ГЛАВА 3. СЕГМЕНТАЦИОННЫЙ АНАЛИЗ РУССКОГО ПРЕДЛОЖЕНИЯ

I. Поверхностный синтаксический процессор группы Диалинг.

Введение

Поверхностный синтаксический процессор русского языка разработан группой Диалинг[1] [А. Сокирко, 2001] в период с 1998-2001 гг. Фундаментом для исследований группы ДИАЛИНГ послужила система французско-русского автоматического перевода (ФРАП), разработанная в ВЦП совместно с МГПИИЯ им. М. Тореза в 1976-86 гг., и система анализа политических текстов (ПОЛИТЕКСТ), разработанная в Центре информационных исследований совместно с ВЦ ИСК РАН в 1991-97 гг [Н. Леонтьева, 1995].

Так же как и STP, процессор относится к алгоритмическим системам модульного типа. На вход синтаксическому анализатору подаются результаты работы графематики и морфологии, где каждая словоформа предложения представлена множеством морфологических омонимов. Принципиальным отличием в архитектуре анализатора можно считать двунаправленное взаимодействие модуля сегментации[2] и синтаксиса (т.е. построения синтаксических групп слов в предложении) [Д. Панкратов и др., 2000]. В случае STP такое взаимодействие было однонаправленным: топологическая структура с результатами сегментации поступала на вход грамматического модуля, отвечающего за выделение фразовых категорий. В процессоре Диалинг два модуля работают параллельно, чередуясь и обмениваясь накопленными знаниями (сегментация $\Leftrightarrow$ синтаксис).

Общая схема действий анализа

Общую схему действий в анализаторе можно представить в виде последовательности шагов:

[1] В разное время над созданием процессора работали А. Сокирко, Д. Панкратов, Л. Гершензон, Т. Кобзарева, И. Ножов.
[2] Фрагментация в терминах группы Диалинг (www.aot.ru).

1. Членение предложения по знакам пунктуации и сочинительным союзам на исходные отрезки; будем их также называть начальными сегментами. Объединение исходных отрезков с простыми случаями однородных рядов прилагательных, наречий, существительных, etc. Определение вершин и типов начальных сегментов.
2. Построение аналитических форм глагола внутри исходных отрезков.
3. Выделение именных групп терминов внутри исходных отрезков с помощью тезаурусов: общего, компьютерного и финансового.
4. Интерпретация вершин начальных сегментов, содержащих тире, и попытка восстановления тире в отрезках с нулевым Copul.
5. Декартово произведение омонимов внутри начальных сегментов – построение множества однозначных морфологических интерпретаций (МИ) одного сегмента. Построение простых синтаксических групп для каждой МИ сегмента подмножеством синтаксических правил: КОЛИЧ (“двадцать восемь”), ПРИЛ-СУЩ (“большой дом”), ПГ (“на столе”), ОДНОР_ПРИЛ (“умный и обаятельный”), etc.
6. Правила для объединения сочиненных начальных сегментов.
7. Построение сочиненных синтаксических групп для каждой МИ подмножеством синтаксических правил: ОДНОР_ИГ (“сын своего отца и дочь своей матери”), Р_С_ОДНОР_СУЩ (“как новые книги, так и пыльные папки”), Р_С_ОДНОР_ИНФ (“если не писать, так читать”), etc.
8. Правила для вложения контактно расположенных сегментов (причастных, деепричастных, придаточных определительных, etc.) – установление иерархии на сегментах.
9. Построение групп подмножеством синтаксических правил для каждой МИ, где вершина – синтаксическая группа подчиняющего сегмента, а зависимое – вершина вложенного сегмента.
10. Правила для объединения разрывных сегментов. Завершение полной сегментации предложения.
11. Построение групп множеством всех синтаксических правил для каждой МИ. Всего в модели процессора используется 40 типов групп.
12. Оценка синтаксического покрытия каждой МИ (лучший вариант получает максимальный вес) и установление порядка на множестве морфологических интерпретаций каждого сегмента.

Таким образом, расширение границ сегментов происходит последовательно с постепенным накоплением синтаксической информации внутри каждого начального сегмента. В свою очередь, каждая новая итерация синтаксических правил пытается расширять границы синтаксических групп, вкладывая уже построенные ранее группы в новые составляющие.

Морфологические интерпретации

МИ сегмента являются результатом декартова произведения омонимов словоформ, входящих в данный отрезок. Сегментация позволяет избежать полного декартова произведения предложения, что значительно сокращает число рассматриваемых вариантов. В словосочетании “рабочие стали” (рабочие: рабочий.N и рабочий.Adj; стали: сталь.N и стать.V) строится четыре однозначных МИ, для каждой из которых будет построена своя структура групп. Лучший вес получают МИ с максимальным количеством синтаксических групп. Множества синтаксических групп двух МИ, как правило, имеют непустое пересечение. Покрытием МИ называется число словоформ, вошедших в синтаксические группы данной МИ. Построим пример анализа отрезка (‘масса рабочего стекла’), демонстрирующего отличие синтаксической структуры групп на уровне разных МИ. Для данного отрезка строится четыре МИ (рабочего: рабочий.N и рабочий.Adj; стекла: стекло.N и стечь.V). Рассмотрим две из них, синтаксически равноправных и имеющих одинаковое покрытие (рис.1):

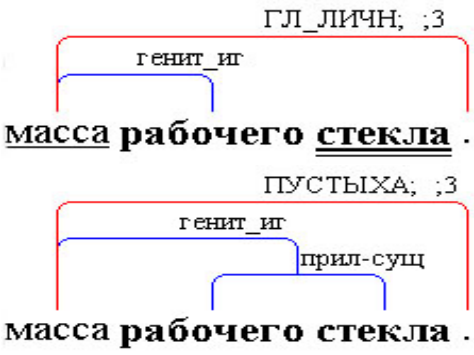

рис.1

Первая МИ задается комбинацией ‘рабочий.N’ и ‘стечь.V’, вторая – ‘рабочий.Adj’ и ‘стекло.N’. Структура групп первой МИ ПОДЛ(‘стекла’, ГЕНИТ_ИГ(‘масса’, ‘рабочего’)), второй - ГЕНИТ_ИГ(‘масса’, ПРИЛ_СУЩ (‘стекла’, ‘рабочего’)).

Внутрисегментный анализ

Внутрисегментный анализ в процессоре Диалинг – это синтаксис неразрывных групп, определенный в работе Сущанской как «синтаксис первого ранга» [Н. Сущанская, 1989]. Для синтаксической группы определены: (а) координаты (номер первого и последнего слова); (б) тип группы (ПРИЛ-СУЩ, ПГ, ОДНОР_ИГ, ПРЯМ_ДОП, etc.), который задается правилом; (в) координаты главной подгруппы группы. Например, группа “высокий дом” имеет соответствующие параметры: (а) {1, 2}; (б) ПРИЛ-СУЩ; (в) {2, 2}. Каждый тип группы строится отдельным правилом, которое имеет направление поиска зависимого и определяет вершину группы. Используя параметры группы, легко вычислить синтаксически главное слово, что позволяет переводить структуру вложенных групп в размеченное дерево зависимостей. Порядок применения синтаксических правил жестко задан как на отдельных подмножествах правил (шаги 5, 7, 9 в схеме), так и на множестве всех правил (шаг 11 в схеме). Так, правило ПРИЛ-СУЩ (согласованные прилагательное + существительное) должно отрабатывать раньше правила ГЕНИТ_ИГ (существительное + существительное в родительном): “высокий дом отца” ГЕНИТ_ИГ(ПРИЛ-СУЩ(1, 2), 3) = {1, 2} → {3, 3} и {2, 2} → {1, 1}, обратный порядок применения правил дает неправильную иерархию групп *ПРИЛ-СУЩ(1, ГЕНИТ_ИГ(2, 3)) = {2, 3} → {1, 1} и {2, 2} → {3, 3}. Подключение тезаурусов на первоначальном этапе синтаксического анализа (этап 3 в схеме) позволяет сохранять синтаксически правильную иерархию групп в тех случаях, когда устойчивые словосочетания входят в состав предложения. Для каждого текстового входа в тезаурусе определена синтаксическая модель, включающая в себя набор синтаксических групп, которые должен построить процессор [А. Сокирко, 2001]. Так, для отрезка предложения “высокий бюджет государства” на этапе 3 синтаксическим анализатором, при обращении к финансовому тезаурусу, будет обнаружено словосочетание ‘бюджет государства’, с приписанной к нему группой ГЕНИТ_ИГ; тогда для данного отрезка результирующая структура анализа выглядит как ПРИЛ-СУЩ(1, ГЕНИТ_ИГ(2, 3)), что было запрещено порядком применения правил в предыдущем примере. Синтаксические правила не используют модели управления слов (что неминуемо привело бы к построению разрывных групп и противоречит синтаксису первого ранга), кроме

тех случаев, когда управление напрямую вычисляется по морфологическим характеристикам слова. Единственное исключение в процессоре делается для группы подлежащее-сказуемое, которая в большинстве случаев является разрывной составляющей. Анализатор не ставит цели построить полную синтаксическую структуру внутри сегмента.

Синтаксические группы

В нижеследующей таблице приведен полный перечень типов синтаксических групп, определенных в процессоре. Каждый тип группы в синтаксическом анализе строится отдельным правилом, способным учитывать грамматическое согласование, предложное управление и линейный порядок подгрупп в сегменте предложения.

Тип	**Название**	**Пример**
Количественная группа (последовательность числительных)	КОЛИЧ	Двадцать восемь
Последовательность чисел	СЛОЖ-ЧИСЛ	12,3, II-III
Группа существительного, пре-модифицированная одним или несколькими прилагательными	ПРИЛ-СУЩ	Длинная тяжелая дорога, двигающийся человек
Группа существительного, пре-модифицированная наречным числительным	НАР-ЧИСЛ-СУЩ	Много ребят, мало стульев
Группа существительного, пре-модифицированная числительным	СУЩ-ЧИСЛ	Восемь попугаев, два человека
Предложная группа	ПГ	В дом, на холме
Группа однородных прилагательных	ОДНОР_ПРИЛ	смелый, красивый и умный
Глагол, пре-модифицированный наречием	НАРЕЧ_ГЛАГОЛ	злостно нарушает, тяжело жить
Полное или краткое прилагательное, пре-модифицированное наречием	НАР_ПРИЛ	очень красивый, весьма полезный, особенно хорош.
Цепочка наречий	НАР_НАР	как легко, так интересно
Аналитическая форма сравнительной степени прилагательного или наречия	СРАВН-СТЕПЕНЬ	гораздо сильнее; значительно больше
Отрицательная частица ‘не’ + глагол	ОТР_ФОРМА	Не любить; не знать
Группа контактно расположенного справа прямого дополнения	ПРЯМ_ДОП	Рубить дрова; смотреть фильм
Генитивное определение в постпозиции	ГЕНИТ_ИГ	Рука человека; стол отца; набор грузов
Группа однородных наречий	ОДНОР_НАР	Очень и так
Группа глагола контактно справа пост-модифицированного инфинитивом	ПЕР_ГЛАГ_ИНФ	пойти выпить; позвать гулять.
Группа однородных инфинитивов	ОДНОР_ИНФ	гулять, думать и говорить
Группа имя + фамилия	ФИО	Владимир Набоков
Группа однородных именных групп	ОДНОР_ИГ	красивый дом и густой лес
Прилагательное, пре-модифицированное ‘такой’ или	МОДИФ_ПРИЛ	такая красивая

‘самый’		
Группа существительного, пост-модифицированная причастным оборотом (сегментом)	ПРИЧ_СУЩ	Дом, построенный …
Группа подлежащее-сказуемое	ПОДЛ	Человек идет
Группа приложения	ПРИЛОЖЕНИЕ	Его отца, очень обидчивого человека, эта реплика вывела из себя. (отец -> человек)
Группа существительного, пост-модифицированная группой обособленного прилагательного	СУЩ_ОБС_ПРИЛ	сестра, совсем больная,… (сестра -> больная)
Группа однородных наречий, Р_С: сочиненных повторяющимися или разрывными союзами	Р_С_ОДНОР_НАР	не только вчера, но и сегодня
Группа однородных Р_С прилагательных	Р_С_ОДНОР_ПРИЛ	хотя и очень больной, но довольно сильный
Группа однородных Р_С причастий	Р_С_ОДНОР_ПРИЧ	как работающий, так и преуспевющий
Группа однородных Р_С сущ-ных	Р_С_ОДНОР_СУЩ	как книги, так и папки
Группа однородных Р_С мест-ний	Р_С_ОДНОР_МС	ни он, ни она
Группа однородных Р_С инфинитивов	Р_С_ОДНОР_ИНФ	если не писать, так читать
Группа однородных Р_С деепричастий	Р_С_ОДНОР_ДЕЕПР	если не думая, то говоря
Предикатив, пре-модифицированный наречием	НАР_ПРЕДИК	очень интересно
Группа сущ-ного, пост-модифицированного необособленным прил.	ПРИЛ_ПОСТПОЗ	впечатление необычное
Прил., пре-модифицированное ‘более’ или ‘менее’	АНАТ_СРАВН	более сильный, менее привлекателен
Группа однородных Р_С предложных групп	Р_С_ПГ	как на шкафу, так и в столе
Конструкция ‘каждый’ или ‘один’ + ПГ с предлогом ‘из’	ЭЛЕКТ_ИГ	Один из них, каждый из ваших людей
Формат электронного адреса	ЭЛ_АДРЕС	www.aot.ru
Сравнительное степень прил. + именная группа в генитиве	ОТСРАВН	левее сапога, умнее человека

Структура сегмента

Для каждого сегмента определены: (а) координаты (номера слов в предложении, соответствующих левой и правой границе сегмента); (б) вершина сегмента: номер слова и тип вершины $h \in H$ = { ГЛ_ЛИЧН (глагол в личной форме), КР_ПРЧ (краткое причастие), КР_ПРИЛ (краткое прилагательное), ПРЕДК (предикативное слово), ПРИЧ (причастие), ДПР (деепричастие), ИНФ (инфинитив), ВВОДН (вводное слово), ПУСТЫХА }, где ПУСТЫХА означает пустую вершину, а все типы в H иерархически упорядочены; (в) союз или союзное слово. Тип вершины сегмента задается по значению селективного признака вершины и может быть представлен в виде множества значений, соответствующих допустимым значениям морфологических омонимов. В

сегменте “когда ему весело” (весело: веселый (кр. прил.), весело (предикатив), весело (наречие)) определены соответствующие параметры: (а) {1, 3}; (б) [3] и { КР_ПРИЛ, ПРЕДК, ПУСТЫХА }; (в) ‘когда’.

Порядок выполнения сегментационных правил (отраженный в этапах 1, 6, 8 и 10 схемы) определен последовательностью работы их подмножеств, внутри этих блоков правил порядок выполнения – свободный. В процессоре определено три операции на сегментах: объединение, вложение и деление. Каждое правило, фактически, является проверкой ряда условий на возможность применения той или иной операции над сегментами.

Операция объединения сегментов

Помимо условий, накладываемых на вершины двух претендующих на объединение сегментов, основным критерием объединения служит возможность построения группы (однородных именных групп, групп с разрывными союзами или группы подлежащего-сказуемого) на границе двух сегментов. Получившейся в результате объединения сегмент наследует вершину того начального сегмента, чей тип вершины находится выше в иерархии Н. Морфологические интерпретации нового сегмента получаются путем перемножения МИ объединившихся начальных сегментов. Пример операции объединения (рис. 2)[3]:

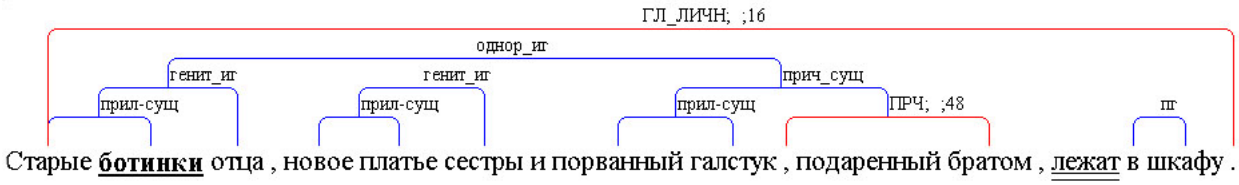

рис.2

Операция вложения сегментов

Вложение одного сегмента в другой может быть как согласованным, так и произвольным. Правила, отвечающие за согласованное вложение сегмента, ищут вершину придаточного определительного или причастного оборота в левостоящем сегменте. Для выполнения операции вложения вводится специальное понятие – юнит (unit) [Д. Панкратов и др., 2000]. Юнитом в

сегменте может быть либо слово, представленное множеством своих омонимов, либо вложенный сегмент, представленный множеством омонимов своей вершины. Тогда морфологическая интерпретация (МИ) сегмента есть линейная последовательность омонимов его юнитов. Понятие юнит позволяет строить синтаксические группы для вложенных сегментов. Так, на рис.2 построена группа ПРИЧ_СУЩ, где главной подгруппой является ПРИЛ_СУЩ(порванный, галстук), а зависимой – вложенный сегмент с вершиной ‘подаренный’. Произвольное вложение сегментов происходит в случае объединения двух дистантно расположенных исходных отрезков, тогда все сегменты, находящиеся между ними, вкладываются в новый сегмент, полученный в результате объединения. Пример произвольного вложения (рис. 3):

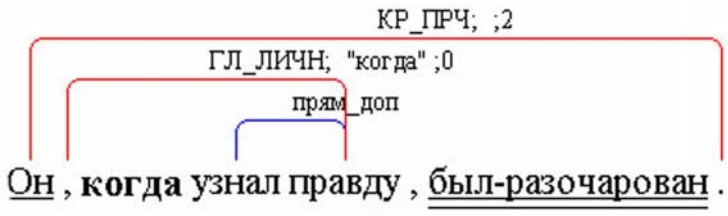

рис.3

При вложении одного сегмента в другой МИ подчиняющего сегмента умножаются на омонимы вершины подчиняемого сегмента с сохранением в МИ ранее построенных синтаксических групп.

Операция деления сегментов

Операция деления сегмента выполняется только для одного правила: выделения необособленного согласованного определения (НСО). Не имея границ, выраженных знаками препинания, НСО вычленяется алгоритмически в отдельный подсегмент. Пример деления (рис. 4):

рис.4

[3] Использован графический интерфейс процессора Диалинг.

Преобразование групп в бинарные отношения

Возможность вычисления синтаксически главного слова для каждой группы позволяет преобразовывать иерархическую структуру групп в множество бинарных отношений. Вычисление лексической вершины сегмента также позволяет переводить иерархическую структуру вложений сегментов во множество бинарных связей, где синтаксически главным словом является вершина вышестоящего сегмента, а зависимым – вершина непосредственно вложенного в него сегмента. Так, синтаксическая структура предложения

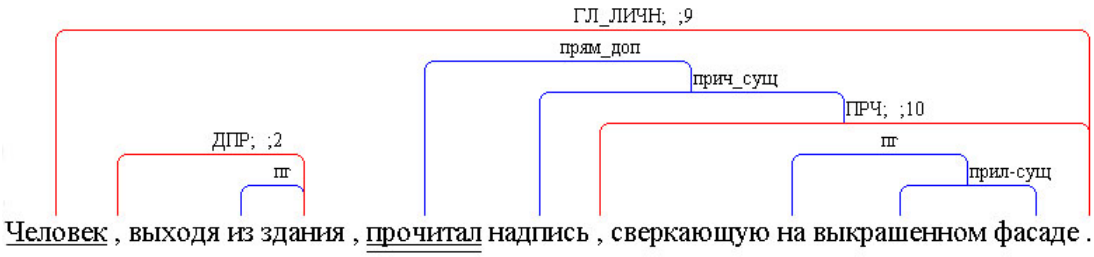

рис.5

автоматически преобразуется в множество бинарных отношений:

ИЗ [ПРЕДЛ][] --- ЗДАНИЕ [С,ср,но][ед,рд] : ПГ;

ФАСАД [С,мр,но][ед,пр] --- ВЫКРАСИТЬ [ПРИЧАСТИЕ,св,пе][мр,ср,ед,пр,стр,прш] : ПРИЛ-СУЩ;
НА [ПРЕДЛ][] --- ФАСАД [С,мр,но][ед,пр] : ПГ;
НАДПИСЬ [С,жр,но][ед,им,вн] --- СВЕРКАТЬ [ПРИЧАСТИЕ,нс,нп][жр,ед,вн,дст,нст] : ПРИЧ_СУЩ;
ПРОЧИТАТЬ [Г,св,пе][мр,ед,дст,прш] --- НАДПИСЬ [С,жр,но][ед,им,вн] : ПРЯМ_ДОП;
ПРОЧИТАТЬ [Г,св,пе][мр,ед,дст,прш] --- ЧЕЛОВЕК [С,мр,од][мн,ед,им,рд] : ПОДЛ;
ПРОЧИТАТЬ [Г,св,пе][мр,ед,дст,прш] --- ВЫХОДИТЬ [ДЕЕПРИЧАСТИЕ,нс,нп][дст,нст] : ДЕЕПР;

где в левой части находится главное солво, а в правой – зависимое; все словоформы внутри связей лемматизированы; для каждой леммы внутри первых квадратных скобок выведены значения ее селективных признаков, во вторых квадратных скобках указаны ее граммемы в анализируемом предложении; двоеточием отделяется тип связи, в большинстве случаев совпадающий с типом синтаксической группы.

Заключение

Алгоритмическая прозрачность (rule-based подход) процессора Диалинг, к сожалению, приводит к большим вычислительным затратам. Недостатком

системы является отсутствие синтаксической омонимии как на уровне иерархии и границ сегментов, так и на уровне групп; также в анализаторе не обрабатываются случаи сочинения предикатов, что служит иногда препятствием для объединения исходных отрезков и приводит к ошибочному построению сегментационной структуры.

Программная реализация процессора выполнена на языке C++. Взаимодействие между графематическим, морфологическим и синтаксическим модулями в программе организовано через стандартный COM интерфейс, результаты синтаксического анализа также доступны внешним приложениям через COM интерфейс. Неоспоримым достоинством процессора Диалинг является его завершенность: программная реализация доведена до уровня промышленного использования, - система характеризуется приемлемой скоростью анализа и устойчивостью на открытом пространстве реальных текстов.

II. Сегментационный процессор группы ОИС.

Введение

Синтаксический анализатор научный группы отделения интеллектуальных систем (ОИС) Института Лингвистики РГГУ (Д.Г. Лахути, Т.Ю. Кобзарева, И.М. Ножов) был создан в 1999-2003 гг. при финансовой поддержке РФФИ и ФЦП «Интеграция высшего образования и фундаментальной науки». Предлагаемый проект продолжает развиваться и содержит наиболее полную реализацию идей сегментации русского предложения. Фундаментом для проводимых исследований послужила модель автоматического поверхностно-синтаксического анализа русского предложения, разработка которой была начата еще в 1971 г. в Информэлектро в секторе (затем отделе) Д.Г.Лахути группой лингвистов под руководством Г.А.Лесскиса. Последняя версия этой модели, положенная в основу описываемой реализации, разработана Т.Ю. Кобзаревой.

Стратегии

Сохраняя принцип модульности проектируемой системы, процессор не использует в своей модели правила, а применяет грамматические стратегии для

анализа предложения, каждая из которых является независимым модулем, вызываемым в качестве подпрограммы. Стратегии позволяют эффективно работать с морфологической и синтаксической омонимиями в системе. Анализ в процессоре однонаправлен: первая фаза работы анализатора выполняет построение предложных PRN и именных NRA групп (PRN-NRA модуль), во второй фазе осуществляется сегментация сложного предложения с вызовом (только в алгоритмически зафиксированных случаях) модуля сочинения, - PRN-NRA модуль⇒сегментация.

Клауза, включающая в себя хотя бы одну другую клаузу, называется сложной клаузой, или полипредикативной конструкцией [Тестелец, 2001]. Поскольку цель нашего сегментационного анализатора состоит в выделении простых сегментов в составе сложного предложения, на данном этапе присутствие полипредикативной конструкции лишь иногда фиксируется межсегментной связью в построенном графе.

Выделяются два алгоритмически обоснованных класса сегментов [Т.Ю.Кобзарева, 2002]: α- и β-сегменты. β-сегментами называется множество простых предложений в составе сложного, α-сегментами – все остальные. α-сегменты делятся на следующие подклассы:

1. Придаточные предложения - SubS сегменты.
2. Деепричастные обороты - DvS сегменты.
3. Причастные обороты и обособленные определительные обороты - AS сегменты.
4. Предложные обороты - PS сегменты.
5. Вводные обороты - PrtS сегменты.

В начале работы системы отрабатывает блок алгоритмов (PRN-NRA модуль), отвечающий за построение основных синтагм (синтаксических связей между словоформами), без которых невозможно последующее выделение простых сегментов.

Последовательность построения синтагм:

1. Предложные группы (PRN), где предлог - хозяин, существительное - слуга.
2. Группы прилагательное-существительное (NRA), где существительное - хозяин, прилагательное - слуга.

3. В алгоритмически определенных случаях фиксируется группа управления существительным (LRN), где лексема L - хозяин, существительное - слуга.

NRA и PRN позволяют элиминировать часть потенциальных границ сегментов – операторы (запятые и сочинительные союзы), которые служат разделителями в однородных цепочках, вложенных в проективные синтагмы. Также NRA и PRN позволяют косвенно снимать некоторые типы омонимии в граммемах: например, вошедшее в PRN (‘на стол’) или вложенное в NRA (‘сменяющая день ночь’) существительное (‘стол’ и ‘день’) утрачивает именительный падеж и больше не может претендовать на позицию субъекта в сегменте. Синтагмы, так же как и сегменты, обладают свойством рекурсивности. Предложные группы PRN могут иметь неограниченной глубины вложения между предлогом и существительным-слугой, а согласованные определения NRA – любое количество параллельных вставлений произвольной глубины. Построенные PRN и NRA сворачиваются до уровня одной синтаксической единицы со всеми вложениями, находящимися внутри границ, определенных связью [Т. Кобзарева и др., 2001].

Следующий этап анализа разбивает предложение на первоначальные отрезки, границами которых являются знаки пунктуации, и приписывает каждому отрезку значение класса или подкласса. Завершающий и основной этап анализа (непосредственно сегментация) объединяет отрезки, тем самым укрупняя узлы и формируя простые сегменты. Сегментация состоит из двух модулей: α- и β-анализа. В обусловленных алгоритмически случаях, в процессе сегментации фиксируются цепочки сочиненных составляющих, связь подлежащее-сказуемое или связь глагола с прямым дополнением. Поиск и фиксация всех таких синтагм в предложении, в отличие от PRN и NRA, не являются обязательным условием анализа.

Таким образом, любое предложение естественного языка в системе описывается двумя графами: граф синтагм и граф сегментов [И. Ножов, 2002]. Узлами графа синтагм являются терминальные единицы (словоформы), дуга в графе образует синтагму и задает тип связи. Узлами графа сегментов являются нетерминальные единицы - сегменты, - дуга в графе задает межсегментную связь. Грамматическое сочинение терминальных единиц в графе синтагм и

сочинение однородных сегментов в графе сегментов нарушает древесность графов, так как каждый элемент множества узлов, образующих сочинительную связь, попарно связан со всеми остальными элементами множества и одновременно является как слугой, так и хозяином всех узлов, принадлежащих множеству сочинения. Таким образом, граф синтагм и граф сегментов - ориентированные графы, содержащие контуры и замкнутые пути [А. Белоусов, С. Ткачев, 2001]. Как следует из выше изложенного подхода к построению синтагм и сегментов, связность графа не является обязательным условием.

Морфологическая и синтаксическая омонимии

Сегментационный анализатор работает с двумя типами омонимии: морфологической и синтаксической. Морфологическая омонимия в предложении обусловлена присутствием словоформ, принадлежащих одновременно двум и более лексемам. Так во фразе '*Женщина мыла оконное стекло*': слово '*мыла*' омонимично: родительный падеж существительного 'мыло' и прошедшее время глагола 'мыть'; слово '*стекло*' также омонимично: винительный падеж существительного 'стекло' и прошедшее время глагола 'стечь'. Декартово произведение омонимов дает четыре комбинаторно возможных интерпретации данного предложения, а следовательно порождает четыре графа синтагм (МИ в процессоре Диалинг). Ниже будет описан метод активизации омонимов, который часто позволяет избежать декартова произведения. Результаты сегментации также сокращают число комбинаторных вариантов (как и в системе Диалинг), так как построение связного дерева синтагм проводится в пределах одного простого сегмента, а не в рамках всего предложения. Морфологическая омонимия в сегментационном анализаторе определена только на графе синтагм.

Синтаксическая омонимия представляет собой гораздо более абстрактный случай и задается стратегиями анализа в алгоритмах. Так, синтаксическая омонимия выражается наличием или отсутствием той или иной связи в графе синтагм. Например, синтаксическая омонимия часто возникает в случаях интерпозиции обособленного определительного оборота, когда зависимое определение может иметь хозяина как слева, так и справа. Синтаксическая омонимия в графе сегментов возникает при объединении первоначальных отрезков в более крупные узлы. Омонимичный граф сегментов,

порожденный таким типом омонимии, будет отличаться от своего родителя границами сегментов и, возможно, числом узлов. Синтаксическая омонимия определена как на графе синтагм, так и на графе сегментов.

Граф синтагм

Внешними составляющими в механизме построения синтагмы - направленной синтаксической связи между словоформами - служат:

- Линейное распределение словоформ в цепочке терминальных единиц предложения.
- Процедура проверки полного согласования.
- Процедура проверки частичного согласования для множественного числа.
- Процедура проверки согласования по управлению.

Линейная структура предложения S естественного языка состоит из множества словоформ $S = \{W_1, W_2, \ldots, W_n\}$, где каждая словоформа представлена множеством морфологических омонимов $W_i = \{h_1, h_2, \ldots, h_m\}$, где h_i является кортежем значений {часть речи, граммемы, примитивная модель управления}. Таким образом, предложение можно представить как упорядоченную цепочку элементов $S' = \{e_{11}, e_{12}, \ldots, e_{1m}, \ldots, e_{np}\}$, где первый индекс элемента соответствует номеру словоформы в предложении, а второй – номеру морфологического омонима словоформы. Первоначальный этап синтаксической сегментации, отвечающий за построение графа синтагм, начинает работать с линейным представлением S. При построении синтагм и поиске предикатов происходит активизация омонимов, в результате чего возникают смешанные цепочки типа $S'' = \{W_1, e_{2j}, W_3, \ldots, e_{np}\}$. Таким образом, графом синтагм предложения S называется граф $G=(S'', E)$, где S'' - множество узлов, состоящих из элементов смешанной цепочки S'', а E - множество упорядоченных пар на S'', то есть множество синтагм; в частном случае, при отсутствии морфологических омонимов, $G=(S', E)$. Существует динамически пополняемый список омонимичных графов синтагм $L = \{G_1=(S''_1, E_1), G_2= (S''_2, E_2), \ldots, G_k= (S''_k, E_k)\}$, активизация нового омонима является событием, которое вызывает пополнение списка. Каждый G_i содержит минимальное число

синтагм, необходимых для дальнейшей сегментации (т.е. связность графа – необязательное условие).

Алгоритм активизации омонимов построен на принципе ленивых вычислений. Аргументом функции, принимающей решение о порождении омонимичного графа, является проверяемое алгоритмом условие или тип построенной синтагмы.

Здесь приводится пример[4] построения анализатором графа синтагм фрагмента предложения, на котором отчетливо проявляется свойство рекурсивности синтагм:

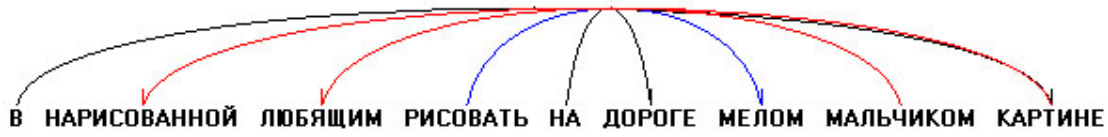

рис.1

Граф сегментов

Операция объединения задана на множестве первоначальных отрезков. Условием объединения служит:

- Сочинение.
- Процедура определения и устранения разрыва, образованного вклиниванием α-сегмента, между двумя первоначальными отрезками.

Синтаксическая сегментация проводится для каждого графа синтагм из списка L, собирая разорванные вложениями α- и β- сегменты. Предложение S представляется в виде графа, в узлах которого находятся сегменты, а дуги являются связями между сегментами, такой граф сегментов GS=(ST, SE) можно представить как множество узлов $ST = \{Segm_1, Segm_2, \ldots, Segm_n\}$, где $Segm_i \subset S''$, и множество SE межсегментных связей на ST. Каждому графу синтагм G из списка L соответствует множество графов $L' = \{GS_1=(ST_1, SE_1), GS_2=(ST_2, SE_2), \ldots, GS_m=(ST_m, SE_m)\}$, множественность интерпретаций графа синтагм G обусловлена возникновением синтаксической омонимии. После того, как проанализированы все элементы списка L, мы получаем множество всех возможных графов сегментов данного предложения $M = \{GS_1, GS_2, \ldots, GS_q\}$, из которых в дальнейшем должны выбираться лучшие структуры.

[4] Использован графический интерфейс процессора.

Множественность синтаксических интерпретаций зачастую определяется естественной смысловой омонимией в предложении. Уже на этой точке выбора отсекается часть активизированных морфологических омонимов. После завершения сегментации возможно проведение полного синтаксического анализа внутри простых синтаксических единиц, каковыми являются α- и β-сегменты.

Ниже приведены два омонимичных графа сегментов предложения: “Он постоянно видел отца, красящего забор младшей сестры, старый дом и сарай.”

рис.2

рис.3

Оба варианта являются синтаксически и семантически допустимыми вариантами интерпретации этого предложения.

Сегментная проективность

Базовым ограничением, на котором строится анализ в процессоре, является проективность сегментной структуры предложения. Первоначально линейная последовательность предложения S разбивается по знакам препинания на исходные α- и β-отрезки в соответствии с приведенной классификацией. Класс и подкласс отрезка определяется по наличию в нем «сегментообразующего» слова: подчинительного союза, деепричастия, полного причастия/прилагательного, оставшегося после анализа NRA без хозяина, и т.д. α-отрезки на этом этапе – еще не сегменты, но безусловные «начала» α-сегментов. Все остальные отрезки называются β-отрезками [Т. Кобзарева и др., 2000]. Рассмотрим известный уже нам пример предложения "Девочка, решив уже, когда ее позвали, задачу, засмеялась", разделив его на исходные отрезки:

β_1[девочка] α_1[DvS: решив уже] α_2[SubS: когда ее позвали] β_2[задачу] β_3[засмеялась]

Строя сегменты ([1β]'девочка засмеялась', [2α]'решив уже задачу' и [3α]'когда ее позвали') путем объединения исходных отрезков, получаем схему такого объединения: $\alpha_1 \rightarrow \beta_2$ и $\beta_1 \rightarrow \beta_3$, где стрелки не демонстрируют привычные грамматические связи, а лишь отражают тот факт, что подсоединяемый β-отрезок переносится влево. Между уже собранными полными сегментами определяется структура связей: в данном примере [1β]→[2α]→[3α]. Построим предложение с более сложной и глубокой иерархией вложений “Мать, когда мальчик, выйдя во двор, где стояла машина, к которой было необходимо подойти, споткнулся, не заметив приступка, и упал в сугроб, наметенный за ночь, выбежала ему помочь”:

β_1[мать] α_1[SubS: когда мальчик] α_2[DvS: выйдя во двор] α_3[SubS: где стояла машина] α_4[SubS: к которой было необходимо подойти] β_2[споткнулся] α_5[DvS: не заметив приступка] β_3[и упал в сугроб] α_6[AS: наметенный за ночь] β_4[выбежала ему помочь]

Схема объединения исходных отрезков в полные сегменты представлена на рис.4:

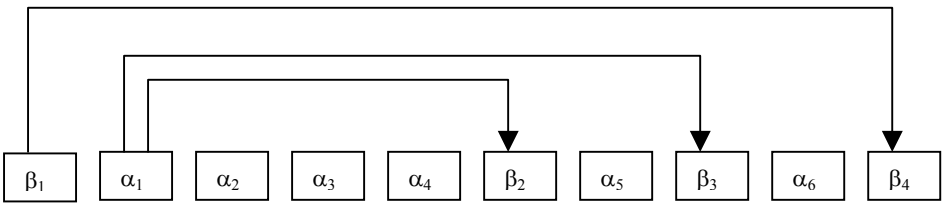

рис.4

В результате процедуры объединения образовано семь сегментов: [1β]'мать выбежала ему помочь', [2α]'когда мальчик споткнулся и упал в сугроб', [3α]'выйдя во двор', [4α]'где стояла машина', [5α]'к которой было необходимо подойти', [6α]'не заметив приступка', [7α]'наметенный за ночь'. Структура межсегментных связей показана на рис.5:

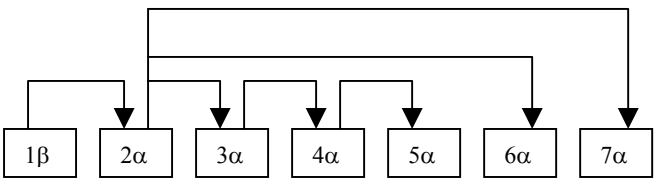

рис.5

Получившаяся схема объединения (рис.4) проективна, т.е. не содержит пересечения стрелок. Такого рода проективность, связанная с сегментной структурой предложения, - абсолютно жесткая, в отличие от традиционной синтаксической, которая может нарушатся для русского языка в устной речи или не определена для языков с полностью свободным порядком слов (австралийский язык вальбири [Я. Тестелец, 2001]). Связи между уже построенными полными сегментами (рис.5) также проективны. Такое свойство схемы объединения и свойство структуры полных сегментов будем называть сегментной проективностью [Т.Ю.Кобзарева, 2002]. Гипотетически сегментная проективность не может нарушатся ни в одном из существующих естественных языков, но для доказательства этого утверждения необходимо проводить типологическое исследование.

Таким образом, можно вывести три простых структурных ограничения на сегментации:

1. Сегментная проективность.
2. β-отрезок может быть присоединен к α-отрезку только в том случае, если он находится справа от α в линейной последовательности предложения.
3. Два α-отрезка не могут быть объединены; в случае сочинения сегментов (сочиненные цепочки придаточных или причастных оборотов, или простых сегментов в составе сложного предложения, etc.) сохраняется независимость каждого сегмента в цепочке сочиненных как отдельной единицы анализа.

Метод монтажа

Методом монтажа (в соответствии с приведенной аналогией) будем называть α- и β-анализ, опирающийся на свойство рекурсивности сегментов и определенные выше структурные ограничения. α-анализ, являясь центральной и алгоритмически наиболее сложной стратегией сегментации [Т. Кобзарева и др., 2002], предшествует β-анализу и позволяет ликвидировать обусловленные вложением α-сегментов разрывы простых β-сегментов в составе сложного предложения S. Алгоритм α-анализа идет справа налево по S, осуществляя поиск α-отрезков, и заново вызывается рекурсивно для текущего α-отрезка после каждого нового присоединения правостоящего β-отрезка, что отражает

свойство рекурсивности сегмента в стратегии алгоритма. Поиск справа налево каждого следующего α-отрезка в S дает возможность начинать сборку α-сегментов с вложений максимальной глубины. Возможны две ситуации при рассмотрении найденного правостоящего β-отрезка:

1. β-отрезок находится непосредственно (контактно) справа от α-отрезка;
2. между α- и β-отрезками находятся один и более (число вложений не ограничено) полных (уже обработанных) α-сегментов.

Определяются грамматические условия для присоединения β-отрезка [Т. Кобзарева и др., 2001]:

а) синтаксическая неполнота α-отрезка (α-incompleteness);

б) синтаксическая неполнота β-отрезка (β-incompleteness);

в) в α-отрезке существует слово W_1, способное управлять словом W_2 в β-отрезке (α-manage);

г) слово или группа слов в α-отрезке образует сочинительную связь со словом или группой слов в β-отрезке (coordination);

В первой ситуации – контактного расположения β- относительно α-отрезка – допустимо использование только условия сочинения (coordination) для присоединения β-отрезка, при этом накладываются дополнительные грамматические ограничения (constraints) на поиск сочинения.

Представим множество исходных α- и β- отрезков на предложении S в виде вектора $V = \{Sg_1, Sg_2, .., Sg_n\}$, где, например, для S (“девочка, решив…, засмеялась”), заданного $V = \{Sg_1=\beta_1, Sg_2=\alpha_1, Sg_3=\alpha_2, Sg_4=\beta_2, Sg_5=\beta_3\}$, $V[3] = \alpha_2$, $V[5] = \beta_3$, etc. Тогда запишем алгоритм α-анализа псевдокодом [Т. Кормен и др., 2001, стр. 20]:

```
α-Analyse
1  for i ← length[V] downto 1
2      do if V[i] = α
3          then V[i] ← Montage(V, V[i], i, i+1)

Montage(V, α, i, j)
1  if j ≤ length[V]
2      then if V[j] ≠ β
3              then α ← Montage(V, α, i, j+1)
4              else if ( j = i + 1 and coordination(α, V[j], constraints) )
                       or (
                            j > i + 1
                            and (
                                    α-incompleteness(α)
                                    or (
                                         β-incompleteness(V[j])
                                         and (
                                                α-manage(α, V[j]) or  coordination(α, V[j])
                                         )
                                     )
                             )
5                      ) then α ← α ⊕ V[j]
6                              delete(V[j])
7                              α ← Montage(V, α, i, j)
8  return α
```

Учитывая рекурсивный характер задачи построения α-сегментов, в алгоритме α-анализа, записанного псевдокодом, для большей наглядности используется рекурсивный вызов функции Montage. Очевидно, что для эффективной программной реализации подобного типа рекурсию можно и необходимо переводить в итеративную форму, используя while-цикл [Н. Вирт, 2001]. Приведем результат работы процессора в ходе α-анализа сложного предложения, содержащего разрывные сегменты с α-вложениями, на рис.6:

“Когда, увидев в зеркале, принадлежавшем, как говорил брат, отцу, свое заплаканное лицо, Мария схватила письмо, лежавшее на столе, и зажгла свечу, в комнату вошел Иван.”

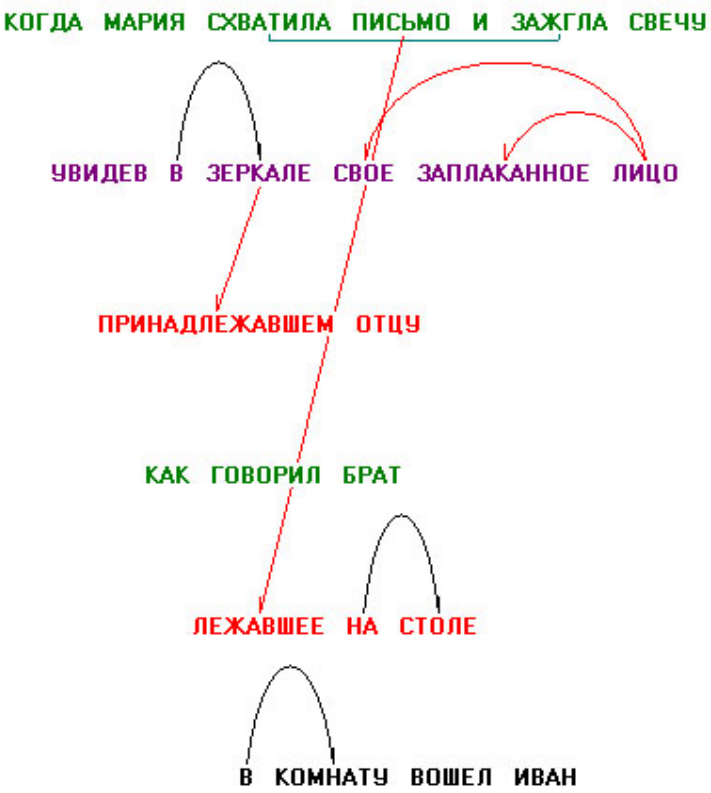

рис.6

После завершения α-анализа построенные полные α-сегменты “изымаются” из линейной последовательности S, вследствие чего на вход β-анализу поступает вектор V, содержащий последовательность β-отрезков, оставшихся непроанализированными или неприсоединенными к α-сегментам. Так, для S (“девочка, решив…, засмеялась”) вектор V на входе β-анализа принимает вид V = {$Sg_1=\beta_1$, $Sg_2=\beta_3$}. В упрощенном виде алгоритм β-анализа [Т. Кобзарева, 2002] можно представить в виде последовательности итераций: (а) процедура поиска неморфологического предиката (НМП) в β-отрезках; (б) установление границ между подгруппами последовательно расположенных β-отрезков, при условии фиксации НМП или постановки ‘;’ (в определенных случаях ‘:’) между отрезками; (в) объединение внутри подгрупп контактно расположенных β-отрезков, при условии синтаксической неполноты одного из двух β-отрезков и/или сочинения именных составляющих или предикатов двух β-отрезков. Приведем результат работы процессора в ходе β-анализа сложного предложения, содержащего два простых сегмента, разорванных α-вложениями, на рис.7:

“Едва уловимую особенность, отличавшую его сына от всех тех детей, которые по его мнению должны были стать людьми, ничем не замечательными, он

понимал как тайное волнение таланта, и, твердо помня, что покойный тесть был композитором, он в приятной мечте, похожей на литографию, спускался ночью со свечой в гостиную, где вундеркинд в белой рубашонке до пят играет на огромном черном рояле.” (В. Набоков)

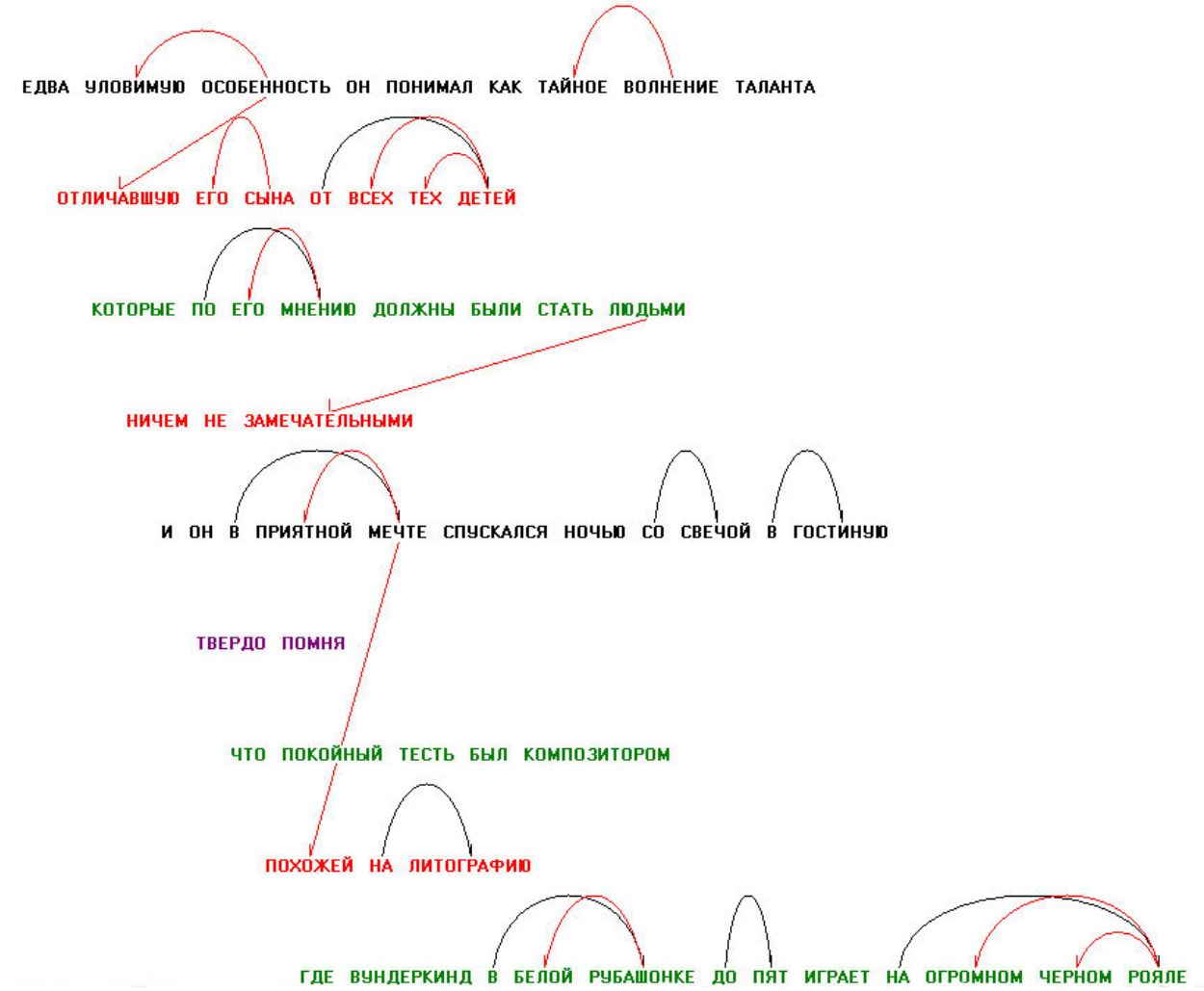

рис.7

Метод активизации омонимов

Активизация морфологического омонима, которая возникает в тех случаях, когда хотя бы один из омонимов словоформы $h \in W$ не отвечает проверяемому условию/ограничению или не способен образовать строящуюся синтагму, порождает отдельный граф синтагм G, состоящий из узлов новой смешанной цепочки типа S''. Интерпретация S'' на уровне синтагм всегда однозначна. В точке выбора порожденный граф наследует текущее состояние своего родителя, копируя ранее построенные синтагмы. Новый граф выделяется в отдельный поток, где процедура анализа продолжается из точки выбора. В

зависимости от прикладной системы, использующей модель сегментации, потоки могут работать параллельно или последовательно.

Активизация синтаксического омонима возникает на этапе построения α- и β- сегментов. В точке выбора порождается граф сегментов GS, что создает множественность интерпретаций для графа синтагм G на уровне сегментов. Как и в случае морфологической омонимии, порожденный граф наследует текущее состояние своего родителя, копируя ранее построенные сегменты, и новый граф выделяется в отдельный поток.

Метод активизации омонимов состоит из следующих понятий, определенных в процессоре:

- Условие/ограничение: в ходе работы алгоритмов-стратегий (PRN-NRA модуль, α-анализ, β-анализ, etc.) в пределах одного графа синтагм или сегментов проверяются условия/ограничения для некоторых элементов e_{ik} и e_{jm} цепочки S''типа: e_{ik} и e_{jm} согласованы или между e_{ik} и e_{jm} ∃ предикат (e_{xy} со значением части речи p, где p ∈ Предикат = [финитная ф. гл., кр. прил., кр. прич., предикатив]). Условия/ограничения, используемые стратегиями построения синтагм и сегментов, могут определять как морфологическую, так и синтаксическую омонимии. Подтвержденные в ходе анализа условия/ограничения, способные задавать синтаксическую омонимию, называются многозначными.
- Синтагма: построение синтаксического отношения $R(e_{ik}, e_{jm})$, каковым является NRA, PRN, LRN, etc. Синтагма, при определенных условиях/ограничениях (интерпозиция: "…сыном, осужденным отцом, …" R(сыном, осужденным) vs. R(отцом, осужденным)), определяет синтаксическую омонимию на графе синтагм.
- Событие: событие возникает в системе в результате подтверждения условия/ограничения или построения синтагмы.
- Точка выбора: событие в системе происходит в некоторой точке процедуры анализа, которая может быть определена в любом из модулей процессора; в случае порождения омонимичного графа такая точка выбора служит координатами места в процедуре анализа, с которого будет начат анализ нового графа.

- Состояние: текущем состоянием графа называется множество его узлов и связей, состояние графа фиксируется в точке выбора.
- Класс эквивалентности: класс эквивалентности $[h]_p$, где p – отношение эквивалентности на множестве омонимов словоформы $W \in S$, h – омоним словоформы W, состоящий из пары значений pos – часть речи и GR – множество граммем, тогда $p = \{(h_i, h_j): h_i, h_j \in W; h_i=\{pos_i, GR_i\}; h_j=\{pos_j, GR_j\}; h_i: (pos_i \in X) \& (Y \cap GR_i \vee Y = \varnothing)$ и $h_i \equiv h_j$, если и только если для h_j справедливо $(pos_j \in X) \& (Y \cap GR_j \vee Y = \varnothing)\}$; X и Y – множества, заданные инициировавшим событие условием/ограничением. Пример 1: между элементами e_{ik} ($e_{ik} := h_k \in W_i$) и e_{jm} ($e_{jm} := h_m \in W_j$) установлено согласование по грамматическому числу и падежу; известно, что для e_{jm} допустимы следующие значения селективных признаков POS = {существительное, местоимение} и согласование определено по одному из возможных значений граммем GR = {{мн., им.}, {мн., вн.}}, тогда условием/ограничением, подтвердившим согласование, формируется множество X = POS и Y = GR для отношения эквивалентности p; предположим, словоформа W_j состоит из трех омонимов $W_j = \{h_1=\{$сущ., {мн., им.}}, $h_2=\{$сущ., {мн., вн.}}, $h_3=\{$гл., $GR_3\}\}$; пусть для h_m m = 2, тогда $f_p(h_m) = [h_m]_p = \{h_1, h_2\}$. Пример 2: допустим, что в цепочке S'' найден предикат e_{jm} ($e_{jm} := h_m \in W_j$), тогда условием/ограничением, осуществлявшим поиск предиката, формируется множество X = Предикат и $Y = \varnothing$ для отношения эквивалентности p; предположим, словоформа W_j состоит из четырех омонимов $W_j = \{h_1=\{$ кр. прил., $GR_1\}$, $h_2=\{$наречие, $GR_2\}$, $h_3=\{$предикатив, $GR_3\}$, $h_4=\{$частица, $GR_4\}\}$; пусть для h_m m = 3, тогда $f_p(h_m) = [h_m]_p = \{h_1, h_3\}$.
- Функция разбиения: аргументом функции является класс эквивалентности $[h]_p$ и множество W; если $B = W \setminus [h]_p$ и $B \neq \varnothing$, то вызвать функцию клонирования, иначе завершить обработку события; $[h]_p$ будем также называть множеством W', а B – множеством W''.
- Функция клонирования: аргументом функции служат точка выбора, в которой возникло событие, и состояние графа в данной точке; функция клонирования изменяет множество узлов или связей исходного графа и порождает омонимичный вариант исходного графа, который добавляется в

конец списка L или L’ для графа синтагм или сегментов соответственно. В точке выбора отличие между исходным и клонированным вариантами графа - незначительное, но по окончании процедуры анализа для каждого из вариантов различие в узлах и связях, как правило, становится существенным. Так, морфологическая омонимия на графе синтагм может повлиять на узлы зависящего от него графа сегментов, т.е. на границы построенных сегментов.

Таким образом, принцип ленивых вычислений, алгоритмически мотивированный условиями/ограничениями, реализуется в системе через событие и обработку каждого такого события. В процессоре зафиксировано три сценария обработки события:

(1) Событие[$G=(S'', E)$, условие/ограничение] $\Rightarrow$ Функция разбиения($[h]_p$, W) $\Rightarrow$ Функция клонирования(точка выбора, состояние $G=(S'', E)$) == $G=(S_x'', E)$ и $G'=(S_y'', E_y)$, где $W' \subset S_x''$, $W'' \subset S_y''$ и $E = E_y$.

(2) Событие[$G=(S'', E)$, синтагма syn] $\Rightarrow$ Функция клонирования(точка выбора, состояние $G=(S'', E)$) == $G=(S'', E_x)$ и $G'=(S_y'', E_y)$, где $syn \in E_x$, $syn \notin E_y$ и $S'' = S_y''$.

(3) Событие[GS=(ST, SE), многозначное условие/ограничение] $\Rightarrow$ Функция клонирования(точка выбора, состояние GS=(ST, SE)) == $GS=(ST_x, SE)$ и $GS'=(ST_y, SE_y)$, где $ST_x \neq ST_y$ и $SE = SE_y$. Синтаксическая омонимия на графе сегментов возникает при неоднозначности присоединения некоторого β-отрезка. Например, для исходного графа GS узел $(\alpha \oplus \beta) = \alpha'$ и $\alpha' \in ST_x$, а в порожденном GS’ узлы $\alpha, \beta \in ST_y$ (ситуация синтаксической омонимии на границах сегментов была показана на рис.2 и 3).

Основная идея изложенного метода активизации омонимов состоит в том, чтобы избежать (в тех случаях, когда это возможно) полного декартова произведения морфологических омонимов ($W_1 \times \ldots \times W_n$) и повторного построения общих для омонимичных графов синтагм и сегментов.

Общая схема реализации анализатора

В приведенной ниже схеме (рис. 8) отражено взаимодействие модулей в программной реализации автоматической синтаксической сегментации. Каждый модуль на схеме реализует лингвистическую стратегию или механизм

управления анализом в процессоре. Механизм управления отвечает за построение графов синтагм и сегментов, применение метода активизации омонимов и формирование потоков в процессе, также в механизм управления включены программные библиотеки, реализующие общие для всех лингвистических модулей функции (проверка полного и частичного согласования или грамматического управления, проверка общих структурных ограничений и т.д) и сценарии обработки события. Стрелка вида X → Y на схеме означает, что модуль Y может быть вызван модулем X в качестве подпрограммы.

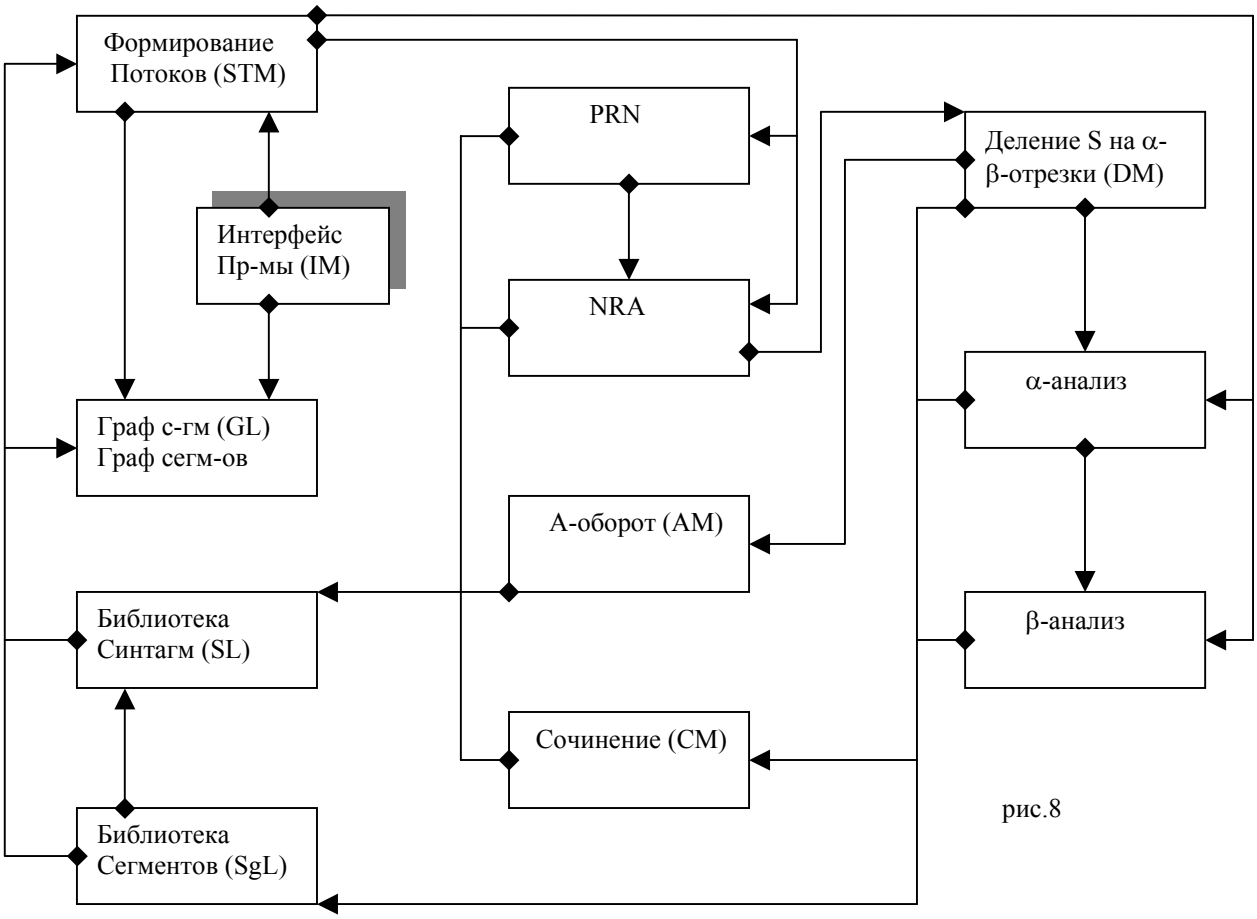

рис.8

Входными данными программы (модуля IM) является текст. Каждое законченное предложение текста, выделенное знаками пунктуации, проходит предварительную графематическую и морфологическую обработку. IM для каждого отдельного предложения инициализирует граф синтагм, заполняя исходными терминальными единицами (результатами обработки) его внутренние структуры данных для хранения узлов графа, после чего открывает

центральный поток (модуль STM). Построение графа синтагм и сегментов анализируемого предложения осуществляется в последовательности PRN→NRA→DM→α-анализ→β-анализ, базовых грамматических стратегий анализа. В процессе анализа внутри центрального потока через библиотеки SL и SgL формируются новые потоки в STM. Каждый открывающийся поток является результатом успешного сценария обработки события и принимает при инициализации клонированный граф синтагм или сегментов. Каждый поток в системе использует общую процедуру анализа, в соответствии со схемой, и может создавать неограниченное число новых потоков в процессе работы. Стрелки STM→PRN, STM→NRA, STM→α-анализ и STM→β-анализ на схеме демонстрируют с точностью до модуля местоположение точек выбора, с которых может начинаться процедура анализа в очередном потоке. Дадим функциональные характеристики модулей механизма управления: STM – формирование и инициализация потоков; GL – объектно-ориентированная библиотека классов, позволяющая строить графы синтагм и сегментов и предоставляющая набор методов (процедур и функций) для работы с ними; SL реализует общие лингвистические функции (проверка согласования, управления), а также 1 и 2 сценарии обработки события; SgL реализует структурные ограничения на сегментации и 3 сценарий обработки события. Дадим функциональные характеристики лингвистических модулей [Т. Кобзарева и др., 2000]: PRN и NRA проводят синтагматический анализ в процессоре; AM осуществляет анализ обособленных согласованных оборотов в предложении; CM – поиск грамматического сочинения; DM – деление предложения на первоначальные α- и β-отрезки и классификация α-отрезков; α-анализ и β-анализ реализуют синтаксическую “сборку” α- и β-сегментов из первоначальных отрезков. Использование потоков повышает устойчивость работы программы и позволяет эффективно распределять вычисления на серверах с многопроцессорной архитектурой.

Процессор синтаксической сегментации обладает линейной зависимостью скорости анализа от количества слов в предложении (в отличие от LinkParser, где зависимость - экспоненциальная). Для отладки работы и развития анализатора используется корпус тестов, состоящий из 230 сложных предложений. Такой корпус позволяет контролировать влияние вносимых в

процессор изменений на результаты анализа. Приведем значения характеристик такого контроля за системой на примере пяти правильно построенных предложений (т.е. предложений, для которых программой были построены только синтаксически допустимые варианты синтагматических и сегментационных интерпретаций).

Предложение	Кол-во слов (включая знаки препинания)	Кол-во сегментационных вариантов (мотивированное синтаксической омонимией)	Кол-во синтагматических вариантов (мотивированное морфологической омонимией)	Кол-во возможных вариантов (декартово произведение омонимов)	Время анализа (в секундах)
Нелепая провинциальная дама, которая раздражала друзей утверждением, что паровозы, пароходы и прочие новшества изобретены ее сыном, приводила всех в неистовство, деликатно намекая, что он сочинитель каждого прочитанного ею романа.	36	2	4	48	0,3
И потом до самого разъезда мы ни о чем не потолковали, не сговаривались насчет будущих**, в** даль тронувшихся пятнадцати дорожных лет, нагруженных частями наших несобранных встреч, и следя за ней в лабиринте жестов и теней жестов, из которых состоял вечер, я был поражен ее невниманием ко мне, чистосердечнейшей естественностью этого невнимания, ибо я еще тогда не знал, что, если бы сказал я два слова, оно сменилось бы тотчас чудной окраской чувств, веселым, добрым, по возможности деятельным участием, точно женская любовь была родниковой водой, содержащей целебные соли, которой она из своего ковшика поила всякого, если только напомнить.	115	1	8	96	1,3
Железнодорожная проза, как дамская сумочка этого предсмертного мужичка, полна инструментами сцепщика, бредовыми частичками, скобяными предлогами, которым место на столе судебных улик, развязана от всякой заботы о красоте.	34	1	2	4	0,2
Девочка, решив уже, когда ее позвали, задачу, засмеялась.	13	1	1	4	0,001
Участники российских финансовых рынков, продавая рубли, старались минимизировать возможные	31	1	1	1	0,001

негативные последствия углубления финансового кризиса, которые, как свидетельствует мировой опыт, проявляются в резком обесценении национальной валюты.					

Заключение

Метод монтажа и метод активизации омонимов лингвистически адекватны и универсальны, т.е. независимы от анализируемого естественного языка. Адекватность понимается как соответствие модели процессора трем сформулированным принципам: описательному, объяснительному и эмулирующему.

Программная реализация процессора выполнена на языке Object Pascal с использованием C библиотек, система анализа протестирована на пятистах сложных предложениях. Взаимодействие между морфологическим и синтаксическим модулями в программе организовано через текстовый файл заданного формата, выходные данные сегментационного процессора также представляются в виде текстового файла. Настоящий процессор, в первую очередь, рассматривается как экспериментальное пространство для создания промышленных систем синтаксического анализа.

Все рассмотренные в настоящей работе процессоры созданы в течение последних 10-12 лет. Можно выделить три доминирующих подхода к проектированию моделей синтаксического анализа естественного языка: лексикализм (HPSG), контекстно-свободные грамматики (LinkParser) и алгоритмический подход. Последний характеризуется разделением на уровни лингвистического анализа и модульностью системы. Алгоритмический подход состоит из двух направлений: основу первого составляют правила (процессор Диалинг), а второго – грамматические стратегии (анализатор ОИС). STP скорее относится к алгоритмическому процессору смешанного типа. Если основным критерием для построения и оценки правильности синтаксической структуры предложения в лексикализме и CFG служит связность графа, то алгоритмический подход, возвращаясь к шахматной аналогии, оперирует «фокусным пространством», выделяя «куски-ситуации», соответствующие сегментам, каждый из которых содержит явный или скрытый предикат. Модели типа LinkParser подразумевают жесткий порядок слов в предложении и морфологическую простоту анализируемого языка. Унифицирующие

грамматики целиком зависят от полноты лексикона и выверенности каждой из его лексических статей. Модульные анализаторы, стараясь использовать наиболее общие синтаксические законы языка, дают возможность снизить зависимость анализа от словаря и значительно сократить затраты на разработку лингвистического обеспечения. Теряя, в определенном смысле, прозрачность архитектуры процессора и его программной реализации, алгоритмическая модель часто позволяет избежать избыточности вычислений при построении синтаксической структуры и лучше поддается контролю за принятием решений в процессе анализа, что отвечает принципам проектирования систем ИИ.