, *FIRST2'* и *LAST2'* соответственно). Рассмотрим алгоритм вычисления множества *FIRST'(G)* графа *G* на примере.

В множество *FIRST'(G)* помещаются те терминальные символы, которым удовлетворяют всевозможные словоформы, с которых парсер может начать разбор графа *G*, находясь в его стартовом узле. Это значит, что мы можем поместить в данное множество все терминальные символы, которые находятся в дугах, исходящих из стартового узла, без каких-либо изменений.

Предположим теперь, что в одной из таких дуг содержится нетерминал *u*, который ссылается на граф *G'*, для которого множество *FIRST'(G')* уже вычислено. В этом случае в *FIRST'(G)* мы должны поместить терминалы из *FIRST'(G')* сделав их зависимыми только от символов графа *G* и нетерминала *u*. Для этого сначала рассмотрим все параметры у каждого терминала из *FIRST'(G')*, которые не участвуют в формировании разобранного по графу *G'* фрагмента текста. Те из них, которые имеют отношение «+», мы отсеем, а остальным (у которых отношение равно «=» либо «!=») присвоим фиктивный номер группы, например 0. Обозначим эту операцию функцией *sieve(G',t')*, которая возвращает терминал, полученный описанным выше способом, и принимает в качестве аргументов некоторый граф *G'* и терминал *t'* графа *G'*.

Теперь рассмотрим терминал t = *sieve(G',t')*, где $t' \in$ *FIRST'(G')*. Чтобы поместить терминал t в *FIRST'(G)* необходимо:

- убрать из t такие параметры $p=<n_p,r_p,g_p,v_p>$ у которых $g_p=0$, если у нетерминала u найдется параметр $p'=<n_{p'},r_{p'},g_{p'},v_{p'}>$ с таким же именем, как у p ($n_{p'}=n_p$);
- для тех параметров $p=<n_p,r_p,g_p,v_p>$ у которых $g_p \neq 0$ и у нетерминала u нет параметра с именем n_p, присвоить $g_p = 0$ если r_p=«=» либо r_p=«!=»; если r_p=«+», то параметр p необходимо убрать из терминала t;

- у оставшихся параметров $p=<n_p,r_p,g_p,v_p>$ для которых найдется параметр $p'=<n_p,r_{p'},g_{p'},v_{p'}>$ нетерминала u положить $g_p=g_{p'}$ и, если отношение $r_{p'}$ имеет больший приоритет чем r_p, то $r_p=r_{p'}$ и $v_p=v_{p'}$.

Обозначим эту операцию функцией *trail(t,u)*, которая возвращает терминал, полученный описанным выше способом, и принимает в качестве аргументов некоторый нетерминал u и терминал t. Будем говорить, что для пары t и u функция *trail(t, u)* не определена, если найдутся такие параметры $p=<n_p,r_p,g_p,v_p>$ и $p'=<n_{p'},r_{p'},g_{p'},v_{p'}>$ из t и u соответственно, которые противоречат друг другу, то есть $n_p=n_{p'}$, $r_p\neq$ «+», $r_{p'}\neq$«+» и

1) $r_p=r_{p'}$ и $v_p\neq v_{p'}$, либо
2) $r_p\neq r_{p'}$ и $v_p=v_{p'}$.

Если для пары t и u функция *trail* не определена, то это значит, что терминалу t соответствуют такие словоформы, которые могли бы присутствовать в цепочках, выводимых из графа нетерминала u, но не могут присутствовать в цепочках, выводимых из графа, в котором встретился нетерминал u. Например, предположим, что у некоторого терминала t присутствует параметр *p=<gender,"=",1,"male">*, а в списке параметров нетерминала u присутствует параметр *p'=<gender,"=",1,"female">*. Терминал t описывает словоформы мужского рода, тогда как нетерминал u требует, чтобы словоформа была женского рода, следовательно, терминал t никак не может быть использован в контексте u.

Теперь, когда мы описали функции *sieve* и *trail*, мы можем дать формальное определение множеству *FIRST'(G)*. Рассмотрим все дуги графа G, исходящие из его начальной вершины. Пусть α – символ, приписанный одной из таких дуг. Тогда:

- если α является терминалом, то α входит в *FIRST'(G);*
- если α является нетерминалом и ссылается на граф G', тогда для каждого терминала t' из *FIRST'(G')* в *FIRST'(G)* входит терминал *t=trail(sieve(G',t'),α),* если для *sieve(G',t')* и α функция *trail* определена.

Будем называть операцию *t=trail(sieve(G',t'),α)* преобразованием контекста терминала t' в контекст нетерминала α.

Далее определим еще два множества терминальных символов *LAST'(G)* и *ONLY(G)*, которые необходимы для вычисления множеств *FIRST2'(G)* и *LAST2'(G)*.

В множество *LAST'*(G) помещаются те терминальные символы, которым удовлетворяют всевозможные словоформы, на которых парсер мог бы закончить разбор графа G. Пусть α – символ, приписанный дуге, которая ведет в один из конечных узлов графа G. Если α - терминал, то $\alpha\in$ *LAST'(G).* Если α - нетерминал, который ссылается на граф G', то для каждого терминала $t'\in$*LAST'(G')* в *LAST'(G)* входит терминал *t=trail(sieve(G',t'),α),* если для *sieve(G',t')* и α функция *trail* определена.

В множество *ONLY(G)* помещаются терминалы, которым удовлетворяют всевозможные словоформы, с которых парсер может начать и одновременно закончить разбор графа *G*. Данное множество определяется аналогично множествам *FIRST'(G)* и *LAST'(G)* с той разницей, что в определении рассматриваются дуги графа *G* ведущие из стартового узла в любой конечный узел.

Для сокращения записи удобно определить множества FIRST', LAST' и ONLY для произвольного символа грамматики α. В случае, если α – терминал, тогда любое из перечисленных множеств состоит из единственного элемента, которым является α. Если α– нетерминал, то мы будем считать, что любое из перечисленных множеств для α будет равно соответствующему множеству графа, на который ссылается α.

Теперь определим множество пар терминальных символов *FIRST2'(G),* которым удовлетворяют всевозможные пары словоформ, с которых парсер может начать разбор некоторого графа *G*. Рассмотрим все пары дуг c_1 и c_2 такие, что c_1 исходит из начальной вершины, а c_2 исходит из вершины, в которую ведет первая дуга. Пусть α_1 и α_2 – символы, приписанные соответственно дугам c_1 и c_2, и пусть $x' \in ONLY(\alpha_1)$ и $x'' \in FIRST'(\alpha_2)$. Положим $t'=x'$, если α_1 – терминал и $t'=trail(sieve(G',x'),\alpha_1)$, если α_1 — нетерминал и G' – граф, на который ссылается α_1. Аналогично $t''=x''$, если α_2 терминал и $t''=trail(sieve(G'',x''),\alpha_2)$, если α_2 — нетерминал. Тогда в *FIRST2'(G)* входят такие пары терминалов ab, что:

1) в множество параметров a и b входят соответственно все параметры терминалов t' и t'', у которых номер группы равен 0;
2) в множество параметров a входят такие параметры $p'=<n_{p'},r_{p'},g_{p'}\neq 0,v_{p'}>$ терминала t', для которых в t'' найдется параметр $p''=<n_{p''},r_{p''},g_{p''},v_{p''}>$ у которого $n_{p'}=n_{p'}$ и $g_p=g_{p''}$; если такой параметр не нашелся и $r_{p'} \neq «+»$, то в a входят параметры $p=<n_{p'},r_{p'},0,v_{p'}>$ (аналогично для параметров b);
3) Ни один параметр терминала a с группой отличной от нуля не противоречит ни одному параметру b, имеющему ту же группу, и наоборот.

Аналогично определяется множество *LAST2'(G)*, с той разницей, что c_1 будет обозначать дугу, ведущую в один из конечных узлов графа *G*, а c_2 – дугу, ведущую в вершину, из которой исходит c_1. Символы x' и x'' будут браться соответственно из множеств $ONLY(\alpha_1)$ и $LAST'(\alpha_2)$.

Для выполнения предсинтаксического анализа текста нам понадобится еще одно множество. Определим множество пар терминальных символов *MIDDLE(G),* которым удовлетворяют всевозможные пары словоформ, которые могут встретиться в середине

цепочек словоформ, выводимых из графа G. Для определения множества $MIDDLE(G)$ нам понадобятся два вспомогательных множества $FIRST''(\alpha)$ и $LAST''(\alpha)$.

Пусть α – символ, приписанный некоторой дуге графа G. Если α приписан дуге, которая ведет в узел, из которого исходит хотя бы одна дуга, то $FIRST''(\alpha)=FIRST'(\alpha)$. В противном случае, если α является терминалом, то $FIRST''(\alpha)=\emptyset$; если α - нетерминал, который ссылается на граф G', то $FIRST''(\alpha)=FIRST''(G')$, где $FIRST''(G')$ определяется аналогично множеству $FIRST'(G')$ с той разницей, что для $FIRST''(G')$ рассматриваются дуги, которые исходят из начального узла G' и ведут в вершины из которых исходит хотя бы одна дуга.

Если α приписан дуге, которая исходит из узла, в который ведет хотя бы одна дуга, то $LAST''(\alpha)=LAST'(\alpha)$. В противном случае, если α является терминалом, то $LAST''(\alpha)=\emptyset$; если α - нетерминал, который ссылается на граф G', то $LAST''(\alpha)=LAST''(G')$, где $LAST''(G')$ определяется аналогично множеству $LAST'(G')$ с той разницей, что для $LAST''(G')$ рассматриваются дуги, которые ведут в конечные вершины и при этом исходят из вершин в каждую из которых ведет хотя бы одна дуга.

Теперь дадим формальное определение множества $MIDDLE(G)$ некоторого графа G. Рассмотрим все пары дуг c_1 и c_2 такие, что c_1 исходит из произвольной вершины графа G, а c_2 исходит из вершины, в которую ведет дуга c_1. Пусть α_1 и α_2 – символы, приписанные соответственно дугам c_1 и c_2. Пусть $x' \in LAST''(\alpha_1) \neq \emptyset$ и $x'' \in FIRST''(\alpha_2) \neq \emptyset$. Положим $t'=x'$, если α_1 – терминал и $t'=trail(sieve(G',x'),\alpha_1)$, если α_1 — нетерминал и G' – граф, на который ссылается α_1. Аналогично $t''=x''$, если α_2 терминал и $t''=trail(sieve(G'',x''),\alpha_2)$, если α_2 — нетерминал и G'' – граф, на который ссылается α_2. Тогда в $MIDDLE(G)$ входят такие пары терминалов ab, что:

1) в множество параметров a и b входят соответственно все параметры терминалов t' и t'', у которых номер группы равен 0;
2) в множество параметров a входят такие параметры $p'=<n_{p'},r_{p'},g_p \neq 0,v_{p'}>$ терминала t', для которых в t'' найдется параметр $p''=<n_{p''},r_{p''},g_{p''},v_{p''}>$ у которого $n_{p''}=n_{p'}$ и $g_p=g_{p''}$; если такой параметр не нашелся и $r_{p'} \neq «+»$, то в a входят араметры $p=<n_{p'},r_{p'},0,v_{p'}>$ (аналогично для параметров b);
3) Ни один параметр терминала a с группой отличной от нуля не противоречит ни одному параметру b, имеющему ту же группу, и наоборот.

На этапе синтаксической сегментации для каждого слова во входном предложении вычисляется два набора графов, назовем их N_1 и N_2. В N_1 входят те графы, разбор которых парсер мог бы начать, а в N_2 входят графы, разбор которых парсер мог бы закончить, находясь в данном месте предложения. Набор N_1 для некоторого слова w_i вычисляется

следующим образом. Рассмотрим соседнее справа слово w_{i+1}. Рассмотрим всевозможные пары словоформ $d'd''$, где d' - одна из словоформ w_i, а d'' - одна из словоформ w_{i+1}. Если пара словоформ $d'd''$ удовлетворяет хотя бы одной из пар терминалов из *MIDDLE(G)* для любого графа G, то набор N_1 пуст. В противном случае в N_1 помещаются такие графы G, для которых выполняется условие $d'\sim t'$, $d''\sim t''$ и $t't''\in FIRST2'(G)$.

Для вычисления N_2 слова w_i рассмотрим соседнее слева слово w_{i-1}. Рассмотрим всевозможные пары словоформ $d'd''$, где d' - одна из словоформ w_i, а d'' - одна из словоформ w_{i-1}. Если для любого графа G грамматики выполняется условие $d''\sim t''$, $d'\sim t'$ и $t''t'\in MIDDLE(G)$, то набор N_2 пуст. В противном случае в N_2 помещаются такие графы G, для которых выполняется условие $d''\sim t''$, $d'\sim t'$ и $t''t'\in LAST2'(G)$.

Для проведения предсинтаксического анализа некоторого предложения с использованием предложенного метода необходимо выполнить следующие действия:

- Если для синтаксического анализа используется грамматика БНФ, то до начала анализа текста преобразовать ее в грамматику ATN;
- До начала анализа текста провести расчет множеств *FIRST2' LAST2'* и *MIDDLE* для используемой ATN грамматики.
- Во входном предложении для каждого слова вычислить наборы N_1 и N_2;
- Каждое слово пометить как начинающее разбор по всем синтаксическим правилам из N_1 данного слова и заканчивающее разбор правил по всем правилам из N_2.

В четвертой главе содержится описание практической реализации разработанного метода и результаты вычислительного эксперимента, который проводился с целью тестирования предложенного метода.

Для тестирования предложенного метода был разработан программный комплекс, который использует модуль лексического анализа системы "Crosslator" и позволяет проводить синтаксический анализ предложением с использованием различных методов оптимизации. Предложенный метод тестировался на корпусе, составленным из технических текстов, и на корпусе литературного текста. Для тестирования была использована грамматика русского языка, записанная в формате расширенных БНФ и состоящая из 129 правил, которая конвертировалась в формат ATN. Для сравнения с существующими методами оптимизации анализа, корпусы тестировались также с использованием метода LL(1) разбора.

В результате тестирования первого корпуса установлено, что применение предложенного метода дало прирост производительности на 4%, тогда как применение LL(1) разбора только понизило производительность анализа. В результате тестирования второго корпуса было установлено, что применение предложенного метода дало прирост

производительности на 7%, тогда как комбинация предложенного метода с методом LL(1) разбора дало прирост на 8,7%.

Исходя из проведенного эксперимента можно установить следующее:

- Предложенный метод дает преимущество перед использованием метода LL(1) разбора для небольших корпусов текста.
- Для больших корпусов текста оптимальным является использование предложенного метода в сочетании с методом LL(1) разбора.
- Наилучшее ускорение анализа достигалось на предложениях, в середине которых присутствовали метки, однако встречаемость таких предложений в корпусе довольно низкая.

При экспериментах с системой "Crosslator" (использующей правила в формате БНФ) было установлено, что ускорение этапа анализа отдельных предложений увеличивалось как минимум в два раза и колебалось в среднем между 10 и 20 процентами.

В заключении содержится описание основных результатов работы.

АОД позволяет решать такие задачи, как, например, поддержка документации на нескольких языках и автоматическое исправление ошибок в тексте, информационный поиск и составление баз знаний о проектах. Для выполнения этих задач требуется использование методов компьютерной лингвистики, в частности, проведение полного синтаксического анализа текстов. Среди подобных методов, методы классической школы являются наиболее изученными и обладают широкими возможностями. Однако производительность подобных систем значительно ниже производительности систем неполного анализа. Внедрение этапа сегментации в системы полного анализа текстов позволит ускорить процесс обработки проектной документации, что в результате позволит сократить общее время на разработку изделия. На основании анализа существующих систем, использующих синтаксическую сегментацию, сделан вывод, что ни в одной системе, использующей сегментацию, не проработан вопрос автоматического пополнения базы правил сегментации. Вследствие этого правила сегментации добавляются лингвистами вручную, что является трудоемким процессом и представляет собой препятствие для разработки систем обработки документации для широкого спектра предметных областей.

В диссертационной работе предложен метод автоматического предсинтаксического анализа текстов проектной документации, который может быть использован для проведения этапа сегментации в системах, использующих КС-грамматики для полного анализа текстов на естественном языке. В качестве входных данных метод использует грамматики, записанные в формате расширенных БНФ или ATN. Предложенный метод

может быть использован как совместно с применением правил сегментации, описанными лингвистом, так и самостоятельно. Таким образом, использование данного метода позволяет сократить труд разработчиков систем анализа проектной документации, а также ускорить процесс настройки таких систем на новую предметную область.

Для описания метода автоматического предсинтаксического анализа текстов проектной документации был предложен метод вычисления терминальных множеств. В качестве входных данных метод вычисления терминальных множеств принимает грамматику в формате расширенных сетей переходов. Предполагается, что при использовании грамматики расширенных БНФ, данную грамматику можно преобразовать в эквивалентную грамматику расширенных сетей переходов. Для этой цели в диссертационной работе предложен алгоритм эквивалентного преобразования грамматики расширенных БНФ в формат ATN.

Для тестирования предложенного метода был разработан программный комплекс, который использует модуль лексического анализа системы "Crosslator" и позволяет проводить синтаксический анализ предложением с использованием различных методов оптимизации. В результате проведенных экспериментов с использованием разработанного программного комплекса, а также с использованием системы "Crosslator", было установлено, что сокращение времени разбора отдельных предложений в результате применения разработанного метода превышает 80%, тогда как среднее ускорение находится на уровне 10%.

Публикации по теме диссертации:

1. **Манушкин Е.С., Клышинский Э.С. Метод автоматического порождения правил синтаксической сегментации для задач анализа текстов на естественном языке // Информационные технологии и вычислительные системы, 4, 2009 г., С. 57-66.**
2. **Манушкин Е.С., Клышинский Э.С. Метод автоматического порождения правил синтаксической сегментации для расширенных сетей переходов // Информационные технологии и вычислительные системы, № 2, 2011г., С. 58-67.**
3. Манушкин Е.С. Метод автоматической генерации правил синтаксическо сегментации // Материалы ежегодной научно-технической конференции студентов, аспирантов и молодых специалистов МИЭМ. – М. МИЭМ, 2009, С. 147.

4. Клышинский Э.С., Манушкин Е.С. Метод автоматической генерации правил синтаксической сегментации // Сб. трудов двенадцатого научно-практического семинара «Новые информационные технологии», М. 2009, С. 135-148.
5. Клышинский Э.С., Манушкин Е.С. Математическая модель порождения правил синтаксической сегментации // Сб. трудов второй Всероссийской конференции «Знания – Онтологии – Теории», Новосибирск 2009, Том 2, С. 182-186.
6. Манушкин Е.С. Выделение правил синтаксической сегментации в нотации расширенных сетей переходов // Материалы ежегодной научно-технической конференции студентов, аспирантов и молодых специалистов МИЭМ. – М. МИЭМ, 2010, С. 204.
7. Манушкин Е.С. Применение метода автоматической генерации правил синтаксической сегментации для расширенных сетей переходов // Сб. трудов тринадцатого научно-практического семинара «Новые информационные технологии», М. 2010, С. 93-106.
8. Манушкин Е.С. Метод автоматической разметки предложения для этапа синтаксической сегментации // Сб. трудов пятнадцатого научно-практического семинара «Новые информационные технологии», М. 2012, С. 191-198.

Подписано в печать 19.04.2012г.
Тираж 120 экз. Заказ № 197
Отпечатано в типографии «ДЦ «Каретный Двор»»
101000, Москва, Лубянский пр., д.21, стр.5-5а
Тел.: (495) 621-70-09
www.allaprint.ru