ЧАСТЬ III. НАЧАЛЬНЫЕ ЭТАПЫ АНАЛИЗА ТЕКСТА *(КЛЫШИНСКИЙ Э.С.)*

Глава 1. Этапы анализа текста

Самые большие возможности и высокое качество анализа текстов можно получить, проведя его полный анализ. Однако сложности, возникающие при создании подобного анализа таковы, что на практике до сих пор не реализованы все теоретические положения, разработанные на данный момент. Основными проблемами здесь являются сложность синтаксического анализа текста и сложность создания полноценной экспертной системы, реализующей полноценную модель окружающего мира. Сложность анализа текста заключается в том, что текст эллиптичен, неполон и насквозь пронизан умолчаниями. Ярким примером может служить китайский театр, в котором человек, который при ходьбе выбрасывает в стороны несгибающиеся ноги и поглаживает бороду, воспринимается как положительный гражданский герой, тогда как «обольстительная красавица» должна переступать плотно сжав колени. Аналогично и в тексте встречаются конструкции, предназначенные скорее для живого воображения, чем для формальной обработки: «давить мух», «сделать ноги». Анализ подобных текстов может составить серьезную проблему не только для ЭВМ, но и для человека, так как большинство ситуаций имело под собой какую-то реальную или вымышленную основу и вставка их в текст служит как бы ссылкой на такую ситуацию (хотя зачастую большинство может уже и не помнить о чем идет речь, а просто восстанавливает истинный смысл фразы).

Для полноценной работы система анализа текста должна иметь возможность проанализировать тест, поданный пользователем на вход, с точки зрения синтаксиса (структуры предложений), семантики (понятий, применяемых в тексте) и прагматики (правильности употребления понятий и целей их употребления). Далее система должна сгенерировать свой отклик во внутреннем представлении, пригодном для логического вывода, и просинтезировать свой отклик на естественном языке.

В целом система, поддерживающая полный анализ, должна содержать в себе следующие модули.

Графематический анализ – обеспечивает выделение синтаксических или структурных единиц из входного текста, который может представлять собой линейную структуру, содержащую единый фрагмент текста. Однако в более общем случае текст может состоять из многих структурных единиц: основного текста, заголовков, вставок, врезок, комментариев и т.д. При машинном переводе ставится задача сохранить подобную структуру текста. Однако в случае диалоговых систем обычно используется первый вариант (без вставок). Но и в этом случае графематический анализ должен выделять синтаксические единицы: абзацы, предложения, отдельные слова и знаки препинания. В ряде случаев здесь же проводится предморфологический анализ – объединение неразрывных неизменяемых словосочетаний в одну единицу: «_что_-_то_», «_таким_образом_», «_и_так_далее_», … .

Морфологический анализ – обеспечивает определение нормальной формы, от которой была образована данная словоформа, и набора параметров, приписанных данной словоформе. Это делается для того, чтобы ориентироваться в дальнейшем только на нормальную форму, а не на все словоформы, использовать параметры, например, для проверки согласования слов.

Предсинтаксический анализ отвечает за две противоположные задачи: объединение отдельных лексических единиц в одну синтаксическую или, наоборот, ее разделение на несколько. В одну синтаксическую единицу объединяются изменяемые неразрывные словосочетания (например, «бить баклуши»). Делением слов особенно необходимо заниматься, например, в немецком языке, где несколько произвольных связанных между собой слов могут объединяться в одно сложное «на лету», а помещать в морфологический анализ все подобные сочетания не представляется возможным. Еще одной задачей предсинтаксического анализа является проведение синтаксической сегментации. Её задачей является разметка линейного текста на фрагменты, привязанные правилам следующего этапа – синтаксического анализа, который является задачей с экспоненциальным ростом сложности. В связи с этим любая помощь при его проведении может привести к существенному ускорению его работы.

Синтаксический анализ – самая сложная часть анализа текста. Здесь необходимо определить роли слов и их связи между собой. Результатом этого этапа является набор деревьев, показывающих такие связи. Выполнение задачи осложняется огромным количеством альтернативных вариантов, возникающих в ходе разбора, связанных как с многозначностью входных данных (одна и та же словоформа может быть получена от различных нормальных форм), так и неоднозначностью самих правил разбора.

Постсинтасический анализ служит двум целям. С одной стороны нам необходимо уточнить смысл, заложенный в слова и выраженный при помощи различных средств языка: предлогов, префиксов или аффиксов, создающих ту или иную словоформу. С другой стороны, одна и та же мысль может быть выражена различными конструкциями языка. В случае с многоязыковой диалоговой системой, одну и ту же мысль можно выразить различными синтаксическими конструкциями. В связи с этим дерево необходимо нормализовать, т.е. конструкция, выражающая некоторое действие различным образом для различных языков или ситуаций, должна быть сведена к одному и тому же нормализованному дереву. Кроме того, на этом же этапе может проводиться обработка разрывных изменяемых словосочетаний, в которых слова словосочетания могут изменяться и могут быть разделены другими словами («белый офицер» vs «белый корниловский офицер»).

Семантический анализ проводит анализ текста «по смыслу». С одной стороны, семантический анализ уточняет связи, которые не смог уточнить постсинтаксический анализ, так как многие роли выражаются не только при помощи средств языка, но и с учетом значения слова. С другой стороны, семантический анализ позволяет отфильтровать некоторые значения слов или даже целые варианты разбора как «семантически несвязные».

Этапом семантического анализа заканчивается анализ входного текста. Последующие этапы требуются для генерации отклика, например, в ходе диалога с пользователем или при переводе документов с иностранного языка для их дальнейшей обработки аналитиком. Сам отклик может, например, выбираться из некоторого корпуса текстов или генерироваться «на лету». В случае генерации ответа необходимо провести следующие этапы синтеза.

Генерация внутреннего представления отклика. Прежде, чем давать какой-либо отклик, диалоговая система должна сформулировать ответ. Для этого ей, например, может потребоваться собрать и проанализировать какую-то информацию. Отклик

системы будет зависеть от состояния диалога и других параметров. После этого необходимо определить форму ответа (или вопроса), подставить в него конкретные слова и значения и лишь затем приступать к синтаксическому синтезу текста отклика.

Предсинтаксический синтез. Задачи данного этапа прямо противоположны задачам постсинтаксического анализа. Здесь мы обязаны вернуть в предложение языкозависимые конструкции, пытаясь раскрыть роль слов средствами языка. В зависимости от контекста необходимо выбрать ту или иную форму выражения роли слов и основных идей предложения, расшифровать словосочетания, развернуть нормализованное дерево.

Синтаксический синтез превращает дерево предложения в линейный порядок слов. При этом осуществляется согласование параметров слов между собой.

Предморфологический синтез разъединяет слова, объединенные в целях экономии смысла в единую лексическую единицу. Здесь же может осуществляться обратная задача: слияния отдельных слов в одно, если того требуют правила языка.

Морфологический синтез по нормальной форме слова и его параметрам находит соответствующую словоформу.

Графематический синтез объединяет слова в единый текст, следит за соответствием фрагментов входного текста фрагментам выходного. На этом синтез отклика заканчивается.

Генерация отклика в разной мере присуща всем видам диалоговых систем, некоторым видам систем составления рефератов текста, статистического анализа текста, генерации текстов. Вопросно-ответные системы могут генерировать отклик как результат обработки запроса пользователя, системы общения обязаны делать это по определению, исполнительные системы могут комментировать происходящее или генерировать ответ на запрос пользователя. Но действия систем не ограничиваются только генерацией ответов. Вопросно-ответные системы должны сконвертировать запрос пользователя в какой-либо запрос на формальном языке (например, SQL при поиске в базе данных) и на основании полученных результатов решить, какой вид ответа необходимо выбрать. Исполнительная система должна определить алгоритм выполнения запроса пользователя и реализовать его. Но эти вопросы не входят сейчас в наше рассмотрение.

Кратко изложив последовательность этапов, в той или иной степени необходимых для обработки текста, рассмотрим теперь более подробно особенности реализации каждого из этих этапов.

Глава 2. Морфологический анализ и синтез

§ 2.1. Словарный морфологический анализ и синтез

Для того чтобы подчеркнуть различия в употреблении слов люди придумали формы слов или словоформы. Однако какова бы ни была словоформа, она выражает одно и то же понятие. Обсуждая понятие само по себе, принято использовать его нормальную форму – просто одну из словоформ, выделенную для обозначения понятия. Т.е., если у нас есть слово «мама», то для него существует несколько форм: мамы, маме, маму и т.д. К каждой форме приписывается ряд характеристик или параметров (род, падеж, число), характеризующих данную словоформу. Также каждому слову приписывается часть речи, показывающая, какого рода понятием мы оперируем. В речи мы привыкли к тому, что в данном месте должно стоять слово с заданной частью речи в определенной форме, но при машинной обработке подобные интуитивные рассуждения должны быть формализованы. Кроме того, подобное разнообразие вносит известные проблемы при анализе текста. Вместо того, чтобы работать с единственным словом, мы вынуждены обрабатывать все его словоформы. Для того чтобы избежать подобной ситуации были введены этапы морфологического анализа и синтеза.

Задачей *морфологического анализа* является определение по словоформе нормальной формы, от которой была образована данная словоформа, и набора параметров, приписанных к данной словоформе. При этом может оказаться, что одной словоформе может быть сопоставлено несколько таких пар.

Задача *морфологического синтеза* прямо противоположная. Здесь необходимо по нормальной форме и набору параметров получить словоформу.

Дадим более формальные определения, необходимые для рассмотрения этих этапов.

Нормальная форма слова – это форма слова (строка), принятая для обозначения понятия, связанного с данным словом. Обычно считается, что от нормальной формы образуются все остальные формы слова. Однако в таких случаях, как «идти – шел», связь между нормальной формой и словоформой не прослеживается. В связи с этим будем считать, что нормальная форма всего лишь одна из словоформ данного слова, выделенная согласно традиции данного языка. Словоформа – это форма слова (строка), связанная с нормальной формой слова и указывающая на особенности употребления данного слова. Будем считать, что словоформа характеризуется пятеркой – строкой словоформы; частью речи; нормальной формой, от которой была образована данная словоформа; частью речи нормальной формы; набором морфологических параметров, приписываемых к данной словоформе. Часть речи нормальной формы нам необходима, так как, например, деепричастие удобно считать формой глагола, а не выводить в отдельное слово.

Список основных частей речи в целом уже устоялся, хотя различные исследователи всё еще спорят о составе служебных частей речи. При реализации конкретного морфологического словаря важно с самого начала определиться с их списком, так как его изменение потом может оказаться дорогостоящей операцией. Для практических задач удобна любая из имеющихся логически обоснованных систем деления слов на части речи. В связи с этим мы не будем обсуждать здесь различные подходы к классификации слов.

Морфологический параметр – это пара <имя параметра, значение параметра>. Именем параметра может служить род, число, время, склонение, краткость формы прилагательного и другие признаки слов, принятые в данном языке. Значение параметра – это конкретное значение, которое может принимать данный признак. Так, например, падеж может быть именительным, родительным, местным, аккузативным; род может быть мужским, женским, средним; число – единственным, множественным, двойственным и т.д.

Параметры равны между собой, если равны их имена и значения. Параметры равны по имени, если совпадают их имена.

В ряде случаев значение параметра определить невозможно или в этом нет необходимости. Например, в русском языке существительным во множественном числе не приписывают род. Также существуют слова, которые имеют только форму множественного числа. Если словам, обладающим единственным числом значение рода может быть приписано из единственного числа, то слова, не обладающие единственным числом (очки, часы), такой информации лишены полностью. В этом случае будем считать, что значение параметра принимает фиксированное значение, обозначаемое «0». Примем, что параметр со значением «0» равен другому параметру, если равны их имена.

Подобный подход при хранении параметров может быть хорош в целом ряде случаев. Так, например, мы можем просто проверять наличие параметра с каким-либо значением. Это может пригодиться для того, чтобы убедиться, что параметр принял значение, отличное от заданного. Кроме того, мы можем приписать параметр слову для того, чтобы как-то выделить его среди других слов. В этом случае само наличие параметра у слова будет нести важную информацию.

Однако если нам необходимо просто провести морфологический анализ слов в тексте, может использоваться другой подход. Мы можем составить полный перечень всех значений параметров и дать им уникальные имена. В этом случае мы можем сэкономить место, так как хранится только имя параметра, сохраняя при этом различительную силу параметров. Но так как имена обычно даются символьные, то степень экономии зависит от фантазии разработчиков.

Вместо символьных имен параметров может использоваться цифровое представление. В этом случае мы можем создать справочник, в котором каждому символьному имени параметра будет сопоставлено некоторое уникальное число. При машинной обработке подобный подход позволит сэкономить место в памяти и ускорит процесс выдачи результатов. Заодно он объединит оба подхода, оставив людям уникальные и понятные для них символьные имена и предоставив компьютеру иметь дело с более удобным и компактным представлением.

При различных подходах слово «мама» может быть записано следующим образом.

мама	сущ., женск., одуш., единств., именит.
мама	сущ., род=женск., одуш=одуш., число=единств., падеж=именит.
12345	01 0202 0501 1101 1201

В последнем случае должен иметься справочник, указывающий, что существительному соответствует код 01, параметр род имеет код 02, а для него женский род кодируется числом 02 и т.д. Сопоставлением нормальной формы и ее кода занимается сам морфологический словарь. Так, если на анализ подается строка,

то на выходе будет числовой идентификатор, тогда как в синтезе информация преобразуется в противоположную сторону: по идентификатору можно получить строковую запись слова.

Набор параметров для частей речи фиксируется. Среди параметров слова выделяют словообразовательные и формообразовательные. Словообразовательные параметры не изменяются при изменении слова по формам. Так, например, слово «мама» остается женского рода в любой своей форме. Формообразовательные параметры изменяются при изменении слова по формам. Для приведенного примера падеж будет являться формообразовательным параметром. Обычно разделение на словообразовательные и формообразовательные параметры задается для всех слов, принадлежащих одной части речи. При этом словообразовательные параметры для одних частей речи могут являться формообразовательными для других. Например, параметр рода не меняется у существительных, однако будет образовывать формы у прилагательных и глаголов. Отнесение части параметров является спорной. Например, переходность глаголов может относиться как к словообразовательным, так и к формообразовательным параметрам, в зависимости от предпочтений разработчиков и их целей.

Формально морфологическая омонимия – это ситуация, когда одной словоформе можно приписать несколько кортежей, содержащих нормальную форму, части речи и набор параметров. Ниже приведены примеры таких ситуаций.

мамы	Мама	сущ., ж.р., ед. ч., род. п., одуш.
	Мама	сущ., ж.р., мн. ч., им. п., одуш.
стекло	Стекло	сущ., ср.р., ед. ч., им. п., неодуш.
	Стекать	гл., ср. р., ед. ч., 3 л., нн.ф., пр. вр.
дракон	дракон (животное)	сущ., м.р., ед. ч., им. п., одуш.
	дракон (корабль)	сущ., м.р., ед. ч., им. п., неодуш.
замок	замок (за́мок)	сущ., м.р., ед. ч., им. п., неодуш.
	замок (замо́к)	сущ., м.р., ед. ч., им. п., неодуш.

Приведенные примеры показывают различные виды омонимии. Подобная ситуация весьма распространена во многих языках, хотя и в разной мере. Отсутствие данного явления изрядно сократило бы количество шуток. В [7] приводится следующее определение омонимии: «Омонимами (гр. homos - одинаковый + onyma - имя) называются слова, разные по значению, но одинаковые по звучанию и написанию». При этом различают полные и неполные (частичные) омонимы. При полной омонимии слова принадлежат к одному грамматическому классу (у них одна часть речи) и все их формы совпадают. Примерами полной омонимии являются слова «дракон» (животное vs корабль), «кошка» (одушевленная vs неодушевленная), «коса» (прическа, отмель, инструмент). К неполной омонимии можно отнести слова, у которых совпадают лишь некоторые формы. Это, например, «закапывать», происходящее от слов «закапать» и «закопать», или приведенное выше «стекло». Выделяют и более частные явления. Так, полисемией называется свойство слов употребляться в разных значениях. В теоретической лингвистике считается, что слова полисемичны, если они сохранили некоторую логическую связь между собой: существуют некоторые аналогии, ассоциации, не потеряны исторические корни. Так, например, слово «ядро», как нечто центрообразующее, считается полисемичным

(пушечное ядро, ядро ореха, ядро армии), тогда как «коса» будет омонимичной (хотя и здесь можно найти общее – нечто тонкое и вытянутое). Кроме того, слова могут быть омоформами друг друга, если у них совпадают одно или несколько написаний, или омографами, когда у них совпадают написания, но различаются произношения («за́мок» vs «замо́к»).

Рассмотрев то, над чем работает морфология, перейдем к тем методам, с помощью которых она реализуется. Различают два вида морфологических словарей: словарные и бессловарные. Словарная морфология предполагает наличие словаря, в котором каждой словоформе сопоставлены нормальная форма и набор параметров. Т.е. у нас хранится полный словарь слов, и мы не можем проанализировать или просинтезировать слова, отсутствующие в словаре. Самым простым решением проблемы создания морфологического словаря является таблица, в которой в первой колонке будет записана словоформа, во второй – нормальная форма, а в третьей – набор параметров.

Для морфологического анализа в таком словаре необходимо просто найти все соответствующие словоформы и выдать найденные результаты. Для синтеза требуется найти заданную нормальную форму с требуемым набором параметров и выдать словоформу, находящуюся в той же строке (Рис. 2.1).

Однако при подобном подходе очень велики накладные расходы. Если предположить, что среднее слово будет занимать 8 байт, среднее количество параметров положить равным 4 и для обозначения параметров использовать 8 байт, то с частью речи одна запись в среднем будет занимать 48 байт. Предположим, что для каждой нормальной формы у нас имеется 8 словоформ. Тогда морфологический словарь объемом 100 тыс. слов будет занимать 36,6 Мб. Однако это весьма оптимистические предположения. Так, 8 словоформ были взяты просто потому, что это круглая цифра. В русском языке существительные имеют 12 форм, а прилагательные – 24; с другой стороны, наречия и предлоги имеют всего одну форму, но их количество уступает количеству тех же существительных. Максимальное количество словоформ может превышать 300 (если мы считаем деепричастия и причастия формами глагола), и среднее количество в этом случае составит около 25 форм на парадигму. На практике потребуется также место для индексов, в случае реляционной таблицы придется зарезервировать место не под среднюю длину слова, а максимальную и т.д. В итоге словарь в 100 тыс. слов (чего не достаточно для анализа текстов широкой тематики) будет содержать порядка 2–2,5 млн. входов против 800 тыс., которые были взяты для расчета. В связи с этим объем занимаемой памяти может вырасти на порядок – до 256 Мб без учета индексов. Кроме того, и время поиска будет пропорционально логарифму от объема базы, умноженному на среднюю длину слова.

Выходом может служить переход к дереву. Анализ словоформы в таком случае проводится побуквенно, начиная от корня дерева. В каждом узле хранится массив указателей на следующую вершину, причем каждый указатель отвечает за свою букву. Решение в лоб заключается в том, чтобы в каждом узле хранить массив длиной в размерность алфавита языка. Однако для русского языка для хранения всех последовательностей из 8 букв потребовалось бы более 46 млрд указателей. Большая часть таких вариантов будет отсеяна, так как, например, в русском языке нет слов, начинающихся на твердый или мягкий знак. Кроме того, массивы будут заполнены плотно только близко к корню дерева. Ближе к листовым вершинам массивы

становятся сильно разрежены. Это свойство можно использовать и после 2-4 уровней хранить два массива. Первый массив хранит список букв, для которых имеются указатели, а второй массив – собственно указатели в соответствующих ячейках. Кроме того, часть постфиксов слов будут уникальны для данной ветви и будут представлять собой линейную цепочку, то есть их можно хранить в виде строки, а не последовательности вершин дерева. На Рис. 2.2 показан пример дерева префиксов.

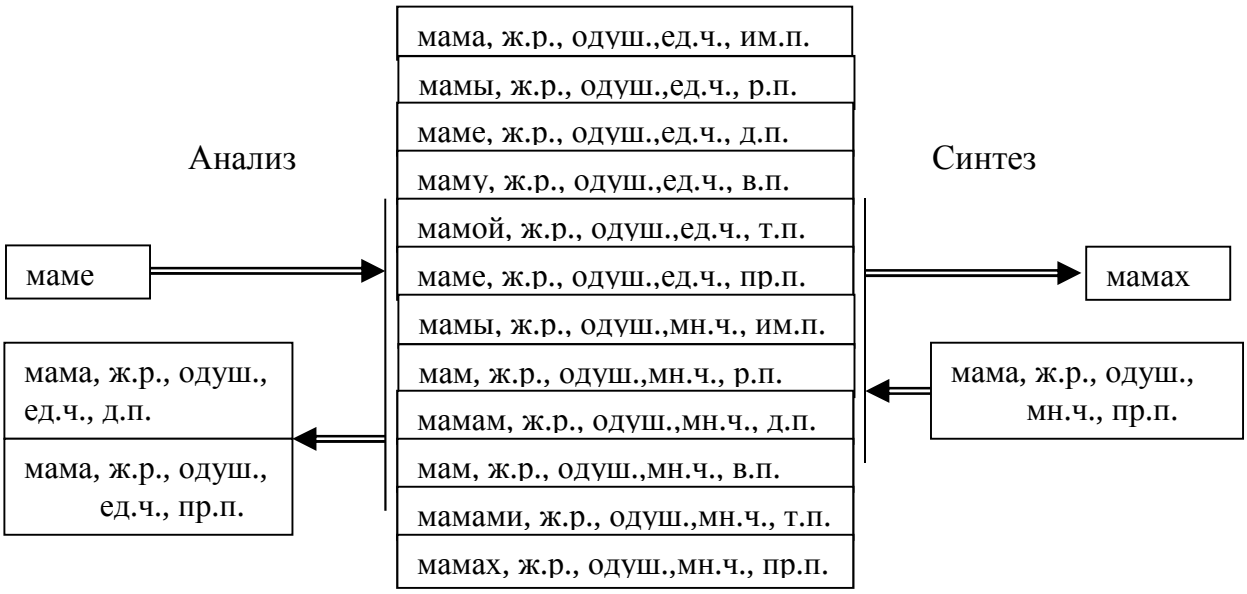

Рис. 2.1. Пример табличного морфологического анализа и синтеза

Кроме того, традиционно раздельно хранят деревья префиксов и постфиксов. Дело в том, что для большинства слов можно выделить неизменяемую часть – префикс – и набор постфиксов с привязанными к ним параметрами. Подобный набор постфиксов будем называть парадигмой изменения слова. Заметим, что мы не употребляем здесь слова «окончания», а именно постфикс, так как при изменении слова у него может появляться, исчезать или меняться не только окончание, но и суффикс. Частым явлением является изменение корневой буквы. Ярким примером является слово «идти», которое целиком попадет в изменяемый постфикс и будет иметь пустой неизменяемый префикс («идти» → «шел»). Кроме того, во многих сложносоставных словах меняется не только первое, но и второе слово. В этом случае в постфикс попадает вся часть слова, начиная с изменяемой части первого слова. Так, например, в фамилии Римский-Корсаков в постфикс попадет часть «ий-Корсаков».

Можно заметить, что слова группируются по парадигмам. Парадигма – это множество всех постфиксов и связанных с ними параметров для всех словоформ данного слова. Так, например, слова «лектор» и «завлаб» имеют одну парадигму. Мы можем хранить единственный набор ветвей в дереве постфиксов, сокращая тем самым занимаемый объем памяти.

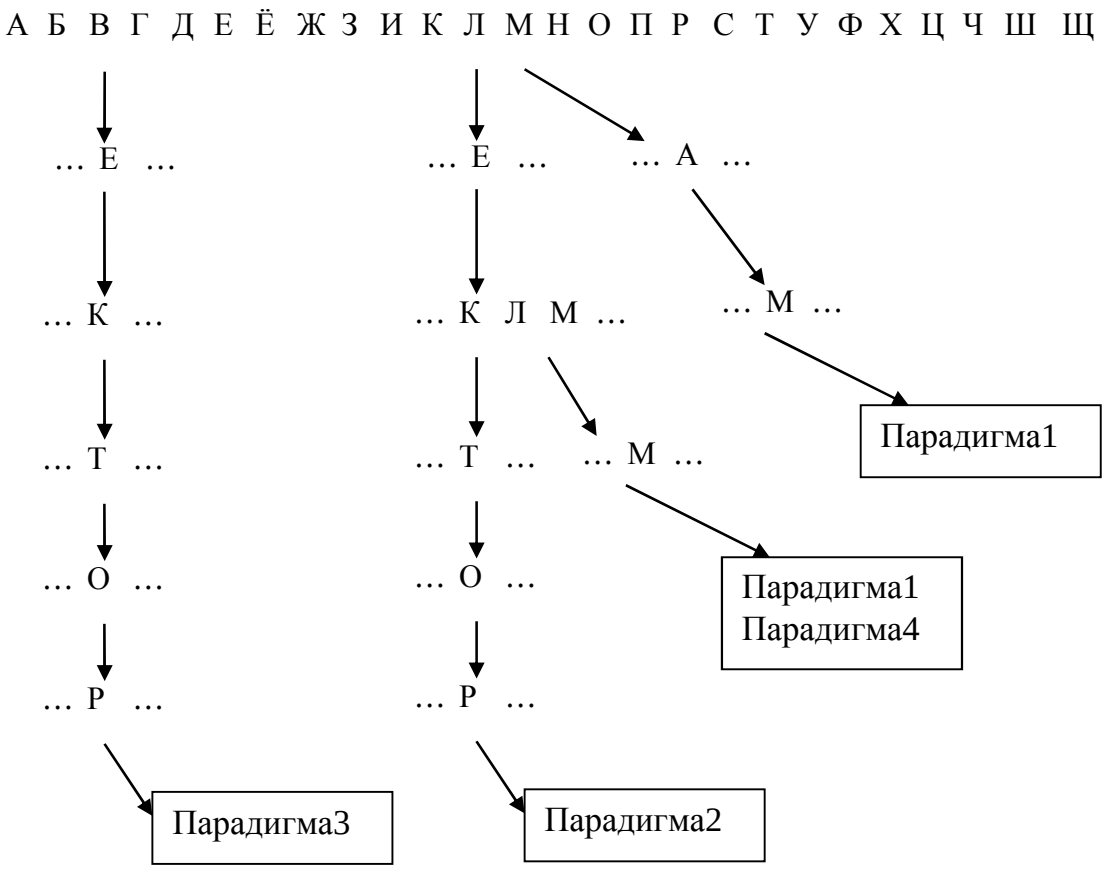

Рис. 2.2. Фрагменты дерева префиксов для слов «вектор», «лектор», «мама», «лемма»

Заметим, что вопрос о совпадении парадигм зависит от воли проектировщика. Так, слова «лектор» и «вектор» ни в коем случае не попадут в одну парадигму, так как будут иметь различные формы в винительном падеже. Однако слова «мама» и «лемма» попадут или не попадут в одну парадигму в зависимости от того, учитываем ли мы их одушевленность или нет. В первом случае парадигмы будут различными, так как для каждой формы слова «мама» будет прописан параметр «одушевленная» (хотя может существовать еще одно слово в морфологии, являющееся неодушевленным и означающее материнскую плату – его парадигма совпадет с парадигмой слова «лемма»), тогда как для слова «лемма» будет приписан параметр «неодушевленная». Во втором случае этих различий не будет, и слова будут принадлежать одной парадигме.

Конечная вершина дерева префиксов должна содержат информацию о том, какой постфикс или набор постфиксов соответствует данному префиксу. При этом указатель должен показывать на листовую вершину постфикса, в связи с тем, что развернуть его от корня будет несколько затруднительно. Заметим, что конечная вершина дерева префиксов не обязана быть листовой, так как в том месте, где закончилась основа одного слова, может продолжиться другое (например, «лук-ом» и «луков-ый»).

Заметим, что листовые вершины представленного дерева сходятся в парадигмах. Кроме того, часть путей в дереве могут совпадать, в связи с чем можно перейти от дерева к графу, объединив совпадающие части в одну ветвь. Такой граф уже можно назвать конечным автоматом, так как для каждого перехода в нем определен символ,

по которому осуществляется переход. Подобные и другие ухищрения позволяют существенно сократить объем памяти, занимаемой морфологическим словарем.

Морфологический анализ будет проходить следующим образом. Мы двигаемся побуквенно по строке, содержащей слово, перемещаясь при этом по дереву префиксов. Изначально в качестве текущей вершины выбирается корень дерева префиксов. Если переход из текущей вершины по очередной букве строки отсутствует, то разбор заканчивается. Если достигнута вершина, помеченная как конечная, то проводится проверка постфикса. При этом мы двигаемся по дереву постфиксов от листовой вершины к корню. Если корень был успешно достигнут, то информация из парадигмы изменения слова переносится в результат. Если в дереве префиксов не была достигнута листовая вершина, то движение по нему продолжается. В случае, когда множество результатов оказалось пустым, сообщаем о неуспешном анализе. В противном случае возвращаем множество результатов. Скорость анализа будет пропорциональна длине слова, а не объему словаря.

Для дерева постфикса возможен и другой вариант хранения. В этом случае мы храним префикс не с конца, а с его начала (движение будет производиться от корня дерева к листовым вершинам). При этом вершины деревьев префиксов и постфиксов хранят номер парадигмы. В этом случае, проведя анализ префикса, мы начинаем поиск постфикса от корня дерева. Если в конце слова была достигнута вершина с тем же номером парадигмы, что и в дереве префиксов, то слово считается успешно найденным.

Следует заметить, что скорость работы морфологического анализа будет сильно зависеть от задач, которые перед ним ставятся. Так, например, если у нас имеется поисковая система, задача которой найти все вхождения данного слова в документах, вне зависимости от формы слова, то нам вполне достаточно вернуть лишь нормальную форму слова. Если мы хотим ранжировать поиск в зависимости от совпадений морфологических параметров, или, как в нашем случае, морфологические параметры будут входить в критерий оценки, то для добавления параметров к результату и их обработки потребуется еще некоторое время. Кроме того, морфологический анализ может хранить и семантическую информацию, добавление которой к результату еще снизит скорость работы.

Морфологический синтез будет осуществляться следующим образом. В индексе нормальных форм находим все нормальные формы, для которых будет производиться синтез. Одной строке нормальной формы может соответствовать несколько слов со своими парадигмами, например, слово «кошка», имеющее одушевленную и неодушевленную формы, будет иметь разные постфиксы в винительном падеже множественного числа: «кошек» vs «кошки». Однако в данном случае ища, например, одушевленную кошку мы не обнаружим параметра «одушевленность» со значением «одушевленная» среди параметров неодушевленной кошки. Следовательно, она не попадет в результат. Аналогичные проблемы возникают и в других словах: «лист» – «листы» vs «листья»; «язык» – «языки» (часть тела) vs «языков» (язычник (древнерусское) или пленник).

Далее мы берем парадигму, соответствующую выбранной нормальной форме, находим нужный нам набор параметров, берем неизменяемую часть слова, присоединяем к ней постфикс, получая тем самым искомую словоформу. Помещаем ее в множество результатов.

При сравнении параметров может получиться так, что успешно сравнятся несколько наборов параметров. Это происходит потому, что как в множестве параметров, хранимых в парадигме, так и в множестве параметров, поступивших на вход, могут содержаться параметры с нулевым значением. Здесь следует помнить, что предпочтение следует отдавать полному совпадению параметров, т.е. желательно, чтобы значения параметров, имеющих на входе нулевое значение, в парадигме также имели нулевое значение. При наличии альтернативы лучше выбирать набор параметров, в котором большее количество параметров сравнилось точно. Еще одной проблемой при синтезе является неполный набор параметров, поступивший на вход. Это связано с тем, что мы не сумели выяснить полный набор параметров на предыдущих этапах. Такой вариант также необходимо предусматривать при реализации системы морфологического синтеза.

Так, например, если мы попытаемся сгенерировать родительный падеж единственного числа от слова «чай», то мы получим два варианта: «чая» и «чаю», которые оба являются морфологически верными и употребимыми. А попытка получить прошедшее время глагола, не уточнив предварительно его род, приведет к тому, что мы получим как минимум три варианта синтеза, так как прошедшее время глагола в русском языке не различается по лицам.

§ 2.2. Автоматизированное пополнение морфологического словаря

Автоматическое порождение гипотез о парадигмах изменения незнакомых слов является хорошей возможностью автоматизировать процесс заполнения баз. При переходе к новой предметной области встает вопрос о неполноте морфологического словаря. Каждая предметная область использует собственную лексику. В связи с этим встает вопрос о пополнении ею словарей. Данный процесс может быть автоматизирован, если имеющийся модуль морфологического анализа позволяет проводить предсказание лексических параметров незнакомых слов. Для этого необходимо выделить все слова, отсутствующие в имеющемся морфологическом словаре, и подвергнуть их анализу с предсказанием. Результатам анализа, как это отмечалось в соответствующем разделе, является кортеж словоформы $<\mathbf{f}_{nf}, r, \mathbf{P}_{const}(r,s) \cup \mathbf{P}_{var}(r,s)>$, где $\mathbf{f}_{nf}=<s_{nf}, r, \mathbf{P}_{var}(r,s)>$ - лексема нормальной формы, r – часть речи словоформы, s и s_{nf} – анализируемый токен (строка слова) и токен нормальной формы, а **P** – наборы параметров. По результатам анализа мы можем объединить все слова, обладающие одинаковыми токенами нормальной формы в единые гипотезы.

В ходе выдвижения гипотез можно использовать несколько сильных, но интуитивно верных положений.

1. Гипотезы, порожденные на основе редковстречающихся парадигм, в рассмотрение не брались. Под редковстречающейся понимается парадигма, по которой изменяется количество лексем не выше заданного порога.

2. Для словарных слов, принадлежащей одной парадигме, определяется список букв, заканчивающих их псевдоосновы. В случае если для словоформы выдвигается гипотеза о ее принадлежности к данной парадигме, и если при этом ее псевдооснова не оканчивается ни на одну из полученных букв, то такая гипотеза отвергается. Использование двух букв псевдоосновы позволяет проводить выбор с весьма высокой точностью.

3. Можно отсеивать гипотезы, образованные от словоформы, встретившейся единственный раз в исследуемом корпусе и являющиеся единственной словоформой,

использованной в данной парадигме, так как подобная словоформа скорее всего написана с ошибкой. Исключение можно делать для парадигм не изменяющихся слов (т.е. содержащих единственную позицию в парадигме).

4. Псевдоосновы несловарных словоформ, объединяемых в рамках одной парадигмы, должны содержать хотя бы один символ.

После кластеризации проводится отсеивание полученных лексем по критерию максимальной встречаемости словоформ, вошедших в лексему. Т.е. для каждого слова определяется, сколько раз оно встретилось в тексте. Далее эти значения суммируются по парадигмам и оставляются лишь парадигмы с максимальной сумой.

Получаемые результаты будут существенно зависеть от типа используемой морфологии. Для стемминга будут объединяться все слова, обладающие одной псевдоосновой. Так, в одну парадигму в зависимости от алгоритма выделения псевдоосновы могут попасть слова «компьютер», «компьютерный», «компьютеризация» и т.д. При использовании лемматизации результаты зависят от списка используемых параметров. При полном отсутствии таковых слова объединяются без образования альтернатив. Однако полный набор параметров создает проблемы. Среди прочего, это связано с тем, что в русском языке встречаются парадигмы, объединяющие один и тот же набор флексий, однако приписывающие им различные наборы параметров. Так, для слова «админ» можно породить лексемы, показанные на Рис. 3.3. Здесь «-» означает пустой постфикс. В скобках написаны словарные представители парадигмы. Из приведенных примеров видно, что даже один и тот же набор словоформ может быть различным образом размещен в различных парадигмах.

Единственное число	им.	род.	вин.	дат.	тв.	пр.
АДМИН (ТЕЛЕФОН) м.р., неодуш	-	А	-	У	ОМ	Е
АДМИН (ТОН) м.р., неодуш	-	А	-	У	ОМ	Е
АДМИН (БАЛ) м.р., неодуш	-	А	-	У	ОМ	Е/У
АДМИН (АКТИВИСТ) м.р., одуш	-	А	А	У	ОМ	Е
АДМИН (ОПЕР) м.р., одуш	-	А	А	У	ОМ	Е
Множественное число	им.	род.	вин.	дат.	тв.	пр.
АДМИН (ТЕЛЕФОН) м.р., неодуш	Ы	ОВ	Ы	АМ	АМИ	АХ
АДМИН (ТОН) м.р., неодуш	А/Ы	ОВ	А/Ы	АМ	АМИ	АХ
АДМИН м.р., неодуш	Ы	ОВ	Ы	АМ	АМИ	АХ
АДМИН (АКТИВИСТ) м.р., одуш	Ы	ОВ	ОВ	АМ	АМИ	АХ
АДМИН (ОПЕР) м.р., одуш	А/Ы	ОВ	ОВ	АМ	АМИ	АХ

Рис. 3.3. Пример неоднозначности предсказания слова по всем его словоформам

Большое количество ошибок, встречающихся в любых текстах, зашумляет выход системы лемматизации и требует длительного ручного труда по отделению корректных вариантов от ошибочных. К счастью, возможностей для ошибки предоставляется очень много, и поэтому большинство ошибок встречается один или два раза и отсеиваются на этапах фильтрации или кластеризации. Однако некоторые ошибочные словоформы могут войти в состав других парадигм, изменив тем самым результаты кластеризации не в лучшую сторону. Кроме того, у многих авторов существуют так сказать «любимые» ошибки, когда одна и та же ошибка допускается многократно в различных словоформах.

Однако даже небольшая автоматизация процесса предсказания парадигм несловарных слов позволяет существенно повысить производительность труда лингвистов в ходе формирования словарей. Кроме того, в ряде задач может использоваться не лемматизация, а, например, обсуждаемый ниже стемминг, в ходе которого лексические параметры указываться не будут. В этом случае необходимо сгенерировать (в том или ином виде) нормальную форму слова и указанные выше проблемы окажутся неактуальны. В этом случае возможно создание полностью автоматической процедуры пополнения словарей.

§ 2.3. Методы бессловарного морфологического анализа

Бессловарные морфологические словари появились во времена, когда оперативная память была существенно ограничена. Однако на данный момент несколько мегабайт или даже десятков мегабайт оперативной памяти не составляют проблемы, в связи с чем наибольшее распространение получили словарные морфологии. Существенным плюсом бессловарных морфологий является то, что ни могут предсказать морфологические характеристики практически любого слова, если его парадигма изменения попадает под одну из хранимых. Классическим примером здесь является предложение «Глокая куздра штеко будланула бокра и курдячит бокрёнка», предложенное одним из основоположников отечетственного языкознания академиком Л.В. Щербой еще около 1930 года при чтении курса лекций «Основы языкознания». Из этого предложения мы можем понять, что «куздра» имеет женский род, единственное число, именительный падеж и т.д. и разобрать синтаксис предложения, при этом совершенно не понимая, о чем идет речь. С другой стороны, наша уверенность в том, что куздра имеет женский род во многом основывается на результатах неявно проводимого синтаксического анализа, который говорит о том, что «глокая» является прилагательным в женском роде и согласуется со словом «куздра».

Бессловарные морфологии хранят парадигмы слов. При этом в парадигме в качестве постфикса может храниться только окончание. Часто в бессловарной морфологии может храниться набор приставок и суффиксов и привязанная к ним семантическая информация. Например, про суффиксы «-онок-» и «-ёнок-» будет написано, что их добавление обозначает детеныша животного, а приставка «при-» означает присоединение или приближение.

Анализ и синтез в бессловарных морфологиях ведется так же, как и в словарных, но без поиска по дереву префиксов и с учетом возможности выделения нескольких последовательно идущих постфиксов.

Однако такой подход часто приводит к существенным ошибкам. Так, например, «октябрёнок» может стать детенышем «октября» (что формально верно), «припевать» будет трактоваться как «приближение + певать» или «присоединение + певать», а «перебиваться» - возвратной несовершенной формой от «перебить», причем не ясно от какого из значений: прервать, уничтожить или прибить заново. При этом также не совсем понятно, какой вид возвратной формы будет иметься ввиду: перебивать себя или перебивать самому. При этом на самом деле возвратная форма от «перебить» будет подразумевать совсем иное значение – «обойтись без чего-либо», хотя вариант «перебить себя» представляется маловероятным, но не невозможным.

Для бессловарных морфологий существовали алгоритмы, позволявшие избегать этих ошибок. Так, например, для сочетания «пере+бивать+ся» можно в явном виде

прописать значение слова. Но в итоге мы получаем словарную морфологию со специальным алгоритмом архивирования базовых понятий, для которых нет исключений.

Также выделяют системы на основе стемминга. В случае стемминга зачастую отбрасывается вся морфологическая информация, а в качестве нормальной формы берется неизменяемая псевдооснова, называемая стем. Так, для слова «мама» стемом будет являться строка «мам». Именно эта основа и используется в дальнейшем для идентификации слова во всех его формах. Неудобство состоит в том, что для различных слов может порождаться один и тот же стем, например, «люб-овь» и «люб-ить». В случаях, когда необходимо различать эти понятия (например, при поиске слов в тексте), возможен единственный вариант – хранить информацию о части речи. В прямо противоположном случае, когда различать слова не обязательно, подобное совпадение может сослужить добрую службу. Однокоренные слова чаще всего относятся примерно к одному и тому же понятию («любить» означает «продуцировать любовь»). В связи с этим при сравнении текстов целиком такие понятия не будут размываться, а скорее наоборот – будут давать совместный вклад в результат сравнения. Однако в случае стемминга весьма вероятно смешение различных понятий. Так к стему глагола «люб-ить» будет отнесен и глагол «люб-оваться» (ведь у него есть форма «люб-уюсь»), что приводит к смешению различных понятий.

Для слов, подверженных флексии, т.е. замене букв в корне слова, берется несколько стемов. Так, например, для слова «шов» будет образовано два стема: «шов» и «шв», а для слова «идти» - «ид», «ше» и «шл». Это не позволяет идентифицировать их как одно со всеми вытекающими последствиями. Для решения этой проблемы создаются сложные парадигмы, объединяющие несколько стемов.

Собственно анализ в подобных системах будет проводиться аналогичным образом с лексической морфологией. Однако здесь возможны два варианта. В первом случае мы храним как стемы, так и парадигмы изменения и алгоритм анализа и синтеза не претерпевает никаких изменений. Во втором случае хранится только набор парадигм. В этом случае оставшаяся основа и будет являться искомым стемом. Из множества полученных стемов выбирается, например, самый короткий или самый длинный. Также применяется вариант, когда проводится анализ последних нескольких букв стема: наречия, имеющие пустое окончание, заканчиваются на «-о» или «-е», некоторые глаголы на «-ова-» или «-ева-» и т.д. Это также помогает отсеять ряд результатов.

Морфология на основе стемминга обладает рядом достоинств. Так, например, за счет упрощения алгоритма и уменьшения объема выдаваемой информации существенно (до нескольких раз) возрастает скорость анализа, а при использовании лишь массива парадигм сокращается объем хранимых баз. Главным достоинством морфологии на основе стемминга является тот факт, что при отсутствии словаря основ мы фактически получаем морфологическую базу неограниченного объема, настраиваемую непосредственно на имеющийся текст. Это очень удобно при создании информационно-поисковых систем с нефиксированной лексикой. В этом случае при индексировании текстов мы получаем некоторый набор стемов, которые и заносим в индекс. При этом морфология никогда не сообщает нам, что такого слова нет в словаре.

Однако подобный подход не лишен недостатков. Первым из них является невысокая точность метода. Так, например, стем «шл» будет соответствовать и глаголу «слать» («шлют» и омонимичное «шли»). Соответственно, при анализе мы объединим два этих разных слова в один «куст». В результате на информационный запрос пользователя о слове «шлют» будет выдана информация и о глаголе «идти». В зависимости от применяемого алгоритма могут быть выданы все формы для обоих глаголов. А в зависимости от слова в один «куст» могут быть объединены и слова различных частей речи. В ряде случаев это может оказаться весьма полезным, так как однокоренные слова обычно относятся к одним понятиям («грузчик» – «грузить») и, например, при информационном поиске начинают за счет этого объединяться в кластеры. Однако для исполнительных систем такой подход неприменим.

Следующим недостатком является невозможность морфологического синтеза на базе без основ. Справедливости ради следует заметить, что подобная задача необходима не во всех практических приложениях. И, как это было замечено выше, стемминговый подход не применим к таким приложениям, как исполнительные системы. Лишившись морфологической информации мы перестаем понимать, является ли слово действием, которое мы должны выполнить, или объектом этого действия, каким именно объектом действия является слово и т.д. Без подобной информации качественное выполнение действий невозможно.

Следует заметить, что грань между стемминговой морфологией, базирующейся на неизменяемой псевдооснове, и лексической морфологией, выдающей полный набор морфологических параметров и оперирующей с нормальными формами слова, довольно тонка. С одной стороны, лексическая морфология использует неизменяемую основу, т.е. стем. С другой стороны, при хранении полного набора лексической информации стемминг отличается от лемматизации лишь выдаваемой строкой нормальной формы. Так, система морфологического анализа MyStem компании Яндекс (http://company.yandex.ru/technology/mystem) хотя и называется стеммером (точнее – парсером), однако выдает полный набор лексической информации о слове. Аналогичный по объемам и выдаваемым характеристикам морфологический словарь «Диалинг» (http://www.aot.ru/) является полноценным лемматизатором и ни в коем случае не заявляется как стеммер.

Одним из современных вариантов реализации бессловарной морфологии в чистом виде является стеммер Портера (http://snowball.tartarus.org/). В нем пришедшая на вход строка проверяется на наличие заданных постфиксов, причем постфиксы проверяются в определенном порядке, а часть постфиксов может комбинироваться. Так, после выделения постфикса прилагательного может остаться постфикс причастия. Все, что осталось после их последовательного «откусывания», объявляется стемом. В зависимости от найденного постфикса слову может быть приписана та или иная часть речи, хотя в подавляющем большинстве задач этого не требуется. Алгоритм предельно прост, обладает очень высокой скоростью, однако дает большой процент ошибок. Так, например, если требуется подсчитать частотность слов, то для уже разбиравшихся слов «идти» и «шов» будут сгенерированы несколько никак не связанных стемов. В результате частотность данных слов будет существенно понижена за счет «размазывания» ее по нескольким группам. Кроме того, деление на постфиксы является в значительной мере спорным. Скажем, постфикс «-ев» относится к существительным, тогда как слово «ошалев» таковым не является. Также алгоритм выдает единственный вариант разбора,

полностью скрывая омонимию слов. Заметим также, что исходный алгоритм был несколько дополнен его отечественными пользователями, что несколько сократило процент ошибок.

Алгоритм Портера очень слабо учитывает тот факт, что для различных частей речи и даже для различных парадигм перед постфиксом могут стоять различные буквы. Этот факт используется в системе морфологического анализа Stemka (http://www.keva.ru/stemka/stemka.html), где хранятся не только сами постфиксы, но и еще две предшествующие буквы псевдоосновы. Сами комбинации букв и постфиксов хранятся в виде конечного автомата справа налево.

Существенным плюсом бессловарных морфологий является то, что они могут выдать результат для любых слов, встречающихся в тексте, что очень удобно при анализе текстов из незнакомой предметной области или содержащих много нелитературных или редко употребляемых слов. Однако корректность выдаваемой информации находится на уровне 90-95%. Это привело к отказу от бессловарных морфологий в задачах, когда точность анализа должна превалировать над его полнотой, и к переходу к словарным морфологиям в таких задачах, как машинный перевод и диалоговые системы. Однако на практике существует большое количество задач, решаемых статистическими методами, в которых вполне достаточно приблизительного знания о связях между словами. Это задачи рубрикации, информационного поиска, частично – задачи реферирования, ряд других задач.

Методы бессловарных морфологий активно используются в словарных морфологиях для предсказания нормальной формы и набора параметров слов, которые отсутствуют в морфологическом словаре. Для этого необходимо проанализировать постфиксы слова и попытаться образовать нормальную форму исходя из полученного префикса и парадигмы, приписываемой постфиксу. Для этого по найденным постфиксам определяются постфиксы нормальной формы, которые присоединяются к полученным префиксам, и наборы морфологических параметров. Существенным недостатком является большое количество предсказанных вариантов. Так, например, слово «кони» может быть предсказано как существительное мужского рода («ог*ни*» – «ко*ни*» → «ог*онь*» – «ко*онь*»), женского рода одушевленное или неодушевленное («кошк*и*» – «кон*и*» → «кошк*а*» – «кон*а*»), обладающее только множественным числом («сан*и*» – «кон*и*»), глаголы «конить» и «кнать» в повествовательном наклонении («г*они*» – «к*они*» → «г*нать*» – «к*нать*»; «юл*и*» – «кон*и*» → «юл*ить*» – «кон*ить*») и т.д.

Количество таких вариантов может быть существенно сокращено за счет фильтрации. Так, например, при наличии морфологического словаря достаточно большого объема можно утверждать, что в нем находятся все местоимения, предлоги, союзы и некоторые другие части речи. Отсев можно произвести и с точки зрения статистики. Достаточно большое количество парадигм содержит всего по нескольку слов. Также много парадигм, созданных для единственного слова. Например, парадигму составляют все формы слова «идти», так как в словарной морфологии оно будет обладать пустой основой. Эти парадигмы в большинстве своем являются закрытыми, т.е. добавление новых слов в них уже невозможно. В связи с этим можно отсечь подобные парадигмы, запретив выдвижение гипотез на их основе. Для этого достаточно подсчитать количество слов, относящихся к каждой из гипотез, и выдвигать гипотезы только на основе парадигм, к которым принадлежит количество слов, большее заданного порога.

Еще один вариант отсеивания основывается на том, что у слов, принадлежащих одной парадигме, совпадает не только изменяемая часть, но и последние несколько символов неизменяемой. Так, например, у слов «шествовать», «повествовать», «любопытствовать» псевдооснова заканчивается на «-ств-». Соответственно, новое слово, обладающее постфиксом «-ств-овать» (или в более общем случае «-ств+постфикс парадигмы») должно быть предсказано именно по этой парадигме. Для предсказания может быть использовано от одной до трех букв с конца псевдоосновы. Три буквы позволяют получить статистически значимые результаты для часто повторяющихся длинных последовательностей. Но так как одни и те же трехбуквенные последовательности встречаются гораздо реже, чем единственная буква, то использование трех букв дает гораздо больше вариантов таких последовательностей, используемых для проверки.

На практике подобный метод используется для некоторого разрешения неоднозначности выдаваемых результатов, когда кроме стема требуется определить и хотя бы часть речи. В такой ситуации три предыдущие буквы позволяют с достаточно высокой точностью определить, к какой части речи должно относится слово. Для этого помимо псевдоокончания хранят и несколько последних букв. Большее количество букв псевдоосновы позволяет более точно определить параметры слова. При этом меньшее количество букв позволяет достичь большей скорости разбора и сокращает объем словаря.

Методы бессловарных морфологий используются также и для расширения словаря. Так, например, очень многие существительные могут употребляться с «не-» в начале. Введение всех существительных с «не-» значительно увеличит объем словаря и снизит скорость его работы. В связи с этим при морфологическом анализе можно предварительно проверить наличие одного из префиксов («пере-», «при-» и т.д.) или аффиксов (например, «-ся» и «-сь»), а оставшуюся часть слова попытаться найти в словаре. При этом, например, для слова «оставшийся» после удаления аффикса основы найдено не будет, а слово «делавшийся» будет успешно разобрано (если предположить, что ни того, ни другого нет в словаре). Заметим, что запускать подобный алгоритм имеет смысл лишь после того, как выяснится, что пришедшее слово целиком отсутствует в словаре. Заметим также, что подобная операция может быть проведена средствами предсинтаксического анализа, в котором могут быть предусмотрены соответствующие средства.

§ 2.4. Коррекция орфографических ошибок

Небольшая модификация алгоритма морфологического анализа может помочь распознавать слова, написанные с ошибками. Для этого используется расстояние Левенштейна – минимальное количество ошибок, исправление которых приводит одно слово к другому. Считается, что существуют ошибки трех видов: ошибка вставки, ошибка замены и ошибка пропуска. Ошибка вставки – это случай, когда в слово вставлена лишняя буква; ошибка пропуска – когда буква была пропущена; и ошибка замены – когда одна буква заменена другой. При поиске слова с ошибкой нам придется поочередно пытаться пропускать буквы то во входном слове (в случае ошибки вставки), то в дереве (в случае ошибки пропуска), или и там, и там (в случае ошибки замены). При этом если пропуск буквы во входной строке является действием тривиальным, то пропуск вершины в дереве ведет к тому, что мы не знаем,

какого именно потомка данной вершины следует выбрать. В связи с этим приходится осуществлять перебор всех потомков и отбирать все успешные варианты.

Ошибка в ходе морфологического анализа может проявляться один из двух способов. При анализе слова мы можем дойти до места, когда переход по текущей букве будет отсутствовать. Ошибка может быть совершена в одной из предыдущих букв, поэтому разбор слова следует начать сначала. Еще одним вариантом определения ошибки в слове будет тот факт, что множество парадигм, привязанных к найденным префиксам, не будет пересекаться с множеством парадигм, найденных для постфиксов.

Кроме того, следует вводить ограничение на количество ошибок, которые мы позволяем допустить. Как говорит поговорка: «Если в слове хлеб допустить всего четыре ошибки, то получится слово пиво». Если мы фиксируем число ошибок, то для коротких слов оно может оказаться избыточным. Так, с одной ошибкой слово «быть» сводится к словам «быт», «мыть», «выть», «бить». С двумя ошибками мы уже получаем «плыть», «забыть», «ять», «зять» и множество других слов, уже гораздо менее похожих на оригинал. Аналогично в длинных словах нельзя применять процентное соотношение, так как, например, 25% букв от слова «великолепнейший» дает нам 3 ошибки, которые позволяют свести данное слово ко всем его словоформам. В связи с этим верхнюю границу числа ошибок обычно ограничивают как процентным соотношением, так и фиксированным числом. Например, не более 30% букв входного слова, но не более 3. При этом все равно стараются найти слова с минимальным количеством ошибок. Т.е., если для слова «великолепнейший» (предположим, что именно этой словоформы у нас нет в словаре) мы найдем слово «великолепнейшая» (2 ошибки), то слово «великолепнейшего» (3 ошибки) рассматриваться уже не будет.

Также могут вводиться специфические варианты ошибок. Так, при ручном наборе текста вариантом ошибки замены является смена порядка следования двух букв (транспозиция): «рпивет» вместо «привет». При распознавании текста могут рассматриваться специфичные замены: «ю» → «io» и т.д.

Компанией «Диктум» был предложен следующий вариант алгоритма коррекции ошибок при анализе текстов. В качестве основных берутся ошибки вставки, пропуска, замены и транспозиции. Кроме того рассматриваются ошибки следующего вида.

1. клавиатурная близость клавиш: *«анеудот – анекдот»*;
2. ошибки в безударных гласных: *«аностасия – анастасия»*;
3. фонетическая похожесть букв: *«брюпетка – брюнетка»*;
4. парные буквы: *«автограв – автограф»*;
5. вставка лишнего пробела: *«сло во – слово»*;
6. отсутствие пробела или дефиса: *«футбольныйклуб – футбольный клуб»*;
7. схожесть написания цифр и букв (ч-4, о-0, з-3): *«4естно – честно»*;
8. идентичное написание букв в разных раскладках: «XP0M0C0MA – хромосома»;
9. буквы и символы в разных раскладках: *«<лизнец – близнец»*;
10. ошибки после шипящих и ц: *«жолтый – желтый»*;
11. перепутывание и смещение рук при слепой печати: *«инвнжае – телефон»*;
12. перевод транслитерации на русское написание: *«kartinki – картинки»*;
13. исправление неправильной раскладки клавиатуры как для целого, так и для части слова: *«jlyjrkfccybrb – одноклассники»*.

Для указанных ошибок расстояние Левенштейна берется из интервала [0;1], так как их вероятность выше замен другими буквами. Для хранения весов расстояния Левенштейна используются квадратные матрицы, определенные для всего алфавита.

Формально задача ставится следующим образом. Пусть дан алфавит A, на данном алфавите определен словарь языка $L \subset A^+$. Тогда для слова $\mathbf{w} \in A^+$ требуется найти множество слов $\{\mathbf{w}'\}$, таких, что $dist(\mathbf{w}, \mathbf{w}') \leq \sigma$ и $F(\mathbf{w}') = \max(v)$, где $\mathbf{v} \in L$ и $dist(\mathbf{w}, \mathbf{v}) \leq \sigma$. При вычислении функции *dist* используются значения из матриц. Если занчение не найдено, вес ошибки принимается равным единице.

Однако для устранения части ошибок требуется учесть лексический контекст. Так, например, одной из широко распространенных ошибок является вставка мягкого знака в глаголы в третьем лице (в этом случае получается нормальная форма глагола): «нравиться» vs «нравится». Подобное слово содержится в словаре, но написано с ошибкой, которая не позволяет провести синтаксический анализ. Кроме того, коррекция ошибок в слове может дать несколько вариантов написания. Для устранения подобных ситуаций следует учитывать контекст слова. Например, «белый грип» – «белый гриб», но «птичий грип» – «птичий грипп», или «очень нравиться телефон» при правильном «очень нравится телефон».

Для исправления подобных ошибок могут использоваться синтаксический анализ с коррекцией или метод n-грамм. При ошибке, изменяющей форму слова, в их базе n-грамм не будет обнаружено соответствующего сочетания. При этом придется предложить варианты замены для всех слов в n-грамме. Так, например, фраза «очень нравиться телефон» может быть истолкована и как «очень нравиться телефону». Однако в полной фразе «Мне очень нравиться телефон» последний вариант представляется маловероятным и будет отсеян. Отсев будет проведен либо на этапе устранения омонимии, либо на синтаксическом анализе, так как фраза не содержит субъекта.

Глава 3. Постморфологический и предсинтаксический анализ

§ 3.1. Автоматизированное снятие омонимии

В некоторых задачах оказывается возможным автоматическое обучение по большому корпусу текстов. Т.е. имеется возможность собрать статистическую информацию, которая потом будет использоваться для вероятностного определения характеристик текстов. К подобным подходам относится метод n-грамм. Суть метода состоит в следующем. На большом размеченном корпусе текстов со снятой омонимией подсчитывается частота встречаемости для всех имеющихся в тексте комбинаций из n последовательно идущих слов. При этом обычно опускается нормальная форма слова, т.е. в расчет принимаются лишь часть речи и лексические параметры. В ходе разбора текста подобная информация может использоваться для вероятностного снятия омонимии. Мы можем предположить, что первые n-1 слово в n-грамме определяют нам вероятность появления n-го слова. Таким образом, зная первые n-1 слово мы можем выбрать наиболее вероятный вариант последнего. В ряде случаев используют линейную комбинацию вероятностей нескольких сокращающихся последовательностей слов.

Другим способом применения данного метода является отсечение наименее вероятных вариантов, т.е. частичное снятие омонимии. Так, если в корпусе текстов нам ни разу не встретилась определенная комбинация слов, то можно считать, что она вообще не должна встречаться в текстах. Таким образом, если мы встретили ее в реальном тексте, то часть входящих в нее омонимов может быть отброшена.

Значение n обычно принимается равным трем, так как биграммы обладают слишком малой историей и в дальнейшем не дают хороших результатов, а 4-граммы порождают слишком большое количество вариантов. При порождении n-грамм количество вариантов может быть уменьшено последующей их обработкой. Так, например, может оказаться, что для некоторой пары слов, стоящей в начале триграммы, существует полный набор вариантов для третьего слова, причем вероятность их появления примерно равна. Это будет означать, что первые два слова не определяют третье, и весь набор троек может быть удален как неинформативный. Подобный подход работает лишь в ситуации, когда мы выбираем наиболее вероятный вариант. При отсеивании наименее вероятных омонимов сам факт наличия подобных троек в базе будет означать, что у третьего слова не может быть удалено ни одного омонима.

Приведенный метод позволяет учитывать согласование параметров слов. Для каждой триграммы можно хранить лишь те значения лексических параметров, которые совпадают у отдельных комбинаций или всех слов в n-грамме. В методе следует различать знаки препинания, не сводя их, например, к одной части речи, так как роли, например, запятой и двоеточия в предложениях существенно различны.

Описанный метод позволяет получить точность снятия омонимии до 95% при выборе единственного наиболее вероятного варианта (при полном снятии омонимии) и порядка 99% (в зависимости от степени снятия) при отсеивании наименее вероятных вариантов (при частичном снятии омонимии).

Существует несколько методов, применяющих информацию об n-граммах. Наивный классификатор Байеса – это наиболее простой вид теггера (программы морфологической разметки, возможно, совмещенной со снятием омонимии), обучающегося на эталонном корпусе, который применяется для снятия омонимии с

помощью лексических параметров соседних слов, используя варьируемое окно контекста. Классификатор Байеса основывается на том предположении, что все параметры статистистически не зависимы между собой. В задаче снятия омонимии контекст, в котором появляется омонимичное слово представляется набором параметров (F_1;F_2; … ; F_n), а значение самого омонимичного слова представляется классом (S). Параметры F_i могут быть бинарными и представлять, появляется или нет какое-либо омонимичное слово с некоторым набором слов слева и справа от него. Для наивного Байесовского классификатора суммарная вероятность появления комбинации контекстных параметров с данным словом описывается следующим образом (более подробно теория по данному вопросу изложена в Часть V.§ 1.3):

$$P(F_1, F_2, ..., F_n, S) = P(S)\prod_{i=1}^{n} P(F_i \mid S) \qquad (3.1)$$

Любой из параметров, который равен нулю, говорит о том, что наше слово никогда не появляется с определенным значением. Забегая несколько вперёд, скажем, что такие значения сглаживаются путём присвоения им по умолчанию очень маленькой вероятности. В общем случае, каждый параметр F_i может входить с соответствующим весовым коэффициентом в выражение 3.1. Знаки препинания могут учитываться или нет, в зависимости от конкретной реализации системы автоматической обработки текста. В системах автоматической обработки теста капитализация слов, как правило, никогда не учитывается. Окно контекста может охватывать только левых, только правых или сразу левых и правых соседей слова. Выбор размера окна контекста оптимальным образом - это отдельная задача.

Как уже было отмечено, статистическое моделирование естественного языка предназначено для морфологической разметки текста с помощью закономерностей, которые не могут быть получены в явной аналитической форме. Здесь возникает проблема выбора наиболее подходящей статистической модели q(x), которая бы учитывала все особенности обучающей выборки. Таким образом, сама обучающая выборка является ограничениями, которые накладываются на q(x). Обратимся к такому понятию как энтропия, которое является основным для теории информации. Энтропия - это мера априорной неопределенности системы. Энтропия обладает следующими полезными свойствами: обращается в ноль, когда одно состояние системы достоверно, а другие невозможны; при заданном числе состояний обращается в максимум, когда эти состояния равновероятны[68]. Согласно принципу максимальной энтропии, вид модели q(x) подбирается таким образом, чтобы максимизировать предмет энтропии $H(q)$, не делая никаких дополнительных предположений для последовательности из N слов, не представленных в обучающей выборке. Принцип максимальной энтропии записывается в следующем виде:

$$H(q) = -\sum_x q(x)\log q(x) \qquad (3.2)$$

Средний показатель энтропии для английских текстов составляет 6-10 бит на слово, который может зависеть от вида N-граммной модели и жанра текста. В рамках задачи по разметке текста, энтропия – это среднее число бит, нужное, чтобы определить значение слова в данной обучающей выборке. Показатель связанности (perplexity) – это среднее геометрическое количество слов, которое может оказывать

[68] В случае N-граммной модели это означает, что вероятность появления N-граммы вычисляется по методу максимального правдоподобия.

влияние на неизвестное слово. Это еще одна стандартная мера для сравнения моделей языка, которая выражается следующей формулой:

$$PP = 2^{H(q)} \quad (3.3)$$

Следует подчеркнуть, что энтропия это некоторая функция, которая характеризует как саму модель естественного языка, так и имеющуюся обучающую выборку. Среди двух вероятностных моделей, имеющих одинаковый уровень ошибок предпочтительнее та, у которой энтропия меньше. Вообще же, количество ошибок и энтропия не однозначно связаны между собой.

В соответствии с принципом максимальной энтропии у нас есть возможность выбрать наиболее оптимальную базовую вероятностную модель естественного языка. Но эта базовая модель основана на принципе максимального правдоподобия, который не позволяет учитывать неравномерности в обучающей выборке и делать разметку при неполной информации.[69] Разумеется, что от системы по автоматической обработке текста требуется, чтобы она обрабатывала как можно более широкий круг текстов, а не только тот, что был представлен в обучающей выборке[70]. Таким образом, сглаживание используется, а зачастую просто необходимо, в том случае, когда для обучения доступен небольшой корпус, и есть возможность получить нулевые вероятности для последовательности слов, не представленных в обучающей выборке. Цель сглаживания сделать распределение более равномерным, другими словами, повысить вероятности для последовательностей слов, которые встречаются редко или вообще не встречаются и, соответственно, несколько снизить вероятности для комбинаций слов, которые часто встречаются. Как правило, методы сглаживания позволяют повысить качество работы триграммных тэггеров и, тем более, тэггеров на основе скрытых марковских моделей высоких порядков.

Различные методы сглаживания N-граммной вероятностной модели позволяют подобрать оптимальную[71] статистическую модель естественного языка. Проблему выбора оптимального тэггера попробуем обрисовать на следующем примере. Предположим, что у нас есть обучающее множество $X = \{x_1,x_2,\ldots,x_N\}$ и нам нужно получить распределение вероятностей $q(x)$. В самом простом случае, мы используем максимум правдоподобия и полагаем, что $q(x)$ совпадает с эмпирическим распределением $p`(x) = c(x) / N$, где $c(x)$ – число раз, которое встречалось слово x, а N – размерность обучающей выборки. Но в таком случае, мы придём к переобучению модели и не сможем разбирать N-граммы, не представленные в обучающей выборке. Другими словами, необходимо чтобы $q(x)$ соответствовала только наиболее значимым свойствам распределения $p`(x)$.

Для наглядности, приведем примеры. Предположим, что $x = (w_1; w_2)$, где w_1 и w_2 английские слова, которые появляются в некотором большом корпусе английских текстов. Таким образом, задача сводится к оценке частоты появления биграмм, в данном случае, в английском языке. Предположим биграмму, которая не появляется в обучающем множестве – «**PIG DOG**». Имеем, p(**PIG DOG**) = 0 , но интуитивно мы хотим, чтобы p(**PIG DOG**) > 0, т.к. эта биграмма имеет некий шанс появиться. Еще

[69] Такая модель плохо размечает или вообще не размечает последовать из N слов, не представленных в обучающей выборке.

[70] Теоретически, при наличии представительной обучающей выборки, по предельной теореме перейдем от частот появления N-грамм к вероятности их появления, и с помощью принципа максимального правдоподобия получим наиболее оптимальную вероятностную модель автоматической обработки текста. Имеющимися современными средствами на практике такое не достижимо.

[71] Строго говоря, на практике обычно подбирается субоптимальная модель естественного языка.

один пример, предположим, что слово «**Mateo**» может появляться только после слова «**San**» в обучающем корпусе (биграмма «**San Mateo**»). Таким образом, имеем, что p (w_1 Mateo) = 0 для всех $w_1 \neq$ San, но интуитивно, мы хотим, чтобы $p(x) > 0$ для всех w_1, а не только для w_1 = San. Мы хотим, чтобы наша модель максимально хорошо разбирала случаи, представленные в обучающей выборке и, также максимально хорошо разбирала неизвестные случаи. Другими словами, требуется, чтобы система принимала решения при неполной информации.

Прежде чем привести вид сглаженной Марковской модели напомним, что такое оценка максимального правдоподобия для биграммной модели:

$$P_{ML}(w_i \mid w_{i-1}) = \frac{P(w_i \mid w_{i-1})}{P(w_i)} = \frac{c(w_{i-1} \mid w_i)/Nw}{c(w_{i-1})/Nw} = \frac{c(w_{i-1} \mid w_i)}{c(w_{i-1})} \quad (3.4)$$

, где Nw – число слов в обучающей выборке, $c(w_{i-1})$ и $c(w_{i-1} \mid w_i)$ – число раз, которое встречается строка w_{i-1} и $w_{i-1} \mid w_i$ в обучающем корпусе. Нулевая вероятность биграммы может привести к ошибкам при распознавании речи. При снятии неоднозначности с помощью N-граммных моделей высокого порядка можно получить относительно высокий процент ошибок при низком покрытии текста. Таким образом, чтобы получать более точные оценки вероятностей применяются различные виды сглаживания моделей. Пример самого простого сглаживания это прибавить единицу к частоте появления биграммы:

$$P_{ML+1}(w_i \mid w_{i-1}) = \frac{c(w_{i-1} \mid w_i)+1}{c(w_{i-1})+V} \quad (3.5)$$

, где V – размер словаря.

Сглаживание биграммной модели с помощью униграммной выглядит следующим образом:

$$P_{\text{int}\,erp}(w_i \mid w_{i-1}) = \lambda\ P_{ML}(w_i \mid w_{i-1}) + (1-\lambda\ )P_{ML}(w_i) \quad (3.6)$$

, где λ – положительной весовой коэффициент.

В общем виде выражение для сглаженной Марковской модели N-го порядка можно записать в следующей форме:

$$P_{\text{int}\,erp}(w_i \mid w_{i-n+1}^{i-1}) = \lambda_{w_{i-n+1}^{i-1}} P_{ML}(w_i \mid w_{i-n+1}^{i-1}) + (1-\lambda_{w_{i-n+1}^{i-1}})P_{\text{int}\,erp}(w_i \mid w_{i-n+2}^{i-1}) \quad (3.7)$$

, где P_{ML} – оценка максимального правдоподобия для модели предыдущего порядка (порядка N-1), λ – положительные весовые коэффициенты. Таким образом, сглаженная модель N-го порядка определяется рекурсивно как линейная интерполяция между моделью максимального правдоподобия и сглаженной моделью порядка N-1. Чтобы закончить рекурсию, можно взять в качестве сглаженной модели первого порядка оценку максимального правдоподобия (выражение 3.4), простое сглаживание (выражение 3.5) или предположить равномерное распределение вероятности появления каждого слова:

$$P_{unif}(w_i) = \frac{1}{V} \quad (3.8)$$

Для каждой последовательности слов w_{i-n+1}^{i-1} есть свой набор весовых коэффициентов $\lambda_{w_{i-n+1}^{i-1}}$, вычисляемых оптимальным образом. Последовательности слов,

которые наблюдались много раз в обучающем корпусе и те, которые встречались лишь несколько раз будут иметь разные коэффициенты λ. Чтобы не хранить набор параметров для каждой последовательности слов и не затягивать процесс обучения, N-граммы объединяются в относительно небольшое число классов[72], а коэффициенты λ вычисляются отдельно уже для этих классов. В выражении (3.8) интерполяция может быть нелинейной и тогда общий вид выражения несколько изменится. Есть методы сглаживания вероятностей, которые больше подходят для большой обучающей выборки, а есть которые подходят для относительно маленькой выборки. Для биграммной модели, обученной на большом корпусе метод сглаживания Church-Gale предпочтительнее, в то время как, сглаживание по методу Katz лучше применять для биграмм, полученных с небольшого обучающего корпуса.

Для широкопопулярной триграммной модели, впервые описание эксперимента по применению взвешенной суммы вероятностей моделей первого и второго порядка применил Frederick Jelinek в 1980 году. Сглаженная триграммная модель содержит линейные комбинации триграммных, биграммных и униграммных байесовских вероятностей:

$$Psmooth(w_i \mid w_{i-2} * w_{i-1}) = \lambda_3 * P(w_i \mid w_{i-2} * w_{i-1}) + \lambda_2 * P(w_i \mid w_{i-1}) + \lambda_1 * P(w_i) \qquad (3.9)$$

, где сумма коэффициентов $\lambda_1 + \lambda_2 + \lambda_3 = 1$, причем $\lambda_1 > 0$, $\lambda_2 > 0$, $\lambda_3 > 0$. Значения для λ_1, λ_2, λ_3, получены решением системы линейных уравнений. В публикации Чешских исследователей за 1998 год была представлена точность около 93% для обычного HMM тэггера, а с использованием сглаженной триграммной модели точность разметки возросла до более чем 95%.

Еще одним вариантом устранения омонимии является выделение словосочетаний. Входные правила для данного этапа также могут быть порождены автоматически. Для каждого слова на основе большого корпуса текстов (не обязательно даже размеченного) можно посчитать частоту его встречаемости. Далее мы можем посчитать частоту встречаемости, например, пар слов. Для тех пар, вероятность встретить которые выше, чем их совместная вероятность, можно высказать предположение, что они являются словосочетаниями. На практике, если полученная по анализу корпуса частота встречаемости превосходит в два раза теоретическую, то данная пара гарантированно является словосочетанием.

Для словосочетаний большей длины не обязательно учитывать соответствующие комбинации. Дело в том, что пара слов может оказаться началом многословного словосочетания. Таким образом, обнаружив все двухсловные комбинации мы также выделили и кандидатов на многословные. Если сохранить их позиции в тексте, то имеется возможность проверить следующие несколько слов на предмет совпадения. За счет этого снимается необходимость анализировать все возможные комбинации слов, количество которых будет существенно расти с ростом длины комбинаций.

Применение неразмеченного корпуса требует дальнейшего проведения работ по приписыванию отдельным словам их лексических характеристик. Во многом разметка будет определяться стоящими перед системой задачами. В некоторых случаях будет достаточно выделить сами словосочетания, которые потом будут использоваться как единые лексические единицы, например, для определения тематики текста. Для задач синтаксического анализа можно приписать каждому словосочетанию готовое дерево зависимостей. При машинном переводе необходимо

[72] В английском языке это называется bucketing (от слова bucket – корзина).

сопоставить ему эквивалент в другом языке. И если первая задача может быть решена автоматически на размеченном корпусе со снятой омонимией, то вторая и третья обычно решаются вручную.

Приведенные методы позволяют сравнительно быстро получить высокие результаты при условии, что мы имеем доступ к размеченному корпусу текстов со снятой омонимией. В связи с этим создание подобных корпусов является важной научной и практической задачей. Наиболее полным размеченным корпусом в русском языке является Национальный корпус русского языка (www.ruscorpora.ru). Он содержит в себе тексты различной направленности, для которых проведен морфологический анализ, т.е. каждому слову приписаны его лексические характеристики. Для части корпуса снята омонимия, т.е. проделана большая работа по выбору корректных наборов лексических параметров. Вообще, создание представительного корпуса является важной национальной задачей, подхлестывающей развитие исследований в области данного языка, его использование. С практической точки зрения корпусом могут пользоваться, например, переводчики. При наличии сомнений о том, каким образом следует перевести ту или иную фразу, можно задать запрос поисковой системе корпуса и сравнить частоту употреблений имеющихся вариантов. При наличии разметки в корпусе можно проверить корректность фразы с точки зрения синтаксиса. С научной точки зрения корпус дает богатый материал для изучения структуры языка, используемых в нем слов и фраз, построения предложений, истории развития. Для этого корпус должен включать в себя тексты, написанные в разное время в различных жанрах и относящихся к разным предметным областям. Обычно в корпуса включают не только письменные источники, но и живую речь.

Но вернемся к разбору словосочетаний. Многие сочетания слов могут иметь несколько вариантов употребления, в которых они будут или не будут являться словосочетаниями. Так, например, сочетание «так сказать» в различных контекстах трактуется по-разному. В фразе вида «это, так сказать, явление» оно будет являться словосочетанием, определяющим наше отношение или вариативность произношения, но в предложении «Так сказать нельзя!» оно указывает на невозможность произнесения определенного текста. В связи с этим метод требует обязательной ручной доработки результатов, полученных автоматически.

Заметим, что в разобранном случае словосочетание будет трех- или даже четырехсловным, включающим в себя еще одну или две запятые. Та же самая фраза, но с внесенной в нее запятой («Так сказать, нельзя!») будет показывать, что мы пытаемся сформулировать запрет в другом виде («Всё это сделать весьма и весьма затруднительно. Так сказать, нельзя!»). Таким образом, после выделения пар слов, претендующих на роль словосочетаний, необходимо анализировать контекст не только справа, но и слева от них. Кроме того, расширение контекста может привести к получению более однозначных словосочетаний за счет возможной потери более коротких и чаще встречающихся.

§ 3.2. Постморфологический анализ

Предсинтаксический анализ необходим для выделения элементов текста, морфологического анализа этих элементов, разделения сложносоставных элементов на части, объединение простых связанных по смыслу элементов в группы, выделение фрагментов текста, которые могут разбираться самостоятельно. Его название

показывает место предсинтаксического анализа в общей системе: перед синтаксическим анализом. Задачей предсинтаксического анализа является подготовка данных для синтаксического анализа в наиболее удобной форме, максимально облегчающей выполнение задачи последнему.

На вход системы поступает текст. В первую очередь необходимо определить единицы этого текста: абзацы, предложения, отдельные слова и знаки препинания. В отличие от систем машинного перевода, диалоговым системам нет необходимости выделять заголовки, сноски, комментарии, врезки и прочие элементы текста, необходимые для сохранения форматирования. Подобное форматирование текста может понадобиться диалоговой системе только для приобретения новых знаний из существующих текстов, однако создание систем, способных на подобные экзерсизы – дело будущего. Выделение всех описанных элементов текста (как слов, так и врезок) является задачей графематического анализа.

Выделение абзацев в современных редакторах является тривиальной задачей. В них уже существует разметка на абзацы. При полностью текстовом вводе абзацы зачастую отмечаются символом перевода строки. В начале абзаца часто ставят два и более пробела или пробельную строку. В случае, когда каждая строка текста оканчивается символами конца строки, задача выделения абзаца может потребовать специальных знаний о структуре данного текста.

Задача выделения предложений менее тривиальна. Обычно предложение заканчивается точкой, вопросительным или восклицательным знаком, иногда – многоточием. Однако на практике те же знаки препинания используются и для других целей. Точка часто применяется в сокращениях. При этом если сокращение приходится на конец предложения, то ставится только одна точка, относящаяся как к сокращению, так и к концу предложения. Восклицательный и вопросительный знаки часто используются в выразительных вставках в тексте: «… и он, Великая удача!, …», «… он смотрел, А что он мог сделать?,...». Таким образом, знаки препинания не являются стопроцентной гарантией окончания предложения. К счастью, при общении с диалоговыми системами выразительные вставки обычно не используются, однако проблема точки остается. Еще одна проблема на данном этапе – это слова, написанные с большой буквы, после точки. Так в предложении «Мишка очень любит мёд…» без использования прагматики и контекста не совсем понятно о ком идет речь: о ласково называемом медведе или личности по имени Михаил, также называемом уменьшительно-ласкательным именем. В первом случае однозначно рассматривается начало предложения, во втором есть шанс, что предшествующая точка могла стоять после сокращения.

Еще одну проблему представляют собой цитаты и прямая речь, также нечасто используемые при общении с диалоговыми системами. В состав цитаты может входить как несколько слов, так и несколько предложений. Прямая речь обычно содержит некоторый связанный фрагмент текста. В связи с этим и разбирать их лучше как отдельный текст. Однако, например, в английских текстах, прямую речь принято выделять не кавычками, а апострофами. Те же самые апострофы используются для обозначения притяжательного падежа и сокращений: «man’s», «mans’», «it’s», «I’ll», «’cause». В текстах, пытающихся подчеркнуть простонародность речи, сокращения (и как результат апострофы) встречаются очень часто. В связи с этим проблема выделения начала и конца прямой речи стоит остро.

Следует заметить, что притяжательный падеж в английском языке может образовываться не только от существительных, но даже, например, от глаголов: «last gone's daughter». В этом случае не совсем понятно, что делать с притяжательностью: формально глаголы не обладают параметром «притяжательность». Более того, приведенный пример может сам по себе рассматриваться как отдельное предложение: «last gone is (was) daughter». Заметим, что знаки препинания могут встречаться в слове несколько раз: «a-number-one's».

Отдельной проблемой являются тире и дефис. В двухбайтных кодировках они различаются, и системы редактирования текстов имеют возможность ставить их в нужных местах. Однако однобайтные кодировки знают только один символ – минус. В связи с этим в однобайтной кодировке (или в случае ошибок пользователя) отличить «кто-то» от «кто то» в следующих двух предложениях можно только в результате синтаксического анализа: «он знает, что кто-то пришел к другу …» от «он знает кто – то пришел к другу …». Ориентироваться на наличие пробелов в такой ситуации можно, но и в этом случае мы не имеем стопроцентной гарантии.

Для решения этих и других проблем, возникающих при членении текста на составляющие, используется графематический анализ. Для работы графематического анализа нам среди прочего потребуется этап деления сложносоставных слов. Данный этап занимается тем, что делит сложное слово, составленное из нескольких, на составляющие его, например, «сине-зелено-красный». Данная проблема особенно актуальна в немецком и турецком языках. В немецком языке разрешено формировать произвольные сложносоставные слова, если образующие его слова относятся к одному понятию. Часть из них, например «Siecherheitdienst» (Siecherheit + dienst), уже устоялись и считаются одним словом, но большинство подобных слов образуется «на лету» и заносить их в морфологический словарь нет никакой возможности.

Однако примеры из немецкого языка меркнут перед примерами из более редких языков. **Mamihlapinatapai** - слово из яганского языка Племя Яган, (Огненная Земля), указано в книге рекордов Гиннесса в качестве «наиболее сжатого слова» и считается одним из самых трудных для перевода слов. Оно означает «Взгляд между двумя людьми, в котором выражается желание каждого в том, что другой станет инициатором того, чего хотят оба, но ни один не хочет быть первым». Слово состоит из рефлексивного / пассивного префикса ма-(МАМ-перед гласным), корень ihlapi, что значит быть в недоумении, как то, что делать дальше, то stative суффикса-н, достижение Суффикс-ate, и двойной суффикс-apai, который в составе с рефлексивным mam- есть взаимные чувства.

Пример восьмипорядковой деривации в эскимосском языке: *igdlo-ssua-tsia-lior-fi-gssa-liar-qu-gamiuk* (дом-большой-довольно-изготавливать-место-быть-идти-велеть-когда.он.его), «Велев ему пойти туда, где строился довольно большой дом».

К великому счастью, подобные языки обычно не анализируются автоматизированными методами.

В отличие от европейских языков, турецкий язык (и не только он) является агглютинативным. В нем часть информации об употреблении слова добавляется в конец слова в виде аффиксов. Так, например, в одно слово можно сказать kitap_lar_ım_da_ki_ler_i: «те (вин. падеж), что лежат на моих книгах». За счет добавления аффиксов у одного существительного может появиться несколько тысяч словоформ, хранить которые в морфологическом словаре также будет просто невозможно, так как аффиксы будут добавляться после любой имеющейся

словоформы, не содержащей аффиксов, хотя и в строго определенном порядке. В связи с этим было бы проще ввести ряд дополнительных параметров и «откусив» соответствующий аффикс, добавить оставшейся словоформе параметр с заданным для данного аффикса значением.

Остальные сложные слова должны разделяться по определенным правилам. В противном случае только слово «поляк» будет иметь более 20 вариантов деления. Так, все прилагательные, кроме последнего, в примере, приведенном для русского языка («сине-зелено-красный»), должны находиться в краткой форме (и как минимум присутствовать в словаре).

Таким образом, этап деления сложносоставных слов имеет довольно сложную структуру и управляется языкозависимыми правилами. Работа графематического анализа будет выглядеть следующим образом. Сперва, графематический анализ по заданным критериям выделяет абзацы. Далее выделяется строка до первого разделителя (пробела, перевода строки, иного знака препинания). Если строка состоит из одних цифр, то она помечается частью речи «числительное» и отправляется в промежуточный массив. В противном случае строка подается на этап деления сложносоставных слов. Если после выделенной строки стоит единственный знак препинания, разрешенный для присутствия в словах, представленных в морфологическом словаре, то мы выделяем следующую строку, объединяем с предыдущей, вновь проверяем на наличие разрешенного разделителя до тех пор, пока такой разделитель присутствует. После этого подаем полученную строку на этап деления сложносоставных слов. Если слово не представлено в морфологическом словаре и не может быть разделено, этап должен вернуть ошибку. При этом в массив выделенных слов будет помещен специальным образом помеченный кортеж, содержащий выделенную строку, с неизвестным словом. В противном случае мы возвращаем выделенный набор слов и помещаем его в массив выделенных слов. Выделенные знаки препинания также подаются в массив выделенных слов.

Для борьбы с неоднозначной расстановкой точек можно вводить правила анализа сокращений. Можно выделить несколько видов сокращений. Фиксированные сокращения не изменяют своей формы записи. Это, к примеру, «и т.д.», «и т.п.», «т.о.». Найдя подобные обороты, мы можем заменить их отдельными словами. При этом приведенные примеры лучше заменить одним сложным словом, так как они являются неразрывными неизменяемыми словосочетаниями и обрабатываются как одна лексическая единица. Заменив их на несколько слов, мы можем получить неоднозначность анализа структуры предложения.

Изменяемые сокращения представляют собой сокращение и некоторую обязательную, но произвольно изменяющуюся часть. Это, например, сокращения в датах, именах, названии населенных пунктов и географических мест: «о. Врангеля», «г. Москва», «1904 г.», «г-н Герострат», «А.С. Пушкин». Часть из сокращений имеет смысл развернуть в полное слово. При этом может возникнуть проблема с приписыванием параметров развертываемому слову. «1904 г.» может обозначать «год», «году», «года» и т.д.

Такие сокращения, как «и т.д.», «1904 г.» и прочие, могут встречаться в конце предложения. При этом точка будет обозначать как конец предложения, так и точку в сокращении (для примера см. последнее предложение предыдущего абзаца). В связи с этим следует ввести дополнительный параметр в правила – может ли данное сокращение заканчивать предложение.

Устранив все «лишние» точки и расставив маркеры о возможном окончании предложений, мы можем считать, что все точки, восклицательные и вопросительные знаки или маркеры, после которых идет слово с большой буквы, заканчивают предложение.

По окончании выделения отдельных слов имеет смысл провести свертку части таких слов. Это необходимо сделать, чтобы облегчить работу синтаксическому анализу.

Во-первых, можно свернуть числительные, написанные в виде слов, превратив их в числовое написание. Во многих языках мира словесное написание числительных выглядит следующим образом. Изначально присваиваем текущей и итоговой сумме нулевые значения. При этом итоговая сумма будет хранить окончательный результат, а текущая сумма – промежуточное значение при подсчете количества сотен, тысяч, миллионов и т.д. Если следующее за текущим числительным слово обозначает числительное меньшее, чем текущее, то его следует прибавить к текущей сумме. Если следующее слово обозначает числительное большее, чем текущее, то текущую сумму необходимо умножить на это большее числительное и прибавить к итоговой сумме. Текущая сумма при этом обнуляется. По окончании числа текущая сумма прибавляется к итоговой.

Сто	пятнадцать	тысяч	двести	один	
100	+15	*1000	200	+1	=115000+201=115201

Для дробных числительных следует различать рациональные и десятичные дроби. Для дробных числительных основная проблема состоит в том, что они состоят из двух чисел, первое из которых выражается счетным числительным, а второе — порядковым. При этом в общем случае довольно сложно определить их границу. Основным критерием может служить тот факт, что первое число может быть меньше второго (семь восьмых, пятьсот шесть девятьсот одиннадцатых). Однако когда числитель больше знаменателя, задача может не иметь однозначного решения (один миллион двести восемнадцать тысяч трехсотых может быть эквивалентно 1000000/218300 и 1218000/300). К счастью, большинство подобных дробей легко сокращаются и приводятся к более очевидному варианту.

Для десятичных дробей на этапе анализа числительных необходимо ввести разделитель, показывающий место, с которого начинается дробная часть (целых в русском языке, point в английском). Сама дробная часть в различных языках выражается по-разному. Так, в русском языке после разделителя будет идти рациональная дробь, вторая часть которой выражается единственным словом (десятых, тысячных, стомиллионных. ...). Также возможна такая же запись дроби, как и в английском языке: после разделителя перечисляются цифры в том порядке, в котором они идут после запятой (точки). Но такая форма записи редко встречается на письме, особенно в человеко-машинном диалоге.

Для немецкого языка указанный алгоритм работать не будет в связи с тем, что порядок слов в немецких числительных несколько иной. Так, единицы в нем ставятся перед десятками, например, «fünf und zwanzig» - «пять и двадцать». Аналогично и в английском языке могут встречаться конструкции, включающие в себя слово «and»: «twenty and five». В связи с этим в алгоритм необходимо вводить слова, обозначающие сложение и даже умножение текущей и последующей сумм. Следует

помнить, что подобные слова сами могут вносить некоторую неоднозначность. Так, например, во фразе «Twenty and five vehicle» в качестве ответа на вопрос «How many peoples goes?» будет иметься в виду двадцать человек и пять машин.

Кроме того, слово «один» (и аналогичные ему в других языках) имеет собственную семантику. С одной стороны, оно может употребляться для обозначения некоторого объекта: «Дай одну», «В одном зоопарке, не помню каком,…». С другой стороны, оно может опускаться в числительных: обычно говориться «тысяча пять», а не «одна тысяча пять», хотя второй вариант также употребим, особенно в формальных документах.

Аналогично слова «десяток», «сотня» и «тысяча» могут выступать в роли существительных: «Во главе пяти сотен воинов с осадными орудиями он подошел к стенам города и начал правильную осаду». В данном примере слово «сотня» означает воинское подразделение, а не количество людей.

При обработке числительных следует помнить, что в тексте может встречаться запись групп числительных: телефоны, IP-адреса и т.п., записанные не числами, а словами. Т.е. по ходу выполнения алгоритма необходимо отслеживать, что следующее число не принадлежит одной группе (десятки, сотни…) с предыдущим и текущая сумма не больше итоговой. Например: «Сто девяносто два сто шестьдесят восемь ноль один». Однако подобная запись может обозначать как IP-адрес 192.168.0.1, так и число 19216801, хотя наиболее вероятным с точки зрения прагматики в такой ситуации представляется первый вариант. Второй вариант более вероятен при группировке цифр тройками для выделения тысяч, миллионов и т.д., например, 108.891.452. В этом примере не может быть записан IP-адрес, так как он должен содержать в себе четыре группы цифр от 0 до 255. Приведенная запись чисел характерна для финансовых расчетов и может предполагать до или после себя обозначения денежной единицы ($108.891.452 или 108.891.452 руб.).

Семантически нагруженные группы цифр: даты, номера телефонов, IP-адреса и т.д. также имеет смысл объединять в одно слово. Их можно объединять по шаблонам. Например, xx/xx/xxxx или xx.xx.xxxx для дат, xxx-xxxx, +x(xxx)xxx-xxxx, xxx-xx-xx для телефонов и т.д. Это также уменьшит количество слов, поступающих на синтаксический анализ и, как следствие, упростит его работу.

Превращая слова в цифры, следует иметь в виду, что слова могут подчеркивать роль цифр в слове. Они могут играть роль порядковых или счетных числительных. В связи с этим следует приписывать различную часть речи в зависимости от того, какое слово идет последним в группе. Так, «сто один» будет счетным числительным, а «сто первый» - порядковым.

Еще одной задачей предсинтаксического анализа является обработка словосочетаний. Можно выделить следующие виды словосочетаний: неразрывные неизменяемые, неразрывные изменяемые и разрывные. Неразрывные неизменяемые словосочетания состоят из одних и тех же словоформ, идущих одна за другой. Например, «таким образом», «так сказать», «не взирая на» и т.д. В состав неразрывных неизменяемых словосочетаний могут входить и знаки препинания: ср. «для того чтобы» в начале предложения и «для того, чтобы» в середине. Для поиска подобных словосочетаний необходимо просто проанализировать входной текст. Найденные словосочетания имеет смысл обрабатывать дальше как единую словоформу.

Кроме того, словосочетания можно разделить на открытые и закрытые. В закрытых словосочетаниях отдельные слова теряют собственный смысл и могут трактоваться только в составе словосочетания. При этом слова в открытых словосочетаниях сохраняют все лексические связи и, как следствие, могут подчинять себе другие слова. При этом может происходить разрыв словосочетания.

Следует опасаться неоднозначности у словосочетаний. Так, например, сочетание «так сказать» может встретиться в фразе вида «так сказать нельзя», хотя как вводная конструкция оно должно быть выделено запятыми.

Неразрывные изменяемые словосочетания состоят из идущих подряд словоформ, образованных от фиксированных нормальных форм, но в зависимости от контекста обладающих различными параметрами. Например, «бить баклуши» («бил баклуши», …), «наивная модель мира», «искусственный интеллект» и т.д. Обычно слова в неразрывном изменяемом словосочетании согласуются. При желании можно выделить главное и зависимые слова, однако свертка подобных словосочетаний позволяет, как и в предыдущем случае, сократить количество синтаксических единиц и тем самым упростить задачу синтаксического анализа. Заметим, что грань между изменяемым словосочетанием и простой синтаксической конструкцией довольно тонка. Так, если «Черный квадрат» можно отнести к словосочетаниям, то «красный квадрат» таковым являться не будет. Дело в том, что словосочетания имеет смысл выделять лишь в тех случаях, когда сдвигается семантика или прагматика предметов, указанных в словосочетании. «Черный квадрат» обозначает название картины и является именем собственным (кстати, в такой роли он должен быть обрамлен кавычками), а красный квадрат является просто квадратом, одно из свойств которого – цвет – обладает значением «красный». Аналогично, если искусственный интеллект является специализированной отраслью знаний, то, например, обширный интеллект – это интеллект, обладающий свойством обширности. Как всегда и здесь не обходится без неоднозначности. Так, в фразе «несколько искусственный интеллект» будет иметься в виду интеллект, обладающий свойством заполненности искусственно придуманными знаниями. Приведенный пример и сам является искусственным, однако на практике подобная ситуация встречается довольно широко.

Для анализа неразрывных неизменяемых словосочетаний необходимо предварительно провести морфологический анализ. Далее мы ищем требуемые словоформы в заданном порядке. При этом может быть необходимо проверить согласование слов по параметрам: заданные параметры должны обладать одними и теми же значениями. Как уже упоминалось, прилагательное и существительное в русском языке согласуются по ряду параметров. Следовательно, для словосочетаний, составленных из существительного и подчиненных ему прилагательных, необходимо проверить подобное согласование.

Разрывные словосочетания – это связанные слова, между которыми могут вклиниваться другие слова. Как и в предыдущем случае, связка слов дает несколько иное значение, чем просто сумма значений слов. В случае с разрывными словосочетаниям связь между словами либо очевидна, либо используемый вид согласования не требует того, чтобы слова стояли рядом. Более того, кроме подчиненного слова, образующего словосочетание, к главному слову могут присоединяться и другие зависимые слова, являющиеся его неоднородными членами. В итоге все подчиненные члены имеют право идти вперемешку. Например, «отправка самолетов», «прибытие самолетов» → «отправка и прибытие самолетов».

Так как слова в разрывных словосочетаниях могут быть разнесены по предложению, то сперва требуется провести синтаксический анализ предложения. Следовательно, работа с разрывными словосочетаниями не может быть отнесена к предсинтаксическому анализу.

§ 3.3. Синтаксическая сегментация

Еще до проведения синтаксического анализа есть возможность выдвинуть некоторые предположения о структуре разбираемого предложения, выделить его фрагменты (сегменты), которые можно разобрать независимым образом. Дальнейший синтаксический анализ будет опираться на эти предположения и получит возможность сразу отбросить часть вариантов. Для использования этих возможностей вводится этап синтаксической сегментации.

Первой задачей *синтаксической сегментации* является уменьшение количества омонимов, соответствующих каждой словоформе. Так, например, если слово может являться как наречием, так и прилагательным, то следует проанализировать следующее за ним слово. Если оно является однозначным глаголом, то слово будет наречием. Если следующее слово является однозначным прилагательным или существительным, то данное слово будет прилагательным. Подобных правил достаточно много, но очень часто они могут служить лишь предположениями, так как существует альтернативный вариант прочтения данного фрагмента. Однако зачастую даже выделение нескольких альтернатив помогает сократить количество возможных вариантов.

Вторая задача – выделение синтаксических конструкций. Например, мы можем выделить начало сложноподчиненного предложения, обнаружив ключевые слова «, который», «, потому что», «, когда» и т.д. Можно выделить деепричастные обороты, найдя место, где за существительным после запятой идет деепричастие. Кроме того, можно попытаться определить связность и подчинение фрагментов. Так, например, во фразе «...**когда на столе**, покрытом скатертью**, они расставили тарелки**...» жирным выделен единый фрагмент, одному из слов которого (стол) подчинен вставленный в него в фрагмент.

Для поиска фрагментов нам потребуется понятие шаблона поиска. Под шаблоном поиска слова будем понимать кортеж <N,S,P>, где N – нормальная форма искомого слова, S – часть речи и P={p} множество искомых параметров искомого слова. Нормальная форма слова может представлять собой строку с искомой нормальной формой, либо пустую строку, если нормальная форма нас не интересует. Аналогично описывается и часть речи. Множество параметров может быть пустым, если параметры при поиске нас не интересуют. Искомые параметры, в отличие от обычных параметров, будут содержать дополнительный флажок, указывающий на тип поиска. Предполагаются следующие типы поиска:

- точный, когда сравнивается как имя, так и значение параметра;
- по имени, когда проверяется наличие параметра у данного слова вне зависимости от его значения;
- совпадающий, когда значение параметра должно совпадать со значениями таких же параметров у других шаблонов;
- несовпадающий, когда значение параметра может принимать любое значение, кроме указанного.

Под шаблоном поиска фрагмента будем понимать упорядоченное множество шаблонов поиска слов.

Приведем пример шаблона поиска фрагмента.

<‘’,’прилагательное’, {[‘род’,совп.,’’],[‘число’,совп.,’’],[‘падеж’,точн.,’им’]}>
<‘’,’существительное’, {[‘род’,совп.,’’],[‘число’,совп.,’’],[‘падеж’,совп.,’’]}>

Здесь мы пытаемся найти прилагательное и следующее за ним существительное вне зависимости от их нормальной формы, при этом у найденных слов должны совпадать род, число и падеж, причем падеж должен быть именительным.

Как уже упоминалось, нам может потребоваться найти слово, имеющее единственное значение. В связи с этим в шаблоне поиска слова необходимо добавить флаг, который будет показывать, должно ли значение быть единственным.

Следует заметить, что распространенной является ситуация, когда к одному и тому же месту в предложении подходят сразу несколько имеющихся шаблонов поиска фрагментов. При этом шаблоны могут пересекаться или один может включать в себя другой. Подобную ситуацию следует учитывать при работе с шаблонами.

Шаблон поиска фрагмента только ищет необходимый нам фрагмент. Второй частью задачи является преобразование найденного фрагмента. Для этого потребуется шаблон формирования фрагмента. Данный шаблон будет показывать, какие найденные слова требуется включить в выходной фрагмент и какие новые слова в него вставить. Так, например, нам может потребоваться найти несколько слов и слить их в одну лексическую единицу. В этом случае шаблон формирования фрагмента будет включать единственный элемент, полностью формирующий новое слово.

Элементы шаблона формирования фрагмента – шаблоны формирования слов – будут показывать, откуда следует взять нормальную форму слова и его часть речи, параметры слова (ввести новое, взять из слова входного предложения, из какого именно слова). Кроме того, нам могут потребоваться метки для синтаксического анализа. Формат меток будет существенно зависеть от методики проведения синтаксического анализа. Не нарушая общности рассуждений, возьмем метки начала и конца правил синтаксического анализа, разбирающих некоторые синтаксические конструкции.

Правило синтаксической сегментации будет состоять из шаблона поиска фрагмента, шаблона формирования фрагмента и списка исключений. Исключения показывают, какие правила необходимо исключить из исполнения в случае, если их шаблоны сравнятся с уже найденным фрагментом. Такая ситуация возможна, например, когда у нас есть два шаблона, один из которых описывает более частную ситуацию.

Синтаксическая сегментация будет работать по следующему алгоритму. Во входном предложении ищутся места, попадающие под хранимые в правилах шаблоны поиска фрагментов. Далее анализируется список исключений, и при нахождении пересекающихся исключаемых правил мы выбрасываем их из рассмотрения. Если после этого остались пересекающиеся фрагменты, необходимо создать несколько копий входных данных и разнести найденные шаблоны по копиям таким образом, чтобы исключить пересечения. При этом непересекающиеся шаблоны должны быть включены во все копии. Далее для всех полученных копий по шаблонам

формирования фрагмента производится формирование новых слов и замена слов, подошедших под шаблон, на вновь сформированные.

Приведем пример работы синтаксической сегментации.

Пусть у нас имеется следующий набор правил.

<«»; прил.; род +, число +, падеж +><«»; однозначно сущ.; род +, число +, падеж +> ⇒ <однозначно прилагательное><не изменяется>

<«,»; знак; ><«который»; мест.;> ⇒ с первого слова начинается сложноподчиненное предложение

Пусть на вход поступает следующее предложение: «Я увидел человека с красным лицом, который быстро бежал по улице».

Слово «красный» может быть как прилагательным, так и существительным («Борьба красных и белых в ходе гражданской войны…»). По первому правилу вариант существительного будет отсеян. По второму правилу мы заранее определим, что фрагмент «…, который бежал по улице» является сложноподчиненным предложением, и не будем разбирать другие варианты, например, перечисление «… лицом, … улице».

Уровень омонимии может быть снижен на этапе синтаксической сегментации за счет поиска часто употребляющихся конструкций, оборотов и словосочетаний. Так, например, в научной литературе часто употребляются такие конструкции, как «Под … будем понимать …», «Допустим, что …» и т.д. При этом в первой фразе определяемое слово должно находиться в творительном падеже, а основное слово определения и согласуемые с ним слова (обычно прилагательные и местоимения) – в винительном. Найдя подобную конструкцию можно с большой долей вероятности утверждать, что определяемое слово является существительным, а определяющая конструкция отвечает конкретным требованиям. Исходя из этого, можно отбросить часть омонимов, не подходящих под указанный шаблон.

Используя подобные положения, на ВМК МГУ был разработан язык лексико-синтаксических шаблонов (http://lspl.ru/), служащий, правда, для несколько иных целей. Язык шаблонов позволяет на основе отдельных слов и отношений между ними описывать целые конструкции. Отдельное слово описывается следующим образом: часть речи <нормальная форма; список параметров через запятую>. Параметры записываются в формате имя=значение. При необходимости нормальная форма и параметры могут опускаться. Приведем несколько примеров записи отдельных слов.

A<важный; c=nom, g=fem> - описывает формы «важная» и «важна», так как показатель формы не указан.

A<важный> - описывает все формы слова «важный».

V<t=pres, p=3, n=plur> - описывает любой глагол в настоящем времени, третьем лице множественного числа.

Каждый шаблон обладает именем и в него может входить несколько шаблонов для слов. При совпадении части речи у слов шаблона проводится их нумерация.

N1 N2 – два последовательно идущих произвольных прилагательных.

Полная итерация (ноль и более употреблений) некоторой конструкции обозначается при помощи фигурных скобок. При этом имеется возможность указать в треугольных скобках количество повторений этой конструкции.

{A}<1,3> N - от одного до трех прилагательных, после которых идет существительное.

Конструкция, заключенная в квадратные скобки, считается факультативной.

{A} N ["не"] V – существительное, перед которым может идти произвольное количество прилагательных, за которым следует глагол, перед которым может стоять «не».

Шаблон позволяет задавать альтернативы одной конструкции с использованием символа |. Также язык позволяет задавать согласование параметров между отдельными словами конструкции.

A<тяжелый> N <A.g=N.g, A.n=N.n, A.c=N.c>

A<тяжелый> N <A=N> - прилагательное «тяжелый» в произвольной форме, за которым идет существительное, согласующееся с данным прилагательным.

Для шаблонов задается главное слово, от которого берутся все лексические параметры конструкции. Имя конструкции может быть использовано в прочих шаблонах.

AP = A|Pa – шаблон AP задает последовательность из одного прилагательного или причастия.

AN = {AP} N <AP=N> (N) – шаблон AN задает полную итерацию AP (прилагательных или причастий), за которой следует существительное, являющееся главным словом конструкции, согласующееся с первой конструкцией.

ACT = AN V <AN=V> - шаблон ACT задает последовательность из конструкции AN и согласующего с ней глагола.

NP = AN1 {AN2<c=gen>} (AN1) – существительное с подчиненной ей группой существительных.

DT = NP1<c=acc> ["мы"] «назовем» NP2<c=ins> <NP1.n = NP2.n> - конструкция вида «… мы назовем …».

Используя подобные шаблоны можно как проводить упрощенный синтаксический анализ, так и снимать омонимию в определенных конструкциях.