БОЛХОВИТЯНОВ А.В., ЧЕПОВСКИЙ А.М.

АЛГОРИТМЫ МОРФОЛОГИЧЕСКОГО АНАЛИЗА КОМПЬЮТЕРНОЙ ЛИНГВИСТИКИ

Москва, 2013г.

Рассматриваются методики морфологического анализа в задачах автоматической обработки текстов на естественном языке. Рассматривается обобщенная схема алгоритмов аналитического выделения основ. Подробно описаны алгоритмы аналитического выделения основ для большинства индоевропейских языков.

Пособие предназначено для студентов, изучающих курсы «Обработка текстовой информации», «Компьютерная лингвистика», «Поисковые системы», «Математические основы компьютерной лингвистики», «Вычислительная математика и прикладная статистика в издательском деле», «Математические и статистические методы обработки текстовой и графической информации», «Математика и информатика», и для всех интересующихся методами обработки текстовой информации в информационных системах и проблемами компьютерной лингвистики.

ОГЛАВЛЕНИЕ

ВВЕДЕНИЕ

Автоматическая обработка больших объемов текстов на естественном языке включает в качестве одной из ключевых проблем морфологический анализ слов с целью получения либо словоформы, представляющей парадигму слова, и всей его парадигмы, либо основы встреченного в тексте конкретного словоупотребления. Без определения неизменяемой для данной словоформы основы невозможно осуществлять корректную индексацию и последующий поиск информации по построенному индексу, решать более сложные задачи искусственного интеллекта, связанные, например, с классификацией текстов, определением психолингвистических характеристик текста.

Разделяют два принципиально различающихся подхода к морфологическому анализу: методы, основанные на словарях, и бессловарные морфологические анализаторы. Наиболее полно задача морфологического анализа решается словарными анализаторами, позволяющими определять грамматические характеристики для словоупотреблений в текстах на естественных языках, без которых становится невозможно, например, проводить фрагментацию текстов, выделение именных и предложных групп, однородных членов предложения.

Однако методы морфологического анализа с помощью словаря обладают одним главным недостатком – если анализируемой словоформы нет в словаре, то и получить какую-либо морфологическую информацию о ней невозможно. В условиях активно развивающихся и модернизирующихся языков, применение только словарных морфологических компьютерных моделей не решает проблему полноты определения всех возможных слов, встречаемых в текстах на естественном языке. Ситуация осложняется при обработке текстов, возникающих при телекоммуникационном взаимодействии абонентов.

Для решения проблемы морфологического анализа при построении поискового индекса для полнотекстового поиска разработаны аналитические методы выделения основ, не использующие лингвистические словари. Методы

такого типа не решают полностью задачу морфологического анализа: трудно определить часть речи и грамматические признаки словоформ. Однако аналитические алгоритмы оказываются эффективны для задач индексации текстовых массивов, создания процедур работы со словарями естественных языков. Описанию таких алгоритмов посвящен основной материал данного учебного пособия.

Данное учебное пособие предназначено для слушателей курсов «Компьютерная лингвистика», «Обработка текстовой информации», «Математические основы компьютерной лингвистики», «Вычислительная математика и прикладная статистика в издательском деле», «Математические и статистические методы обработки текстовой и графической информации», «Математика и информатика».

Изучивший материал данного учебного пособия должен

Знать: основы информационной культуры, назначение и квалификацию программных средств цифровой обработки информации, принципы и методы их использования в издательском деле, иннновационные особенности различных видов печатных и электронных средств информации, основные технологические процессы производства печатных и электронных средств информации;

Уметь: использовать компьютерную технику в решении конкретных практических задач, разрабатывать предложения по организации информационного пространства с использованием современных технологий;

Владеть навыками: использования программного обеспечения в процессе подготовки печатных и электронных изданий, работы с прикладными программными средствами.

Обладать следующими компетентциями: использовать информационные технологии и программное обеспечение при разработке издательских проектов (ПК-14); использовать современные достижения науки в практической издательской деятельности (ПК-41).

1. МОРФОЛОГИЧЕСКИЙ АНАЛИЗ ПРИ ОБРАБОТКЕ ТЕКСТОВ НА ЕСТЕСТВЕННЫХ ЯЗЫКАХ

Требования к усвоению данного раздела учебного пособия:

Знать: основы информационной культуры;

Уметь: разрабатывать предложения по организации информационного пространства с использованием современных технологий;

Владеть навыками: использования программного обеспечения в процессе подготовки печатных и электронных изданий.

1.1. Морфологические модели

Задаче полного автоматического морфологического анализа словоформ естественного языка посвящено множество теоретических исследований и практических разработок [4, 31, 37, 109]. В частности, такого рода исследования проводились в рамках работ по созданию систем автоматического перевода [26, 27], информационно-поисковых систем [46, 47, 109], систем орфографического контроля [8]. При этом многие вопросы, связанные как с задачами автоматизации лингвистических знаний в общем, так и с задачей автоматического морфологического анализа в частности, остаются либо не решены, либо решены лишь частично [16].

Среди работ, рассматривающих системы морфологического анализа, можно условно выделить теоретические, в рамках которых строятся формальные модели русской морфологии, и прикладные, в которых ставится цель разработки программных систем автоматического морфологического анализа.

В теоретических работах строятся формальные модели морфологии, направленные в первую очередь на решение задач синтеза. Для таких моделей требуется наличие больших словарей со сложной структурой. Данные модели предназначены для описания широкого круга морфологических явлений, что зачастую оказывается избыточным в прикладных системах автоматического анализа текстовой информации.

Теоретические основы лингвистической морфологии, лежащей в основе систем автоматического морфологического анализа, изложены В.А. Плунгяном и И.А. Мельчуком в работах [32, 33, 41, 42]. Авторами дается систематизированное и полное изложение формальных аспектов морфемики, приводятся особенности морфологических явлений для различных языков, в первую очередь для индоевропейских языков. Исследуются различные механизмы формирования слов соответствующих языков, что дает возможность разрабатывать модели и алгоритмы для конкретных языков и языковых семей.

В компьютерных разработках в области прикладной лингвистики встречается очень много некорректных с точки зрения классической лингвистики терминов. Уточним некоторые используемые здесь и далее понятия, следуя учебному пособию В.А. Плунгян [42].

Ключевым понятием для нас будет являться понятие парадигмы, описывающее некоторые замкнутые классы словоформ, содержащих грамматические характеристики. Под парадигмой будем понимать множество словоформ с одинаковым лексическим значением и с разными грамматическими значениями. Парадигма рассматривается как основной объект грамматического описания. Предусматривается, что в парадигму объединяются словоформы с общей основой слова. Единство основы может подразумевать как прямое совпадение образующих их строк символов, так и известное в лингвистике варьирование основы (например, настоящего и прошедшего времени).

Единицей словаря будем считать лексему. Лексема рассматривается как общая часть парадигмы, как слово в отвлечении от его грамматических значений. Лексему «представляет» некий объект, который можно по договоренности считать самой лексемой. Таким представителем лексемы может быть одна из словоформ, либо основа всех или некоторых словоформ.

Мы в дальнейшем будем пользоваться именно понятием «представляющей словоформы», понимая под этим вполне конкретные

словоформы в зависимости от части речи слова. Например, для существительных в качестве представляющей словоформы выбирается словоформа номинатива единственного числа. Для глаголов в качестве представляющей словоформы рассматривается словоформа инфинитива, словоформа второго лица единственного числа императива или словоформа первого лица единственного числа настоящего времени.

Таким образом, в дальнейшем в данном пособии мы пользуемся понятиями парадигма слова, лексема, представляющая словоформа, основа слова.

В работе [10] приводится полное описание морфологических явлений русского языка и методы их формализации. Ключевыми особенностями данной модели являются: морфологический и синтаксический роды; отнесение темы глагола к флексии; метод описания чередований для существительных и различие для супплетивных основ; выделение специальных признаков глагола, позволяющих описать все способы образования различных форм; отсечение частицы «не» у существительных и прилагательных.

К недостаткам такой модели можно отнести ее сложность, так как требуется несколько уровней представления морфологической информации и специальные грамматики для перехода от одного уровня к другому, а также избыточность грамматических признаков, часть из которых вводится в модель для описания крайне редких частных случаев.

Модели, построенные на основе морфологического словаря, позволяют осуществлять более полный анализ словоформы, так как оперируют большим числом грамматических признаков.

Морфологический словарь задается как словарь словоформ либо как словарь основ или псевдооснов слов с полной морфологической информацией, характеризующей часть речи и присущие ей грамматические признаки (род, число, падеж и другие подобные признаки). В обоих перечисленных выше случаях словарь позволяет получить необходимую для анализируемой

словоформы морфологическую информацию. Именно такой словарь и будем называть морфологическим.

Работа программ морфологического анализа, основанная на морфологических словарях, ограничена объемом и актуальностью использованных словарных баз данных.

1.2. Особенности естественных языков

Естественные языки обладают некоторыми свойствами, накладывающими специальные ограничения на методы и алгоритмы обработки текстов, написанных на таких языках [33, 34, 38, 39, 40].

Каждый носитель языка, объединенная родственными или профессиональными связями группа носителей языка обладают индивидуальными словниками (вернее, тезаурусами) и собственными интерпретациями, которые формируют различные понимания текстов. Это указывает на многоцелевую, многогранную направленность процессов передачи информации и фиксирует ситуацию, в которой между отправителем и получателем текстов на естественных языках должна существовать обратная связь. Другими словами, автор текстов (сообщений) должен приспосабливаться к индивидуальным характеристикам пользователя (получателя) текстов на естественных языках.

Наиболее ярко эта проблема проявляется в семантических интерпретациях отдельных слов. В [39] приведен характерный для русского языка пример: определение границ семантических полей множества слов {ночь, утро, день, вечер}. Легко показать, что носители русского языка по-разному определяют те временные границы, которые связывают с различными словами из приведенного множества.

Нельзя провести четких границ между рассматриваемыми понятиями: границы колеблются в результате размытого восприятия действительности со стороны разных пользователей языка.

Лингвистические множества принадлежат к типу нечетких множеств, которые обнаруживаются на всех уровнях языка. Фонема, как звуковая единица, объединяющая различные звуковые оттенки, принадлежит к нечеткому множеству. В частности, в разговорной речи происходит совпадение оттенков различных фонем. Нечеткими множествами являются диалекты и говоры языка, его литературные варианты.

В современных системах указанная обратная связь неосуществима, а это приводит к потере информации, содержащейся в текстах, обрабатываемых компьютером в автоматическом режиме.

Нечеткие языковые объекты и размытость границ лингвистических множеств ограничивают использование многих традиционных для математики и компьютерных наук методов, в частности, методов дискретной математики. Попытки создания универсальных, четких (дискретных) искусственных языков, которые были бы семантически выразительнее любых естественных языков. Дискретная формализация при построении модели ЕЯ оказалась нежизненной при решении главной задачи компьютерной лингвистики – задачи машинного перевода текстов на естественных языках. Несостоятельной оказалась и идея универсального языка-посредника.

Лингвистические процессоры построены на использовании конечных множеств, в них не удается воспроизвести бесконечность и полиморфность лингвистических множеств естественных языков. Прямолинейная замена нечетких лингвистических множеств четкими множествами формализованных языков приводит к существенным потерям информации, содержащейся в обрабатываемых на компьютерах текстах на естественных языках.

Р.Г. Пиотровским в [39] был сформулирован «Парадокс Ахиллеса и черепахи» в применении к компьютерной лингвистике. Суть этого парадокса - в непрерывности развития языка. Любой лингвистический процессор разрабатывается на основе некой формализации, опирающейся на ситуацию с состоянием языка, которая устаревает уже к моменту реализации данного лингвистического процессора. Изменяется словарный состав языка, его

терминологические множества, может измениться словообразование, практика применения синтаксиса. Получается, что созданный процессор не обеспечивает стопроцентное извлечение информации из текстов, для обработки которых данный процессор предназначался.

Увеличение скорости изменения и модернизации естественных языков в эпоху развития массовых коммуникаций и средств телекоммуникационного взаимодействия роль «Парадокса Ахиллеса и черепахи» возрастает и проявляется во всех областях текстовой информации.

Здесь имеет смысл упомянуть глоттохронологию**,** именуемую также глоттохронологической лексикостатистикой, или просто лексикостатистикой, представляющую собой исследовательский метод, разработанный в 1940-х годах американским лингвистом Моррисом Сводешем первоначально для датировки времени расхождения родственных языков [38, 53]. Термин «глоттохронология» образован от греческого glossa "язык" и дословно означает "языковая хронология. При создании глоттохронологии Сводеш вдохновлялся примером успешно применявшегося к тому времени метода радиоуглеродной датировки археологических находок.

Основной постулат Сводеша заключался в том, что в словарном составе всех языков мира можно выделить некоторое подмножество слов, отличающихся особой стабильностью. Эти слова обозначают некоторые базовые общечеловеческие понятия, и Сводеш разработал список таких понятий, в который вошли названия частей тела, некоторых живых существ, природных явлений, элементов рельефа, имен родства, некоторые местоимения, прилагательные, обозначающие элементарные геометрические понятия и цвета, ряд глаголов и некоторые другие. Были составлены списки из 100 и из 200 элементов (так называемые стословный и двухсотсловный списки Сводеша).

Доля слов из списка Сводеша, которая сохраняется в течение некоторого промежутка времени (не будут заменены другими словами), примерно одинакова для всех языков. Из двухсотсловного списка через 1000 лет

сохраняется примерно 81% слов, из стословного – примерно 86%. Темп сохранения/не сохранения слов в течение выбранного времени относительно постоянен, а шансы каждого слова сохраниться или не сохраниться равны.

В словаре данного языка можно выделить специальный фрагмент, который называется ядром, стабильной частью. Можно указать список значений, которые в любом языке обязательно выражаются словами из основной части. Эти слова называются основным списком. Все слова, составляющие основной список, имеют одинаковые шансы сохраниться на протяжении некоторого интервала времени.

Сравнительно-исторический метод формировался и оттачивался на материале индоевропейских языков, уникальным свойством которых является большое количество сильно различавшихся уже несколько тысячелетий назад языков (древнегреческий, древнеперсидский, латинский, санскрит, древнеармянский) с рано возникшими фонетическими письменностями, хорошо фиксирующими звуковой облик языка, долгой письменной историей и обилием сохранившихся текстов. Это сделало возможным осуществление сопоставлений между языками, отстоящими от современности на несколько тысячелетий, да еще и с возможностью проверки гипотез о развитии отдельных языковых ветвей данными таких языков, как латинский для романских языков и ведийский и санскрит для индийских – фактически письменно засвидетельствованных праязыков.

Вторая половина двадцатого века была ознаменована существенными научно-техническими достижениями. На научно-технический прогресс и соответствующие изменения в обществе сильно реагировал язык общения индивидуумов социума. Письменный язык отражает те языковые новации, которые возникают как в разговорной речи, так и в профессиональных диалектах языка. Поэтому этот период характеризовался появлением новой терминологии, заимствований из других языков. Этот процесс существенно

изменил лексику, модернизировал классические процессы словообразования и словоизменения [13, 34]. Главным в этом процессе было существенное отставание словарей от языковой практики, что стало затруднять автоматизацию процессов обработки текстов на естественных языках.

В девяностых годах прошлого века главными тенденциями изменения лексики многих языков являлись иноязычные заимствования. Данная тенденция была наиболее ярко выражена в русском и других славянских языках в связи с изменениями политического и экономического устройства общества. Наиболее частыми заимствованиями сегодня являются англицизмы. Причем в русском языке английские слова начинают вытеснять другие более ранние заимствования: например, английское «сэндвич» вместо немецкого «бутерброд» [13]. Такого рода заимствования могут оказаться производящими основами, на базе которых возникают новые слова (например, «пиар» — «пиарщик» [13]). Таким образом может быть задействована система словообразования того языка, в который внедряются заимствования. А это, в свою очередь, требует существенных изменений в морфологических словарях языка.

Особое влияние на модернизацию языка оказывает широкое распространение в последние два десятилетия средств телекоммуникаций, подразумевающих как электронную почту, так и форумы, чаты, блоги, средства обмена короткими сообщениями [21, 34, 58]. Все чаще говорится о медийном варианте конкретного национального языка.

Например, в медийном варианте немецкого языка используются глагольные основы слова без окончаний определенного типа (*freu*) [58]. Такой эффект характеризуется с точки зрения морфологии как «свёрнутость» морфологической парадигмы до глагольной основы. Велика доля сокращений слов в чатах и аналогичных коммуникационных контактах, доходящая до нескольких процентов от всех словоупотреблений в текстовых сообщениях.

Естественно, что использование таких форм сильно осложняет использование традиционного морфологического анализа.

Распространение в телекоммуникационных текстах лексических единиц, ранее относившихся к жаргону, арго, сленгу, является еще одним фактором, требующим существенного изменения во времени морфологических и других словарей естественного языка, необходимых для автоматической обработки текстов.

1.3. Лингвистическая типология

Для понимания возможностей применения различных методов анализа к разным языкам необходимо рассмотреть лингвистическую типологию (классификацию) языков [9, 17, 28, 43, 48, 44]:. В основе лингвистической типологии лежат две характеристики. Первая – метод выражения грамматических значений, который разделяет языки на аналитические и синтетические. Вторая – способ соединения морфем в слове, характеризующая, в частности, флективность и агглютинацию того или иного языка.

1.3.1. Аналитизм и синтетизм естественных языков

По типу выражения грамматических характеристик обычно выделяют аналитизм и синтетизм языков.

Аналитизм характеризуется тем, что грамматические значения слов языка передаются не формами слов, а вне пределов рассматриваемых слов. Для этого могут быть использованы отдельные служебные слова, которые могут быть представлены как самостоятельными словоформами, так и местоимениями, предлогами или частицами. Грамматические формы выявляются по

соотношению с другими словами, определяясь в контексте, а не по формальным признакам слова.

В языках синтетического типа грамматические значения выражаются аффиксами в составе словоформы. То есть формируется фонетическое слово с опорным лексическим корнем, дополненное аффиксами.

Таким образом, при аналитическом выражении грамматических значений слова состоят из малого числа морфем (в пределе — из одной морфемы), при синтетическом — из нескольких морфем. Предельной степенью синтетизма является полисинтетизм, характерный для языков, слова в которых состоят из большого числа морфем.

К языкам, в которых преобладают аналитические способы выражения грамматических значений, относятся романские языки, а к синтетическим языкам большинство славянских языков и, например, немецкий язык.

Количественная оценка для измерения степени синтетичности-аналитичности языков была предложена, как указывается в [35], в 1960 году Д. Гринбергом. В качестве меры синтетичности Д. Гринбергом была предложена характеристика, вычисляемая как количество морфем, приходящихся на одну словоформу. Эта характеристика отражает степень распространенности аффиксации в языке. Естественно, что минимальное значение индекса синтетичности равно 1 и указывает на максимальную аналитичность языка. Если индекс синтетичности превышает значение 2, то в языке преобладает синтетизм.

В таблице 3.1 приведены индексы синтетичности по Гринбергу, взятые из работы [34], для трех индоевропейских языков и для вьетнамского языка как примера языка с очень маленьким индексом синтетичности.

Таблица 1.1. Индексы синтетичности языков.

Язык	**Индекс синтетичности**
Русский	2.39
Английский	1.68
Немецкий	1.97
Вьетнамский	1.06

Другая оценка степени аналитичности языка вводится в [38] на основе суммирования контекстных ограничений, которые подсчитываются через количество лексико-грамматической информации слова. Способы вычисления количественных характеристик для измерения смысловой информации в тексте подробно описаны в [38] и в данном пособии не рассматриваются.

В таблице 1.2 приведены подсчитанные в [38] коэффициенты аналитичности четырех индоевропейских языков. Разница в полтора раза между коэффициентами аналитичности для русского и английского языков соответствует лингвистическому пониманию степени аналитизма и синтетизма этих языков. Малый коэффициент аналитичности для аналитического французского языка определяется флективностью служебных слов (артикль, притяжательные местоимения и другие), которые часто встречаются в текстах на французском языке.

Таблица 1.2. Коэффициенты аналитичности языков.

Язык	**Коэффициент аналитичности**
Русский	22.57
Английский	35.00
Французский	23.55
Румынский	27.01

Приведенные коэффициенты синтетичности и аналитичности языков являются интересной, но не однозначной характеристикой для оценки

возможности применения аналитических алгоритмов морфологического разбора. Например, низкий коэффициент аналитичности по Р.Г. Пиотровскому не означает наличие большого количества флексий для словообразовательных схем основного лексического состава языка. Но эти характеристики могут служить для предварительных оценок сложности работы алгоритмов морфологического разбора.

1.3.2. Флективность и агглютинация

С точки зрения типов морфологической организации слов в естественных языках выделим флективность и агглютинацию [28, 40, 41].

Агглютинация суть образование новых грамматических форм и слов посредством присоединения к основе слова аффиксов с четко выраженными свойствами таким образом, что границы морфов не изменяются. Каждый аффикс имеет единственное значение и каждая функция выражается одним определенным аффиксом.

Явление агглютинации сводится к присоединению без изменения словообразующего суффикса к морфу базового слова. В этом случае каждый конкретный суффикс, каждый аффикс языка обладают конкретными семантическими, психолингвистическими значениями, функциями.

Флективность характеризуется отсутствием четких границ между морфами, а сами морфы могут иметь несколько различных грамматических значений. По сути, семантические и формальные границы между морфами в флективных языках плохо различимы.

Для слов флективных языков характерно, что корень слова несет главные лексико-грамматические значения. С одной стороны, в одном словоизменительном аффиксе могут содержаться различные грамматические категории (например, падеж и число в существительных русского языка). С

другой – одно и то же грамматическое значение может формироваться различными аффиксами. Флективность языков характеризуется системой чередований, возникающих на стыках морфем.

Для агглютинативных языков (например, тюркских) характерна достаточно развитая система словообразовательной и словоизменительной аффиксации, грамматическая однозначность аффиксов, отсутствие чередований. Для флективных языков (например, славянских) характерны многозначность грамматических морфем (например, в русском языке окончание определяет род, число и падеж).

Среди языков с флективной морфологией есть как синтетические языки (например, русский, немецкий), так и языки аналитические (например, французский).

1.3.3. Активные процессы в естественном языке

Отметим те основные процессы в словообразовании и морфологии современного языка общения людей, которые существенно влияют на качество решения задач автоматической обработки текстов на естественном языке [34, 131].

В современных языках наблюдается рост аналитических черт в морфологиях языков синтетического типа, что выражается в изменении ряда морфологических показателей, таких как сокращение числа падежей и рост несклоняемых имен собственных в языках флективного типа.

В русском языке эти процессы начались достаточно давно. Например, исчезновение звательного падежа с сохранением соответствующих функций и интонаций, например, в междометных формах («боже мой», «господи») [14]. Тенденции, особенно ярко выраженные в последние десятилетия, не склонять многие топонимы, первые части сложных терминов (например, «инженер-экономист»). Таким образом, сокращение числа падежей, возникновение большого числа несклоняемых слов привело к большему влиянию

аналитических черт языка при формировании текста. Формирование смысловых единиц языка становится все более зависимым от контекста и в меньшей мере определяется процессами словоизменений.

Другим фактором активных процессов в современных естественных языках является рост агглютинативности в процессе словообразования.

В русском языке это проявляется в ослаблении чередования при соединении морфем [14]. Это проявляется при построении производных слов от заимствований и топонимов. Характерны процессы наложения морфем, когда совпадают конечные части основы и присоединяемого суффикса (например, Динамо - динамовский).

Показатели аналитичности, флективности и агглютинативности языков являются отражением различных моделей словообразования и словоизмения естественных языков. Они могут служить показателями использования правил отсечения аффиксов при построении алгоритмов морфологическго разбора.

1.4. Морфологический анализ на основе словарей

Работа программы морфологического анализа, основанная на морфологической информации, хранимой в специальном словаре, может быть организована разными способами в зависимости от распределения информации между программными компонентами и словарями, реализуемыми моделями морфологии и форматами хранения данных.

Морфологический словарь задается как словарь словоформ либо как словарь основ или псевдооснов слов с полной морфологической информацией, характеризующей часть речи и присущие ей грамматические признаки (род, число, падеж и другие подобные признаки).

1.4.1. Развитие систем словарных морфологий

Можно выделить два способа построения машинных словарей для систем автоматического морфологического анализа.

Первый подход построен на академической лингвистической модели описания морфологических явлений рассматриваемого языка. В рамках такой модели выделяются основные парадигматические классы, соответствующие заданному типу склонения и спряжения, а также правила регулярных чередований. Различные нерегулярные явления языка задаются в виде специальных списков исключений. При этом машинные словари русского языка, формируемые в соответствии с этим подходом, построены на основе морфологического словаря А.А. Зализняка [20].

В рамках второго подхода любого вида регулярное и нерегулярное чередование рассматривается как часть расширенной псевдофлексии. В такой модели число парадигматических классов для русского языка возрастает до 3000 [36], но рост числа классов при проектировании компенсируется однородностью словаря и отсутствием как исключений, так и правил чередования.

Первый подход нашел применение в программном продукте Inxight LinguistX Platform [122], в основу которого легли разработки сотрудников научно-исследовательского центра Xerox. Данная система позволяет осуществлять морфологический анализ европейских и восточных языков. Наиболее распространенные языковые модули, такие как русский, английский и немецкий, имеют четыре уровня текстового анализа:

1. Уровень лексического анализа, который осуществляет деление исходного текста на предложения и словоформы;
2. Уровень приведения словоформ к словоформе, представляющей парадигму слова;
3. Уровень снятия морфологической омонимии и унификации значений грамматических характеристик

4. Уровень синтаксического выделения именных групп из текстов.

Модуль морфологии состоит из двух компонент:

1. Словарь, в котором хранится словоформа, представляющая парадигму слова, а также все словоформы парадигмы и их грамматические характеристики;
2. Множество правил чередования и орфографических правил.

Словарь состоит из подсловарей, делящихся по селективным признакам и парадигматическим классам. Правила второго компонента записываются с помощью регулярных выражений. Технология анализа построена на применении конечных преобразователей (Finite-State Transducer), что позволяет добиться высокой скорости морфологического анализа.

Примером первого подхода также является система, разработанная С.А. Старостиным в рамках создания рабочей среды для лингвистов [49]. В качестве исходного словаря системы выступает словарь А.А. Зализняка [20]. В рассматриваемой системе алгоритмически реализована интерпретация словарных помет, используемых в словаре, что позволяет осуществлять процедуру синтеза словоформ. Алгоритм анализа устроен следующим образом. Подаваемая на вход словоформа подвергается ряду преобразований на основе заложенных в систему правил, в результате чего получаются все возможные варианты исходной формы данного слова. Далее для каждого построенного таким образом варианта производится поиск его в словаре. Высокая точность приближенного анализа отсутствующих в словаре словоформ достигается за счет большого числа правил. Особенностью данной системы является учет ударений. Однако данная система не оптимизировалась с целью достижения высокого уровня быстродействия и в результате этого не нашла промышленного применения. Кроме этого, разработанная система является языково-зависимой и не может быть обобщена на случай других языков без аналогической разработки обширной системы правил.

Морфологический модуль системы Диалинг [67] спроектирован на основе второго подхода. Предлагаемая в данной системе модель была заимствована авторами из работ Ж.Г. Аношкиной [2]. Система Диалинг использует три типа морфологических словарей:

1. Большой словарь, базирующийся на словаре А.А. Зализняка [20];
2. Словарь имен собственных;
3. Словарь топонимов.

Для каждой подаваемой на вход процедуре морфологического анализа словоформы выдается множество морфологических интерпретаций следующего вида: словоформа, представляющая парадигму слова, морфологическая часть речи, множество наборов грамматических характеристик. Морфологическая часть речи определяется традиционным образом согласно классической грамматике русского языка [45] за исключением того, что отдельные специфические морфологические формы получают в системе статус части речи.

Система морфологического анализа Информэлектро, разработанная в начале 70-ых годов группой лингвистов под руководством Д.Г. Лахути и Г.А. Лесскиса [23], реализует второй подход к построению машинных словарей для систем автоматического морфологического анализа. Ее важной особенностью является наличие примитивной модели синтаксического управления.

На факультете ВМиК МГУ в 80-е и 90-е годы разрабатывалась морфологическая модель русского языка в рамках создания системы искусственного интеллекта TULIPS-2 [29]. Анализ подаваемой на вход словоформы в морфологическом модуле системы происходил путем отделения постфиксов и последующего поиска полученной основы в словаре основ. В системе не применяются специализированные алгоритмы и структуры данных для хранения словаря, вместо этого используется система управления базой данных.

В компьютерной морфологической модели русского языка в системе TULIPS-2 слова рассматриваются, в общем случае, как состоящие из основы, окончания и, возможно, постфикса. С каждой основой в словаре ассоциированы такие характеристики, как:

- номер словоизменительного класса;
- номер парадигматического класса;
- номер чередования;
- номер исключения,

кроме этого при каждой основе в словаре указан набор ее неизменяемых грамматических характеристик.

Дальнейшее развитие методов автоматического морфологического анализа текстов на русском языке с использованием словаря А.А. Зализняка осуществлено А.Ф. Гельбухом [15]. В своей работе А.Ф. Гельбух реализует эффективную с точки зрения быстродействия на ЭВМ компьютерную морфологическую модель для флективного языка (в данном случае рассматривается русский язык). В системе используется словарь основ с приписанным каждой основе набором неизменяемых грамматических характеристик и списка морфов с отражаемыми ими грамматическими характеристиками. Анализируемая словоформа первоначально разбивается всеми возможными вариантами на основу и окончание, происходит поиск основы в словаре основ. Если основа найдена, то из словаря берутся неизменяемые грамматические характеристики. Затем анализируется окончание. Строятся все возможные наборы морфов, которые могут сформировать данное окончание и затем, используя статистику взаимной встречаемости морфов в окончаниях, выбирается самый вероятный вариант морфного состава окончания. На основе полученного списка морфов, а точнее соответствующего списка морфем, строится набор грамматических характеристик, соответствующий словоизменительной компоненте морфологии языка.

1.4.2. Метод аналогии в компьютерной морфологии

В морфологических анализаторах Диалинг и Информэлектро предсказание грамматических характеристик словоформ, отсутствующих в словаре, устроено по одному принципу. Если подаваемая на вход анализатору словоформа не была найдена в словаре, то ищется словоформа, максимально совпадающая с конца с анализируемой словоформой. Этот принцип был впервые предложен в 80-ые годы Г.Г. Белоногoвым [6] и был назван методом аналогии. Дальнейшее развитие этого метода предполагает возможность применения метода аналогии в задачах автоматического синтаксического и семантического анализа текстов на естественном языке [7]. Для системы Диалинг в [36] предлагается взвешивание гипотез морфологического предсказания с использования метода корреляции. Матрицы корреляции строятся для основ и значений классифицирующих грамматических категорий. Гипотезы, имеющие максимальную корреляцию, объявляются наиболее правдоподобными.

В основе метода аналогии применительно к задаче автоматического морфологического анализа лежит гипотеза о том, что слова, имеющие одинаковые сочетания букв на концах, с высокой вероятностью имеют совпадающие модели и словообразования, и словоизменения (соответственно, и совпадающие наборы морфологических характеристик, таких как род, число, падеж).

Процедура автоматического морфологического анализа слов в рамках данного подхода может быть реализована следующим образом. Строится словарь словоформ, в котором каждая словоформа сопровождается набором грамматических характеристик. Затем словарь инвертируется и уже инвертированные словоформы могут быть помещены в массив. Тогда в процессе анализа подаваемые на вход словоформы также инвертируются и ищутся в массиве методом дихотомии [25]. Можно существенно снизить объем занимаемой памяти и существенно повысить быстродействие процедуры

анализа, если в качестве структуры данных для инвертированного словаря использовать древовидную структуру данных типа бор [22]. Также можно сократить объем инвертированного словаря словоформ, если исключить из него начальные части словоформ, не оказывающие влияния на результат анализа. Для этого достаточно оставить у каждой пары рядом стоящих словоформ слева только совпадающие конечные буквосочетания и еще по одной букве, которые не совпадают. Таким образом, объем словаря по сравнению с исходным вариантом может сократиться в десять раз [7], и при этом не будет наблюдаться потерь в качестве морфологического анализа.

Метод аналогии нашел применение и в коммерческих информационно-поисковых системах общего назначения. В частности, в [47] указывается, что модуль морфологического анализа Mystem, используемый в информационно-поисковой системе «Яндекс», построен на основе метода аналогии. В качестве словаря данной системы также используется словарь А.А. Зализняка [20].

Для предсказания грамматических характеристик отсутствующих в словаре слов в «Яндексе» используется «быстрословарь», представляющий собой дополнение к основному словарю, и, соответственно, построенный на тех же самых принципах.

Схема построения данного уменьшенного словаря выглядит следующим образом. Составляются частотные словари на основе коллекции документов, проиндексированных поисковой системой «Яндекс», и запросов пользователей. Для каждого слова определяется частота его встречаемости во всем корпусе. Также имеется составленный вручную корпус, в котором для каждого слова запроса указывается вся парадигма. Из данного размеченного корпуса выбираются все слова, присутствующие в основном словаре, за исключением стоп-слов и самых частотных слов. Для каждого выбранного таким образом слова строятся все возможные морфологические интерпретации, и для каждой из них строится список значений всех факторов предсказания. В результате получается обучающая выборка.

На обучающей выборке строится модель, способная предсказывать основу и модель окончания. Качество модели измеряется по тестовой выборке. Затем для самых частотных несловарных слов строятся все гипотезы, предсказывающие основы и модели окончаний. По самым лучшим гипотезам строятся парадигмы. После отсеивания худших вариантов получается «быстрословарь», дополняющий основной словарь поисковой системы. Отметим, что данный дополнительный словарь может быть откорректирован вручную перед добавлением к основному словарю.

В рамках описанного подхода рассматривается большое количество различных факторов: факторы внутреннего состава слова (длина псевдоокончания, отношение числа согласных букв к общему количеству букв слова), контекстные факторы, факторы на основе частот встречаемости конкретных словоформ и др.

Еще одной работой, в которой используются методы распознавания для автоматического пополнения морфологического словаря, является работа [55]. Автором также предлагается алгоритм предсказания модели словоизменения несловарных словоформ. Однако низкая точность данного алгоритма позволяет использовать его только в качестве дополнения к ручному методу пополнения морфологического словаря. При этом результаты алгоритма должны проходить проверку лингвистом перед окончательным попаданием в словарь.

Общая схема алгоритма может быть представлена в виде последовательности шагов:

1. выделить несловарные словоформы из корпуса;
2. по каждой словоформе построить все возможные гипотезы лемматизации; объединить гипотезы без дубликатов;
3. отфильтровать гипотезы по некоторому признаку;
4. кластеризовать гипотезы, выделив компоненты связности в биграфе гипотезы-словоформы;

5. из каждого класса по некоторому критерию выбрать одну или несколько наилучших гипотез.

В качестве количественных характеристик для обучения выступают: частотность словоформ, покрываемых гипотезой; число лексем в словаре с тем же постфиксом псевдоосновы заданной длины; оценка вероятности вхождения грамматических форм в корпус. Для сокращения числа гипотез используются частотные характеристики словоформ, входящих в корпус. Проблема отсутствия размеченного корпуса с несловарной лексикой решается путем разделения множества лексем существующего словаря на обучающее и тестовое подмножества.

В силу низкой точности анализа данный алгоритм не применяется на практике и в данный момент может представлять лишь теоретический интерес.

1.4.3. Популярные морфологические модели русского языка

И.С. Ашманов в работе [3] привел описание собственной компьютерной морфологической модели русского языка. Она имеет много общего с описанными ранее моделями. Так, в ней предполагается, что каждое слово русского языка может быть отнесено к одной из 12 больших групп, называемых частями речи, которые при этом могут не совпадать с общепринятым значением этого термина в классической грамматике. Каждая часть речи характеризуется своим набором грамматических категорий (род, число, падеж и др.) и своей парадигмой словоизменения. Рассматриваются следующие части речи: существительные, прилагательные, предлоги, количественные числительные, порядковые числительные, собирательные числительные, местоимения, местоименные прилагательные, неизменяемые слова, глаголы несовершенного вида, глаголы совершенного вида, сокращения с точкой. В модели принимается, что русское слово, относящееся к конкретной части речи, всегда имеет одно и то же число форм (например, всякое существительное имеет двенадцать словоформ — по шесть падежных форм единственного и

множественного числа соответственно). При этом, конечно, некоторые формы у конкретного слова могут быть вырожденными, но тем не менее формально учитываются (как, например, все шесть форм единственного числа для слова брюки).

Каждой части речи соответствуют свои морфологические таблицы, описывающие ее словоизменение. Две неизменяемые части речи — неизменяемые слова и сокращения с точкой — таких таблиц не имеют, так как относящиеся к ним неизменяемые слова имеют только одну словоформу.

В рассматриваемой модели каждое слово представляется в виде тройки ⟨короткая основа, чередование, окончание⟩. Каждому слову в рамках каждой изменяемой части речи соответствует вектор окончаний фиксированной длины (равной количеству словоформ, принятому для данной части речи), который называется классом окончания, а также вектор чередований, называющийся классом чередования, имеющий ту же длину, что и вектор окончаний, и включающий чередующиеся фрагменты всех словоформ данного слова.

Морфологический словарь состоит из статей. Короткая основа и чередование каждого слова составляют вместе «длинную основу», которая и хранится в статье. В общем случае статья морфологического словаря имеет вид:

$$\boldsymbol{C_1 C_2 \dots C_n \$ N_1 N_2, \dots, N_8}$$

где $\boldsymbol{C_1 C_2 \dots C_n}$ — длинная основа длины n, а после разделителя «$\boldsymbol{\$}$» идут номера $\boldsymbol{N_i}$ класса окончаний, класса чередований, словоформы и дополнительных грамматических признаков.

Алгоритм поиска словоформы в морфологическом словаре устроен следующим образом. Анализируемая словоформа разбивается на пары ⟨основа, окончание⟩ всеми возможными способами. Каждая основа ищется в таблице длинных основ, а окончание — в таблице окончаний. Если соответствующие записи найдены, то можно определить начальную форму слова, ее морфологические характеристики и, при необходимости, получить всю парадигму.

Данная модель оказалась крайне эффективной как с точки зрения качества, так и с точки зрения быстродействия, так как на основе информации из таблиц можно построить эффективный бинарный формат хранения такого морфологического словаря.

Схожий подход при осуществлении автоматического морфологического анализа представлен в работе [19]. Описываемая система RCO Morphology предоставляет три типа анализа:

1. точный анализ словоформы по словарю;
2. анализ неизвестного слова на основе системы правил;
3. вероятностный анализ несловарных словоформ на основе метода аналогии.

Рассматриваемая система реализует собственный формат представления морфологического словаря. Выделяется общая для всех словоформ в парадигме неизменяемая часть, которая в некоторых случаях может быть пустой. Остающаяся часть словоформы описывается списком окончаний, который формируется на основе модели словоизменения конкретного типа. Авторы выделяют четыре парадигматических типа:

1. парадигма существительного;
2. парадигма прилагательного;
3. парадигма глагола;
4. парадигма неизменяемого слова.

Для обеспечения высокого уровня быстродействия неизменяемые части словоформ (псевдоосновы) хранятся в инвертированном виде в древовидной структуре данных типа бор. Аналогично хранятся возможные окончания словоформ.

Точный анализ по словарю организован следующим образом. Подаваемая на вход словоформа разбивается на основу и окончание всеми возможными способами, включая случай пустого окончания. Затем выделенное окончание ищется в боре окончаний. Для каждого найденного окончания соответствующая основа ищется в боре основ. Если основа найдена в боре, то

осуществляется проверка на возможность наличия данного окончания при найденной основе. Если проверка завершается успехом, то из узла бора основ извлекается информация о части речи, а из узла в боре окончаний — о грамматических характеристиках.

Анализ на основе правил можно осуществлять, если словоформа имеет характерный аффикс. Постфиксом в рассматриваемой модели называется сочетание неизменяемого суффикса и окончания. На основе морфологического словаря выделяются наиболее часто встречающиеся префиксы и постфиксы. При анализе словоформы осуществляется выделение всех возможных префиксов и постфиксов и производится их поиск в специализированных борах. Такой подход позволяет анализировать словоформы, образованные от известных словоформ. Так, префикс «авиа» может присоединяться и к существительному, и к прилагательному.

Алгоритм вероятностного морфологического анализа отличается от точного тем, что вместо бора основ используется бор суффиксов, автоматически сформированный на этапе построения морфологического словаря. В данный бор попадают концы основ, встречающиеся не менее тридцати раз в словах с одинаковой парадигмой изменения и имеющие длину не более четырех символов при наличии остатка в основе не менее трех символов. Авторами было установлено, что данные показатели являются оптимальными для обеспечения требуемого качества анализа.

Анализ словоформы построен на основе метода аналогии. То есть для каждого возможного окончания анализируемой словоформы ищется наибольшее совпадение псевдоосновы с одним из суффиксов, хранящихся в соответствующем боре. Поиск считается успешным, если в найденном суффиксе и выделенном окончании присутствует парадигма с достаточно высокой частотой встречаемости (более тридцати).

Заметим, что данный подход практически идентичен поиску словоформы в словаре в модели И.С. Ашманова, за исключением незначительных отличий, касающихся вопросов оптимизации хранения информации о грамматических

характеристиках, выражающихся окончаниями. Также аналогична процедура вероятностного анализа словоформы, так как она построена на описанном ранее методе аналогии.

1.4.4. Различные типы словарей в морфологических моделях

В работе [56] предлагается модель русской морфологии, в которой не требуется наличие большого словаря основ. В предлагаемой модели используются следующие словари:

- словарь окончаний и чередований;
- словарь суффиксов;
- словарь префиксов;
- словарь псевдокорней;
- словарь баз;
- словарь основ.

Для каждой единицы в словарях указаны все возможные грамматические характеристики словоформ, в состав которых может входить данная единица. Анализ словоформ в модели производится путем проверки правил сочетаемости сущностей различных словарей.

Подобного рода анализ не использует всей информации из текстовых данных, поступающих на вход системе. В статье не описаны методы формирования словарей, а предлагаемый подход сводится к процедуре сжатия исходного словаря. Таким образом, данный подход нельзя назвать в полной мере бессловарным.

Как было отмечено выше, важным моментом при реализации морфологического анализатора слов флективного языка является описание правил чередования букв в основах слов. В ряде систем чередования образуют расширенную псевдооснову, в других системах чередование формирует расширенную псевдофлексию. Возможным оказывается и подход, при котором

чередования формулируются в виде последовательности правил и алгоритмов. При этом для нахождения предполагаемой основы анализируемой словоформы необходимо до обращения к словарю произвести отсечение флексий и применить правила чередования. Достоинством такого подхода можно считать его естественность описания рассматриваемого явления чередования. К недостаткам можно отнести сложность формирования правил чередования и появление большого количества основ, отсутствующих в словаре.

Важно заметить, что описание чередований в виде системы правил является более естественным для систем, осуществляющих синтез, а не анализ словоформ. В работе [1] приводится описание метода автоматического морфологического анализа через синтез. Так, для каждой лексемы, гипотетически порождающей данную словоформу, синтезируются все ее формы, из них выбираются те, которые совпадают с анализируемой словоформой, и таким образом определяются грамматические характеристики исходной словоформы.

Особое место среди методов автоматического морфологического анализа занимает метод логического умножения. Основы данного подхода изложены С.Я. Фитиаловым в работе [52]. Функция, определенная на словоформах и сопоставляющая каждой словоформе некоторую информацию, называется словарной функцией. Словарная функция всегда может быть задана в виде словаря словоформ. Также эта функция может быть представлена в виде последовательности четырех операций:

1. словоформа как цепочка букв членится на морфемные сегменты;
2. словоформа как последовательность морфемных сегментов заменяется на неупорядоченное множество элементов — морфем;
3. словоформе как множеству морфем приписывается некоторая информация;

4. информация преобразуется в окончательную информацию о словоформе.

Каждой морфеме можно сопоставить информацию, получаемую в результате объединения информации о словоформах, в которые входит данная морфема. Такого рода объединения соответствуют дизъюнкции. Информация о словоформе получается путем пересечения, или логической конъюнкции, информации о морфемах, входящих в ее состав.

Автоматический морфологический анализ методом логического умножения предусматривает наличие словаря основ. Процедуру анализа рассмотрим на примере анализа словоформ русского языка, предложенного Д. Варгой [14]. Сначала производится поиск словоформы в словаре основ. Если полученная псевдооснова не найдена в словаре основ, тогда отбрасывается одна буква справа и поиск повторяется. При отрицательном ответе отбрасывается следующая буква и т.д. Отброшенные буквы образуют окончание. Каждая отброшенная буква считается элементарной единицей анализа. Ей приписывается булев вектор. Число компонентов этого вектора равно числу грамматических категорий, которые могут быть выражены окончанием, частью которого является данная буква. Поскольку предварительно был произведен поиск по словарю основ и установлена часть речи анализируемой словоформы, то имеется возможность одинаковым буквам, входящим в окончания разных частей речи (например, буква «м» в окончании существительного и прилагательного), приписывать различные векторы.

Развитие методов автоматического морфологического анализа привело к появлению специального типа словарей, получившего название «обратные словари». Обратные словари используются во многих системах машинного перевода для определения грамматических характеристик не найденных в словаре слов при автоматическом морфологическом анализе флективных языков [5, 24, 57].

Особенность обратных словарей заключается в структуре представления слов: слова располагаются в нем в алфавитном порядке, начиная от конца каждого слова из словаря. При такой структуре естественным образом объединяются слова, относящиеся к одному словообразовательному или словоизменительному типу. Это свойство обратных словарей реализуется, в частности, в методе аналогии [7]. Таким образом, структура обратного словаря наилучшим образом подходит для решения задач автоматического морфологического анализа для флективных языков, так как в них большая часть грамматической информации располагается в морфологических формантах.

Современные проблемы систем автоматического морфологического анализа словоформ русского языка приведены в работе [49]. Выделяются проблемы, связанные с лексическим анализом, который является исходным этапом на пути к осуществлению автоматического морфологического анализа; проблемы разграничения так называемой словоизменительной и словообразовательной морфологии; проблемы автоматического пополнения морфологического словаря и др.

В заключение отметим, что те описанные выше системы, которые реализуют второй подход к построению морфологических моделей, построенный на рассмотрении регулярных и нерегулярных чередований как частей расширенной псевдофлексии, обладают целым рядом преимуществ: высокое быстродействие, высокая степень отделения лингвистической информации от алгоритмов, возможность осуществления морфологического анализа отсутствующих в словаре словоформ.

1.5. Компьютерная словарная морфология русского языка

1.5.1. Морфологические характеристики словоформы

В рассматриваемой морфологической модели каждая словоформа относится к одной из 24 морфологических категорий. Морфологическая категория отражает часть речи и характеризуется набором грамматических категорий: род, число, падеж, наклонение и др.

Рассматриваются следующие морфологические категории (номер соответствует признаку в компьютерной реализации словаря и базы данных):

0. Неизменяемое слово,
1. Существительное,
2. Прилагательное,
3. Глагол несовершенного вида,
4. Предлог,
5. Глагол совершенного вида,
6. Количественное числительное,
7. Порядковое числительное,
8. Местоимение,
9. Местоименное прилагательное,
10. Собирательное числительное,
11. Сокращение,
12. Латинское слово,
13. Аббревиатура,
14. Фамилия,
15. Имя,
16. Отчество,
17. Причастие,
18. Союз,

19. Наречие,
20. Частица,
21. Междометие,
22. Топоним,
23. Субстантивированное прилагательное.

Каждой морфологической категории сопоставляется морфологический класс, описываемый вектором окончаний. Все векторы окончаний данной морфологической категории состоят из одинакового числа элементов, равного числу словоформ данной морфологической категории. Внутри данного морфологического класса слова могут отличаться друг от друга чередованиями. Эта информация хранится в морфологической базе данных в виде отдельных таблиц (таблица окончаний и таблица чередований).

Далее перечислены морфологические классы для соответствующих морфологических категорий.

Существительные, Фамилии, Имена, Отчества, Предлоги:

Определено 12 основных словоформ:

0. Именительный падеж, единственное число;
1. Родительный падеж, единственное число;
2. Дательный падеж, единственное число;
3. Винительный падеж, единственное число;
4. Творительный падеж, единственное число;
5. Предложный падеж, единственное число;
6. Именительный падеж, множественное число;
7. Родительный падеж, множественное число;
8. Дательный падеж, множественное число;
9. Винительный падеж, множественное число;

10. Творительный падеж, множественное число;
11. Предложный падеж, множественное число.

Предлоги не изменяют свою форму в зависимости от падежа. Отнесение их к изменяемым словам в данной модели связано с тем, что каждый предлог употребляется с существительным, стоящим только в определенных падежах.

Местоимения, количественные и собирательные числительные:

Определено 6 основных словоформ:

0. Именительный падеж;
1. Родительный падеж;
2. Дательный падеж;
3. Винительный падеж;
4. Творительный падеж;
5. Предложный падеж;

Прилагательные, порядковые числительные, причастия и местоименные прилагательные:

Определено 18 основных словоформ и одна дополнительная словоформа (знаком «+» помечены неразличимые формы, объединяемые в одну).

Полные формы, единственное число:

0. Именительный падеж, мужской род + винительный падеж, неодушевленный мужской род;
1. Именительный падеж, средний род + винительный падеж, средний род;
2. Родительный падеж, мужской и средний род + винительный падеж, одушевленный мужской род;
3. Дательный падеж, мужской и средний род;

4. Творительный падеж, мужской и средний род;
5. Предложный падеж, мужской и средний род;
6. Именительный падеж, женский род;
7. Родительный падеж, женский род + предложный падеж, женский род + дательный падеж, женский род + творительный падеж, женский род;
8. Винительный падеж, женский род.

Полные формы, множественное число:

9. Именительный падеж + винительный неодушевленный;
10. Винительный падеж одушевленный + предложный падеж + родительный падеж;
11. Дательный падеж;
12. Творительный падеж.

Краткие формы, единственное число:

13. Именительный падеж, мужской род + винительный падеж, неодушевленный мужской род;
14. Именительный падеж, женский род + родительный падеж, мужской и средний род;
15. Именительный падеж, средний род + винительный падеж, средний род, неодушевленный.

Краткие формы, множественное число:

16. Именительный падеж, множественное число;
17. Сравнительная степень.

Дополнительные падежи:

18. Второй творительный падеж, женский род.

Глаголы несовершенного вида:

У глаголов несовершенного вида рассматривается 16 словоформ:

0. Инфинитив.

Формы настоящего времени:

1. 1-е лицо, единственное число;
2. 2-е лицо, единственное число;
3. 3-е лицо, единственное число;
4. 1-е лицо, множественное число;
5. 2-е лицо, множественное число;
6. 3-е лицо, множественное число;
7. Деепричастие настоящего времени.

Формы прошедшего времени:

8. Мужской род, единственное число;
9. Женский род, единственное число;
10. Средний род, единственное число;
11. Множественное число;
12. 1-е деепричастие прошедшего времени;
13. 2-е деепричастие прошедшего времени.

Повелительное наклонение:

14. Единственное число;
15. Множественное число.

Глаголы совершенного вида:

У глаголов совершенного вида рассматривается 15 словоформ:

0. Инфинитив.

Личные формы:

1. 1-е лицо, единственное число;

2. 2-е лицо, единственное число;
3. 3-е лицо, единственное число;
4. 1-е лицо, множественное число;
5. 2-е лицо, множественное число;
6. 3-е лицо, множественное число.

Формы прошедшего времени:

7. Мужской род, единственное число;
8. Женский род, единственное число;
9. Средний род, единственное число;
10. Множественное число;
11. 1-е деепричастие прошедшего времени;
12. 2-е деепричастие прошедшего времени.

Повелительное наклонение:

13. Единственное число;
14. Множественное число.

Каждая морфологическая категория имеет дополнительные признаки, которые могут быть реализованы в морфологическом словаре в виде отдельных флагов.

Перечислим возможные признаки.

Общие признаки:

1. Является термином/названием (именем собственным);
2. Обязательное наличие точки в конце слова;
3. Является составной частью слова, содержащего дефис;
4. Не используется как самостоятельное слово.

Существительные, фамилии, имена, отчества:

1. Может употребляться как слово мужского рода;
2. Может употребляться как слово женского рода;
3. Может употребляться как слово среднего рода;
4. Имеет второй родительный падеж;
5. Имеет второй предложный падеж;
6. Одушевленное.

Количественные числительные:

1. Может использоваться с неодушевленными существительными;
2. Может использоваться с одушевленными существительными.

Прилагательные, порядковые числительные, местоименные прилагательные:

1. Может использоваться как неодушевленное существительное;
2. Может использоваться как одушевленное существительное.

Местоимения:

1. Может употребляться как слово мужского рода;
2. Может употребляться как слово женского рода;
3. Может употребляться как слово среднего рода;
4. Имеет множественное число;
5. Одушевленное;
6. Может иметь предлог.

Глаголы и причастия:

1. Переходный;
2. Способен согласовываться с прилагательными в творительным падеже;
3. Имеет беглое «о» в приставке (для причастий — причастие образовано от глагола, имеющего беглое «о»).

1.5.2. Алгоритм морфологического анализа слов русского языка

Поиск поданной на вход словоформы осуществляется по бору словоформ за линейное время от длины словоформы (в символах алфавита). Если искомая словоформа существует в словаре, то для нее будет получена вся морфологическая информация, а также указатели на словоформу, представляющую парадигму слова, и на следующую словоформу в парадигме. В противном случае будет сообщено, что искомой словоформы нет в морфологическом словаре.

В разработанной морфологической модели для сокращения вариантов поиска были разработаны алгоритмы учета словообразования в русском языке. Рассматриваются 2 типа словообразования. Для каждого типа заданы ограничения на части речи, в рамках которых осуществляется словообразование.

В русском языке от имени существительного с помощью префикса «пол» может быть получено новое слово, правильное с точки зрения грамматики. Например: пол-лимона, полкарты. При этом, как видно из приведенного примера, «пол» может писаться слитно или через дефис с последующим словом. При поиске слова в словаре производится отбрасывание префикса вместе с дефисом, если он есть. Полученное слово ищется стандартным алгоритмом в словаре.

Найденный вариант проверяется на соответствие следующим условиям:

- Если найдено имя существительное, то оно должно стоять в первой форме (именительный падеж, единственное число);
- Если найдено прилагательное, то оно должно стоять во второй (родительный падеж, мужской и средний род или винительный падеж, одушевленный мужской род) или седьмой форме (родительный падеж,

женский род, или предложный падеж, женский род, или дательный падеж, женский род, или творительный падеж, женский род);

- Если слово начинается с гласной буквы или с буквы «л», то после префикса «пол» должен стоять дефис.

Если ни одно из перечисленных условий не было выполнено, то рассматриваемый вариант отбрасывается.

В русском языке от любого имени прилагательного с помощью префикса «по» может быть получено новое слово, правильное с точки зрения грамматики. Например: посильнее, получше. Префикс «по» пишется слитно только с семнадцатой формой имени прилагательного (сравнительная степень).

1.6. Компьютерная словарная морфология английского языка

1.6.1. Морфологическая модель

В английском языке формы словообразования и словоизменения очень часто не являются признаками, отличающими одну часть речи от другой, поскольку большее количество английских слов не имеет характерных суффиксов, указывающих на их принадлежность к той или иной части речи, а количество грамматических окончаний крайне ограничено.

В данной модели будем различать следующие части речи:

1. Имя существительное (*noun*).
2. Имя прилагательное (*adjective*).
3. Имя числительное (*numeral).*
4. Местоимение (*pronoun*).
5. Глагол (*verb*).
6. Наречие (*adverb*).
7. Предлог (*preposition*).

8. Союз (*conjunction)*.
9. Междометие (*interjection)*.

Проанализировав грамматические категории английского языка, необходимо выделить информацию, присущую каждой из категорий, и определить, какая информация должна присутствовать в морфологическом словаре.

Имя существительное

Существительные в английском языке имеют два числа — множественное и единственное: *a table* (ед.ч.), *tables* (мн.ч.); имеют два падежа – «общий» и «притяжательный»: *worker* (общий падеж), *worker's* (притяжательный падеж). Поскольку род имен существительных в английском языке определяется не формой слова, а значением, то данная характеристика не учитывается на этапе морфологического анализа. Имена существительные делятся на собственные и нарицательные, на исчисляемые и неисчисляемые.

Множественное число имени существительного образуется по следующим правилам:

1. Образование множественного числа с помощью окончания –**s** (*hand – hands*).
2. Существительные, заканчивающиеся на **-s, -ss, -x, -sh, -ch,** образуют множественное число с помощью окончания -**es** (*dish – dishes*).
3. Слова, заканчивающиеся на **-y** с предшествующей согласной, образуют множественное число с помощью окончания -**es,** при этом **y** заменяется на **i** (*city – cities*).
4. Имена существительные, заканчивающиеся на **-o**, множественное число образуют путем прибавления **-es** (*photo, piano* – исключения).
5. Имена существительные, заканчивающиеся на **-f**, образуют множественное число путем замены **f** на **v** и прибавления **-es.** Существительные на **-fe** образуют множественное число путем замены **f**

на **v** и прибавления **-s.** (Это не относится к словам *chief, handkerchief, roof, safe, wharf* (2 варианта – *wharfs, wharves*)).

6. Существуют особые случаи образования множественного числа (*man – men, datum – data*).
7. Некоторые существительные используются в одной и той же форме при обозначении и множественного, и единственного числа (*sheep, fish(es), dozen(s), score(s), works, mean*s).

Существительные в английском языке могут стоять в общем падеже (*the common case*) и притяжательном падеже (*the possessive case*). Притяжательный падеж образуется следующим образом:

- В единственном числе образуется с помощью окончания -**'s** (*student's*).
- Во множественном числе с помощью только апострофа **'** (*students'*).
- Кроме существительных одушевленных форму притяжательного падежа принимают: существительные, обозначающие время и расстояние (*at a kilometer's distance*), страны, города, суда, слова *world, country, city, ship*, некоторые наречия времени (*to-days's newspaper*).
- В застывших выражениях: *for order's sake, at a stone's throw* и т.д.

Имя существительное имеет четыре формы:

1. Общий падеж в единственном числе;
2. Притяжательный падеж в единственном числе;
3. Общий падеж во множественном числе;
4. Притяжательный падеж во множественном числе.

Для существительных определены следующие флаги:

1. Образование множественного числа при помощи окончания **-s**.
2. Образование множественного числа при помощи окончания **-es.**
3. Образование множественного числа при помощи изменения последней буквы + окончание **-es**.
4. Используется в единственном числе.

5. Используется во множественном числе.
6. Не меняет форму при образовании множественного числа.
7. Используется в притяжательном падеже.
8. Нарицательное имя.
9. Собственное имя.
10. Топоним.
11. Сокращение.

Имя прилагательное

Имена прилагательные в английском языке не изменяются ни по родам, ни по числам, ни по падежам, а изменяются только по степеням сравнения: *long, long**er**, long**est.***

Степени сравнения: сравнительная (*the comparative degree*), превосходная (*the superlative degree*). Основная форма – положительная степень (*the positive degree*).

Правила образования степеней сравнения:

- Односложные и некоторые двусложные прилагательные образуют сравнительную степень путем прибавления суффикса **-er** и превосходную – путем прибавления суффикса **-est.**
- Большинство двусложных, трехсложных и состоящих из большего количества слогов прилагательных образуют сравнительную степень при помощи слова **more**, превосходную – **most**. Например, *active, more active, most active*.
- Если прилагательное заканчивается на немое **-e,** то при прибавлении **-er** и **-est** оно опускается.
- Если прилагательное заканчивается на согласную с предшествующим кратким гласным звуком, то в сравнительной и превосходной степени согласная удваивается (*big – bigger, thin – thinner*).

- Если прилагательное оканчивается на **-y** с предшествующей согласной, то в сравнительной и превосходной степенях **y** переходит в **i** (*busy – busie –- busiest*).

У прилагательных три формы:

1. Положительная степень
2. Сравнительная степень
3. Превосходная степень

Для прилагательных определены следующие флаги:

1. Образует сравнительные степени при помощи суффиксов **-er/-est.**
2. Образует сравнительные степени при помощи слов **more/most.**
3. При образовании сравнительной степени при помощи суффиксов **–er** / **-est** производится замена **y** на **i.**
4. При образовании сравнительной степени при помощи суффиксов **– er** / **-est** происходит удвоение последней согласной.

Имя числительное

Имена числительные делятся на количественные (*Cardinal numerals*) и порядковые (*Ordinal numerals*). Порядковые числительные образуются от количественных с помощью суффикса **-th,** кроме *first, second, third*.

У числительных есть только одна форма. Для числительных определены следующие флаги:

1. Количественное числительное.
2. Порядковое числительное.

Местоимение

Местоимения в английском языке могут иметь формы для единственного и множественного числа (this – these), формы общего падежа и притяжательного падежа (*somebody, somebody's*), объектного и именительного падежа (I – me). Личные, притяжательные и возвратные местоимения обладают категориями числа и рода. Местоимения могут использоваться как в качестве

существительных (местоимения-существительные), так и прилагательных (местоимения-прилагательные).

Местоимения делятся на:

- личные местоимения (personal pronouns),
- притяжательные местоимения (possessive pronouns),
- возвратные местоимения (reflexive pronouns),
- взаимные местоимения (reciprocal pronouns),
- указательные местоимения (demonstrative pronouns),
- вопросительные местоимения (interrogative pronouns),
- относительные местоимения (relative pronouns, conjunctive pronouns),
- неопределенные местоимения (indefinite pronouns).

У местоимений четыре формы:

1. именительный падеж
2. объектный падеж
3. притяжательный падеж
4. множественное число

Для местоимений определены следующие флаги:

1. Личное местоимение.
2. Притяжательное местоимение.
3. Возвратное местоимение.
4. Указательное местоимение.
5. Вопросительное местоимение.
6. Относительное местоимение.
7. Неопределенное местоимение.
8. Местоимение 1-го лица единственного числа.
9. Местоимение 2-го лица единственного числа.
10. Местоимение 3-его лица единственного числа мужского рода.
11. Местоимение 3-его лица единственного числа женского рода.

12. Местоимение 3-его лица единственного числа среднего рода.
13. Местоимение 1-го лица множественного числа.
14. Местоимение 2-го лица множественного числа.
15. Местоимение 3-го лица множественного числа.
16. Местоимение-существительное.
17. Местоимение-прилагательное.
18. Используется в притяжательном падеже

Флаги 8-15 присущи только личным, притяжательным и возвратным местоимениям.

Глагол

Английский глагол имеет три основные формы: первая форма – инфинитив (Infinitive), вторая форма – прошедшее простое время (Past Simple) и третья форма – причастие прошедшего времени (Past Participle). По способу образования прошедшего времени и причастия прошедшего времени глаголы делят на две группы: правильные и неправильные.

Правила образования Past Simple и Past Participle у правильных глаголов:

- Образуются путем прибавления к инфинитиву окончания **-ed**
- Глаголы, оканчивающиеся в инфинитиве на немое **-e,** теряют эту гласную при добавлении **-ed** (*live – lived*)
- Глаголы, оканчивающиеся в инфинитиве на **-y** с предшествующей согласной, меняют **y** на **i** (*cry – cried*)
- Если последний ударный слог закрытый, то необходимо удвоить последнюю гласную (*stop – stopped, permit – permitted*)
- Если глагол заканчивается на **l**, то **l** удваивается вне зависимости от ударения (*travel – travelled*), American – удваивание происходит, если падает ударение (*travel – traveled*)

Глаголы делятся на: смысловые (to speak), вспомогательные (to be, to have, to do, shall (should), will(would)), глаголы связки (to be, to become, to grow, to get, to turn, to look и др.), модальные глаголы (can, may, must, ought, need).

В третьем лице единственного числа настоящего времени глагол принимает окончание -s:

- Глаголы на **-ss, -ch, -sh, -x** принимают окончание **-es.**
- Глаголы, оканчивающиеся на **-y** с предшествующей согласной, меняют **y** на **i.**
- Некоторые глаголы на **-o** (*to do, to go*) принимают окончание **-es,** (*does, goes*).

Продолженные времена (группы Continuous) образуются при помощи причастия настоящего времени, которое образуется путем прибавления окончания -ing к инфинитиву глагола.

- Если глагол заканчивается на немое **-e,** то при прибавлении **-ing** немое **-e** опускается: *to mak**e** – mak**ing***.
- Если последний ударный слог закрытый, то необходимо удвоить последнюю гласную (*si**t** – si**tt**ing, permi**t** – permi**tt**ing*).
- Если глагол заканчивается на **-l**, то **l** удваивается вне зависимости от ударения (*trave**l** – trave**ll**ing*), American E. – удваивание происходит, если ударение падает на последний слог (*trave**l** – trave**l**ing*).

Обычные глаголы имеют пять форм:

1. Инфинитив
2. Простое прошедшее время
3. Причастие прошедшего времени
4. Причастие настоящего времени
5. Третье лицо единственного числа настоящего времени

У модальных глаголов только две формы:

1. Настоящее время

2. Прошедшее время

Для обычных глаголов определены следующие флаги:

1. Образование формы 3-го лица единственного числа настоящего времени при помощи окончания **-s.**
2. Образование формы 3-го лица единственного числа настоящего времени при помощи окончания **-es.**
3. Изменение последней буквы при образовании причастий, форм прошедшего времени и 3-го лица единственного числа настоящего времени.
4. Удвоение последней согласной при образовании причастий и формы прошедшего времени.
5. Правильный глагол.
6. Глагол-связка.
7. Вспомогательный глагол.

Для модальных глаголов определен следующий флаг:

- используется как вспомогательный глагол

Наречие

Английские наречия делятся на: наречия места (here, there), наречия времени (now), наречия меры и степени (much, little), наречия образа действия (nearly, almost).

Образуют степени сравнения, подобно прилагательным. Односложные наречия и наречие early образуют сравнительную степень путем прибавления суффикса -er, а превосходную — -est.

У наречий есть три формы:

1. Положительная степень
2. Сравнительная степень
3. Превосходная степень

Для наречий определены следующие флаги:

1. Образует сравнительные степени при помощи суффиксов **-er/-est.**
2. Образует сравнительные степени при помощи слов **more/most.**
3. При образовании сравнительной степени при помощи суффиксов **–er** / **-est** производится замена **y** на **i.**
4. При образовании сравнительной степени при помощи суффиксов **–er** / **-est** происходит удвоение последней согласной.

Предлог, союз, междометие

Данные части речи являются неизменяемыми, союзы делятся на подчинительные и сочинительные.

Для союзов определены следующие флаги:

1. Подчинительный союз.
2. Сочинительный союз.

Артикль

Артикль не является частью речи. Существуют два типа артиклей: неопределённый (a, an) и определённый (the). Артикли не изменяются. Для артиклей определены следующие флаги:

1. Неопределенный артикль.
2. Определенный артикль.

В используемой морфологической модели было расширено понятие части речи, поскольку необходимо было добавить еще три категории, не являющиеся частями речи: артикли, модальные глаголы, сокращения. Артикли являются служебными словами; отдельно выделены модальные глаголы, поскольку они отличаются от смысловых глаголов по своим функциям и характеристикам и образуют не все глагольные формы. В категорию сокращений попадают все краткие формы вспомогательных и модальных глаголов, сокращения вида: *‘m, ‘ll, ‘d* и т.д., также краткие отрицательные формы: *hadn't, hasn’t*.

Грамматические категории

Номера категорий соответствуют значениям, используемым при формировании битовых флагов:

1. Существительное (*Noun*).
2. Глагол (*Verb*).
3. Модальный глагол (*Modal verb*).
4. Прилагательное (*Adjective)*.
5. Наречие (*Adverb)*.
6. Числительное (*Numeral)*.
7. Местоимение (*Pronoun)*.
8. Предлог (*Preposition*).
9. Союз (*Conjunction*).
10. Артикль (*Article*).
11. Сокращение (*Shortening*).

Для каждой грамматической категории были выделены характеристики данной части речи и характеристики, относящиеся к образованию словоформ.

Номера характеристик соответствуют номерам бит в битовых флагах, каждая из характеристик может принимать значение 0 – отсутствует или 1 – присутствует у конкретного слова. Номера словоформ также соответствуют номерам бит в битовых флагах.

1.6.2. Алгоритмы морфологического анализа

В модуль морфологического анализа включаются функции, позволяющие осуществлять морфологический разбор и синтез словоформ.

Поскольку в словаре хранятся только слова, без векторов окончаний, то возможные окончания для каждой части речи находятся непосредственно в виде массива в программном модуле морфологического разбора; также

существует массив, содержащий флаги, соответствующие каждому окончанию. Первоначально осуществляется поиск слова в словаре исключений, далее производится попытка определить базовую форму слова по его окончанию, наличие которого также проверяется в словаре. Отделяемые окончания для каждой изменяемой части речи представлены в таблице 1.3.

Входной информацией для алгоритма синтеза является представляющая словоформа слова, часть речи и битовый флаг, указывающий на запрашиваемую словоформу; выходная информация содержит символьное представление запрашиваемой формы, либо — при отсутствии флага — все возможные формы данного слова. Найденные словоформы добавляются в массив, каждый элемент которого – это структура, состоящая из строки и битового флага.

Таблица 1.3. Отделяемые окончания для изменяемых частей речи

Часть речи	**Отделяемое окончание**	**Прибавляемое окончание**	**Словоформа (№)**
Существительное	*’*	-	3
	’s	-	2
	s	-	1
	es	-	1
	ies	*y*	1
	ves	*f*	1
	ves	*fe*	1
Глагол	*s*	-	4
	es	-	4
	ies	-	4
	ed	-	1,2
	ed	*e*	1,2
	ied	*y*	1,2
	ing	-	3
	ing	*e*	3
	ying	*ie*	3
Прилагательное, наречие	*er*	-	1
	er	*e*	1
	est	-	2
	est	*e*	2
	ier	*y*	1
	iest	*y*	2
	ier	*ey*	1
	iest	*ey*	2

Алгоритм поиска всех возможных словоформ

1. Поиск слова в основном словаре, в случае неудачи переход к п.5.
2. Извлечение битовых флагов слова
3. Вызов процедуры синтеза словоформ из флагов, синтезированные словоформы помещаются в результирующий массив.
4. Если установлен флаг существования словоформ, образующихся не по правилам, то извлечь информацию из области хранения исключений и добавить к результирующему массиву.
5. Выход из процедуры поиска словоформ.

В случае поиска одной словоформы алгоритм аналогичен, только вместо всех словоформ в п.3. и п.4. извлекается только искомая словоформа.

Для определения вида окончания множественного числа используется следующий алгоритм:

Алгоритм определения типа окончания

1. Если слово является исключением, то алгоритм завершается.
2. Если слово оканчивается на *o, s, x, sh, ch*, то окончание во множественном числе -es, установить флаг окончания -es.
3. Если слово заканчивается на f, fe (кроме слов chief, handkerchief, roof, safe), то окончание во множественном числе –ves, установить флаг изменения последней буквы.
4. Если слово заканчивается на y и предшествующая буква – согласная, то окончание во множественном числе –ies, установить флаг изменения последней буквы.
5. Если не выполнился ни один из пунктов 2 – 4, то окончание во множественном числе -s, установить флаг окончания -s.

Определение окончания у глаголов в третьем лице единственного числа настоящего времени осуществляется аналогично, кроме пункта 3, который отсутствует.

Если глагол заканчивается на *-ie*, то при образовании причастия настоящего времени *-ie* заменяется на *-y*, необходимо установить флаг изменения последней буквы.

Алгоритм определения односложных и двусложных слов

Для определения того, образует ли прилагательное/наречие степени сравнения при помощи суффиксов *-er/-est,* необходимо определить количество слогов в слове.

Пусть *word* – рассматриваемое слово, массив символов от 1 до *len*, где *len* – длина слова.

1. Подсчитать число гласных, разделенных согласными, записать в nVowels, т.е. для *airy nVowels* = 2, для *deep – nVowels* = 1.
2. Если *nVowels*>3, то переход к п.5.
3. Если *nVowels* = 3 или *nVowels* = 2 переход к п.4. иначе — п.5
4. Если (*word[len] = ‘e’*) и (*word[len-1]* – согласная) и (*word[len-2]* – гласная), то *nVowels = nVowels -1*.
5. Вернуть *nVowels*.

Для определения форм сравнения прилагательных используем правило:

Если число слогов = 1 или (число слогов =2 и слово заканчивается на *-y, -er, -ow, -ble*), то устанавливаем флаги образования степеней сравнения при помощи суффиксов *-er/-est.*

Для определения форм сравнения наречий используем правило: если число слогов = 1 или (слово = *“early”, ”easy”*), то устанавливаем флаги образования степеней сравнения при помощи суффиксов *-er/-est*; если слово =

“often”, ”quickly”, “slowly”, то устанавливаются флаги образования степеней сравнения при помощи суффиксов *-er/-est* и слов *more* и *most*; если наречие заканчивается на *-ly*, то устанавливается флаг образования степеней сравнения при помощи слов *more* и *most*.

1.7. Поддержка естественных языков в поисковых системах

В таблице 1.4 приведен список языков, поддерживаемых наиболее распространенными на территории Российской Федерации поисковыми системами. Знаком «+» отмечены языки, поддерживаемые данной информационно-поисковой системой. Под поддержкой языка мы будем понимать наличие в заданной информационно-поисковой системе возможностей по автоматическому определению языка при анализе документа, а также наличие какой-либо схемы анализа словоформ указываемого языка.

Таблица 1.4. Поддержка языков в поисковых системах.

№	Язык	Поисковая система				
		Яндекс	**Google**	**Rambler**	**Bing**	**Apache Lucene**
1	Азербайджанский	+				
2	Албанский	+	+			
3	Английский	+	+	+	+	+
4	Арабский	+	+		+	+
5	Армянский	+				+
6	Африкаанс		+			
7	Башкирский	+				
8	Белорусский	+	+	+		
9	Болгарский	+	+			+
10	Валлийский		+			
11	Венгерский	+	+		+	+
12	Вьетнамский		+		+	
13	Гаитянский креольский		+		+	
14	Галисийский		+			+
15	Греческий	+	+		+	+
16	Грузинский	+				
17	Датский	+	+		+	+
18	Иврит	+	+		+	+

19	Идиш		+			
20	Индонезийский		+		+	+
21	Ирландский		+			
22	Исландский		+			
23	Испанский	+	+		+	+
24	Итальянский	+	+		+	+
25	Казахский	+		+		
26	Каталанский	+	+			+
27	Китайский		+		+	+
28	Корейский		+		+	+
29	Латышский	+	+		+	
30	Литовский	+	+		+	
31	Македонский	+	+			
32	Малайский		+			
33	Мальтийский		+			
34	Немецкий	+	+	+	+	+
35	Нидерландский	+	+		+	+
36	Норвежский	+	+		+	+
37	Персидский	+	+			+
38	Польский	+	+		+	+
39	Португальский	+	+		+	+
40	Румынский	+	+		+	+
41	Русский	+	+	+	+	+
42	Сербскохорватский	+	+			
43	Словацкий	+	+		+	
44	Словенский	+	+		+	
45	Суахили		+			
46	Тайский		+		+	
47	Татарский	+		+		
48	Турецкий	+	+		+	+
49	Украинский	+	+	+	+	
50	Филиппинский		+			
51	Финский	+	+		+	+
52	Французский	+	+	+	+	+
53	Хинди		+			+
54	Чешский	+	+		+	+
55	Шведский	+	+		+	+
56	Эсперанто		+			
57	Эстонский	+	+		+	
58	Японский		+		+	+

Для языков, поддерживаемых в информационно-поисковой системе «Яндекс», как указано в официальной документации[1], реализована поддержка на уровне распознавания и индексации. Под индексацией подразумевается, в том числе, применение алгоритмов автоматического морфологического анализа к словоформам на указанных языках. Морфологический анализ в рассматриваемой системе устроен следующим образом. Для каждого языка на основе единого подхода строятся машинные морфологические словари, на основе либо открытых данных, либо купленных к какой-либо организации. Анализ отсутствующих в словаре слов работает на основе метода аналогии, дополняемого специальным словарем — «быстрословарем».

Поисковая система «Google» в 2003 году на собственной официальной странице опубликовала информацию о том, что для анализа словоформ они используют технологию автоматического выделения основы[2]. Позднее, в 2006 году, в блоге русскоязычного подразделения «Google»[3] была опубликована информация о том, что, в частности, для русского языка ими стала использоваться словарная морфология. При этом не отмечалось, на основе каких данных были построены машинные словари. Также в данной публикации «Google» было заявлено, что использование технологий автоматического морфологического анализа на основе словаря не повышает качество поиска, а в ряде случаев даже приносит негативный эффект. Оставим вопрос о соответствии этого заявления действительности на совести информантов. Можно предположить, что речь идет о поиске в некотором узком смысле его понимания. Приведенные в таблице языки поддержаны «Google» на уровне технологий автоматического определения языка документов[4]. Надо полагать, что для них также существуют механизмы анализа словоформ. При этом для

[1] URL: http://help.yandex.ru/webmaster/?id=1111517
[2] URL: http://google.blogspace.com/archives/001092.html
[3] URL: http://googlerussiablog.blogspot.ru/2006/09/blog-post_18.html
[4] URL: https://developers.google.com/translate/v2/using_rest#detect-language

подавляющего большинства языков это алгоритмы аналитического выделения основы.

На официальной странице «Mail.ru» заявлено, что их поисковая система «Поиск@Mail.Ru» в настоящий момент поддерживает только русский язык[5]. Это обусловлено тем, что «Mail.ru» покупает выдачу у компании «Google», дополняя ее в ряде случаев результатами работы собственной информационно-поисковой системы. По информации, предоставляемой на сайте проекта АОТ, с 2004 года их морфологический модуль используется в поисковой системе, разрабатываемой компанией «Mail.ru»[6]. В ноябре 2012 года появилась информация о том, что компания «Mail.ru» в ближайшее время собирается отказаться от использования поисковых технологий компании «Google» в пользу собственной информационно-поисковой системы[7]. При этом в ряде источников указывалось, что с лета 2012 года в недрах компании была начата работа над собственной системой автоматического морфологического анализа. На каких принципах она будет создаваться не сообщалось. Однако стоит полагать, что она будет носить словарный характер, так как за годы использования словарной морфологии из проекта АОТ имеющиеся там словари были в значительной степени расширены и дополнены.

В поисковой системе «Rambler» для автоматического морфологического анализа словоформ, по крайней мере, русского и английского языков, используется модуль, построенный на основе разработок И.С. Ашманова, который до 2001 года являлся исполнительным директором компании. Таким образом, в основу системы легли разработки, описанные И.С. А шмановым в своей кандидатской диссертации [3]. Список поддерживаемых языков, отмеченных в таблице, приведен на странице официальной документации[8]. С 2011 года «Рамблер» стал показывать на выдаче результаты, предоставляемые

[5] URL: http://help.mail.ru/webmaster/indexing/robots/lang_code
[6] URL: http://aot.ru/
[7] URL: http://www.rg.ru/2012/11/26/mail-site.html
[8] URL: http://help.rambler.ru/rsearch/rsearch-rasshirennyj-poisk/1315/

компанией «Яндекс»[9], и, таким образом, перестал использовать собственные разработки в области информационного поиска и анализа документов, в том числе морфологии.

Поисковая система «Bing» от «Microsoft» поддерживает список языков, которые отмечены в таблице 1.4. Приведенные данные взяты из ресурса[10]. Отмечается поддержка на уровне автоматического определения языка документа. В патенте корпорации «Microsoft» указано[11], что при анализе текста информационно-поисковая система «Bing» применяет технологию аналитического выделения основы.

В проекте Apache Lucene для ряда языков реализовано несколько алгоритмов аналитического выделения основы[12]. Разработчикам предоставлено множество разнообразных схем анализа, обладающих разными показателями качества и производительности. Для некоторых языков существует возможность осуществлять полный морфологический анализ с использованием специализированного словаря. Отметим, что в данном программном продукте нет механизмов по разпознаванию языков и кодировок.

[9] URL: http://www.rg.ru/2011/06/23/poisk-site-anons.html
[10] URL: http://www.emreakkas.com/internationalization/microsoft-translator-api-languages-list-language-codes-and-names
[11] URL: http://www.google.com/patents/US7814108
[12] URL: http://wiki.apache.org/solr/LanguageAnalysis

2. АНАЛИТИЧЕСКИЕ АЛГОРИТМЫ МОРФОЛОГИЧЕСКОГО АНАЛИЗА

Требования к усвоению данного раздела учебного пособия:

Знать: основы информационной культуры, назначение и квалификацию программных средств цифровой обработки информации, принципы и методы их использования в издательском деле;

Уметь: разрабатывать предложения по организации информационного пространства с использованием современных технологий;

Владеть навыками: использования программного обеспечения в процессе подготовки печатных и электронных издани..

2.1. Развитие аналитических методов выделения основ

В настоящее время аналитические методы выделения основы являются одной из основных компонент информационно-поисковых систем как общего, так и специального назначения. Они используются для осуществления индексации и последующего поиска информации в информационно-поисковых системах [82, 97], позволяя повысить полноту поиска и оптимизировать размер индекса информационно-поисковой системы. Кроме этого, алгоритмы аналитического выделения основы находят широкое применение в задачах классификации [85, 119], автоматического реферирования [71, 105] и других задачах автоматической обработки текстовой информации.

Аналитическим алгоритмом выделения основы называется процесс приведения словоформы, полученной от канонической формы слова в результате словоизменения или словообразования, к псевдооснове. Важно отметить, что получаемая псевдооснова не обязана совпадать с грамматической основой рассматриваемой словоформы, но при этом достаточно, чтобы словоформы, соответствующие одной парадигме, получили в результате работы алгоритма одну и ту же псевдооснову.

Можно определить несколько типов алгоритмов аналитического выделения основы, различающихся по скорости обработки текстовой

информации и качеству получаемых результатов. Все их можно разделить на три больших класса:

- Алгоритмы, основанные на отсечении аффиксов;
- Статистические алгоритмы;
- Смешанные алгоритмы.

2.1.1. Алгоритмы, основанные на отсечении аффиксов

Тривиальный алгоритм аналитического выделения основы может быть построен в виде процедуры, в результате работы которой в качестве псевдоосновы выбирается подстрока, состоящая из первых $\boldsymbol{n}$ символов анализируемой словоформы. Если словоформа имеет длину менее n символов, то она целиком выбирается в качестве псевдоосновы. В англоязычной литературе такого рода алгоритмы носят название «$\boldsymbol{Truncate(n)}$», где параметр $\boldsymbol{n}$ указывает длину выделяемой псевдоосновы (например, $\boldsymbol{Truncate(4), Truncate(5)}$).

Схожий подход предлагается в работе Д. Харман (Harman) [88] и называется «*S-stemmer*». Рассматриваемый алгоритм аналитического выделения основы учитывает один только сценарий словоизменения — образования множественного числа имен существительных в английском языке. Таким образом, подаваемая на вход словоформа преобразуется к форме единственного числа, если это возможно, и результат возвращается в качестве псевдоосновы.

Исторически первый алгоритм аналитического выделения основы, который учитывал в максимальной степени словоизменительные аспекты языка, был предложен Дж.Б. Ловинс (*Lovins*) в 1968 году в работе [98] и предназначался только для английского языка. В своей работе автор определяет 294 окончания, каждое из которых может быть удалено при выполнении одного из 29 условий. Кроме того, определяются 35 правил трансформации словоформ после отсечения окончаний, которые упорядочиваются по принципу

наибольшего совпадения. Мы будем именовать отбрасываемую часть на конце анализируемой словоформы псевдофлексией; если же отбрасывание происходит от начала словоформы, то эту часть мы будем называть псевдопрефиксом. Приставка «псевдо» указывает на то, что данная часть словоформы может быть представлена несколькими аффиксами. Рассматриваемый алгоритм выбирает наиболее длинную псевдофлексию, которую можно отсечь при выполнении условий и ограничений. Рассмотрим работу алгоритма на примере английского слова «*nationally*». Существует два суффикса, которые могут быть удалены для получения основы: «*ationally*» и «*ionally*». Первый не может быть удален, потому что существует ограничение: «выделенная основа должна быть длиннее 3-х символов». Таким образом, мы получаем основу «*nat*».

Данный алгоритм является самым быстрым алгоритмом выделения псевдоосновы и при этом может обрабатывать случаи удвоения согласных в корне анализируемой словоформы. Его недостатком является необходимость кропотливой работы лингвистов по составлению списков правил и исключений, что представляет собой тяжелую задачу для языков с развитой морфологического структурой, в частности, для языков флективного морфологического типа.

В развитие идей, предложенных в работе Ловинс (*Lovins*), в 1974 году Дж. Доусен (*Dawson*) публикует статью [74], в которой излагается новый алгоритм аналитического анализа словоформ, впоследствии получивший название «алгоритм Доусена». Это однопроходный алгоритм удаления псевдофлексий. Основная идея заключается в том, чтобы взять алгоритм Ловинс (*Lovins*) и, дополнив его новыми правилами, исправить ошибки и неточности, которые были выявлены после исследований качества алгоритма Ловинс (*Lovins*). В результате список всех суффиксов был расширен до примерно 1200 наименований. Псевдофлексии хранятся в таблице в обратном порядке и индексируются своей длиной и последней буквой. Алгоритм не получил

широкого распространения из-за своей сложности, большого количества ошибок и большего времени работы по сравнению с базовым алгоритмом Ловинс (*Lovins*).

В 1980 году М. Портер (*Porter*) опубликовал свой алгоритм аналитического выделения основы для английского языка [111, 112], который был принят научным сообществом за стандарт для алгоритмов аналитического выделения основы. Алгоритм представляет собой последовательность шагов, на каждом из которых, в зависимости от выполнения условий, может происходить одно из определенных преобразований окончаний, задаваемое правилом. Правила имеют вид: ⟨условие⟩⟨окончание⟩ → ⟨новое окончание⟩**.**

В качестве примера приведем одно из правил, предлагаемое в оригинальной статье: $(\boldsymbol{m > 0})\boldsymbol{EED} \rightarrow \boldsymbol{EE}$, которое означает, что если в словоформе есть хотя бы одна гласная буква и словоформа оканчивается на согласную и окончание $-\boldsymbol{eed,}$ то окончание заменяется на $-\boldsymbol{ee}$. Например, применение этого правила к словоформе «agreed» приводит ее к псевдооснове «***agree***».

Алгоритм состоит из примерно 60 правил, каждое из которых последовательно применяется к поступившей на вход словоформе. В результате получается быстрый алгоритм с точки зрения вычислительной сложности. Для уменьшения ошибок выделения основы для форм, образующихся с исключением, вводят специальную таблицу исключений (например, для неправильных глаголов). Также было показано [78], что алгоритм Портера (*Porter*) допускает меньше ошибок, чем алгоритм Ловинс (*Lovins*), а также, что алгоритм Ловинс (*Lovins)* является более «тяжелым» алгоритмом, то есть он отсекает большую часть анализируемой словоформы по сравнению с алгоритмом Портера *(Porter*).

В 1990 году К. Пейсом (*Paice)* и Г. Хаском (*Husk*) был предложен новый алгоритм аналитического выделения основы [106]. Алгоритм получил название

«Ланкастерский алгоритм», также в зарубежной литературе его можно встретить под именем «алгоритм Пейса-Хаска». Это смешанный итерационный алгоритм аналитического выделения основы.

В алгоритме используется таблица правил, каждое из которых задает преобразование псевдофлексии (удаление или замену). Замены используются для оптимизации алгоритма с целью сокращения числа дополнительных шагов, которые происходят уже после удаления псевдофлексий (например, замена удвоенной согласной в алгоритме Ловинс (*Lovins*)). Правила в таблице индексируются последними буквами соответствующих псевдофлексий. Каждое правило состоит из следующих частей:

- псевдофлексия, представляемая в инвертированном виде;
- опциональный флаг целостности «*»;
- число, указывающее длину удаляемой псевдофлексии (включая 0);
- опциональная добавляемая строка длиной 1 или более символов;
- управляющие символы («>» или «.»).

В 1993 году Р. Кровец (*Krovetz)* предложил свой алгоритм аналитического выделения основы [96]. Данный алгоритм обрабатывает наиболее распространенные случаи словоизменения в английском языке, в результате чего почти никогда не допускает ошибок. Рассматриваются следующие правила трансформации анализируемой словоформы:

- преобразование формы множественного числа в форму единственного числа (отсечение одного из суффиксов: $-\mathbf{ies}, -\mathbf{es}, -\mathbf{s})$**;**
- преобразование формы прошедшего времени в форму настоящего времени (удаление суффикса $-\mathbf{ed})$;
- удаление суффикса $-\mathbf{ing}$**.**

Сначала алгоритм удаляет один из возможных суффиксов и затем, проверяя полученный результат в словаре исключений, возвращает основу словоформы. Использование словаря позволяет анализировать словоформы с

грамматическими ошибками; кроме того, возвращаемый результат представляет собой реальное слово языка, а не псевдооснову, что позволяет четко интерпретировать полученные результаты.

Так как исторически первые исследования по разработке алгоритмов аналитического выделения основы осуществлялись для английского языка, то в конечном счете появились работы, в которых проводился сравнительный анализ разработанных алгоритмов [90].

В качестве альтернативы чисто аналитическим методам, требующим от авторов алгоритма знания языка, его морфологической структуры, выступают статистические методы аналитического выделения основы, которые в последнее время приобретают все большую популярность в различных задачах обработки текстовой информации. В ряде работ [99, 101, 102, 108] было показано, что статистические методы могут выступать в качестве альтернативы методам, построенным на правилах грамматики языка, обеспечивая приемлемые показатели качества и скорости обработки текстов на естественном языке.

2.1.2. Статистические алгоритмы выделения основ

Приведем примеры алгоритмов аналитического выделения основы, построенных на статистических методах анализа. Отметим, что не все из данных алгоритмов выделяют псевдооснову путем отсечения аффиксов.

В работе [86] исследуется принцип минимального описания (*MDL — Minimum Description Length*), который гласит, что морфологическая теория, которая описывает тот или иной корпус, должна быть минимальной по длине. Например, если в корпусе можно выделить $\boldsymbol{N}$ основ и $\boldsymbol{K}$ псевдофлексий, тогда такая теория лучше, чем теория, которая выделяет $\boldsymbol{N+1}$ основу или $\boldsymbol{K+1}$ псевдофлексию. Использование принципа минимального описания приведено в работе [87], где описывается алгоритм *Automorphology*, который позволяет

выделять, по сути, псевдофлексии, не используя предварительных сведений о языке, а имея только некоторый корпус. Построенный с помощью данного алгоритма список псевдофлексий используется для последующего анализа с целью выделения псевдоосновы. Схожий подход, определяющий взаимные частоты встречаемости псевдооснов и псевдофлексий в корпусе, был предложен в работе [104].

В работе Мейфилда (*Mayfield*) и МакНейми (*McNamee*) [101] предлагается «*n*-граммный» алгоритм аналитического выделения основы, который может быть применен к анализу текстов на произвольном языке. Основная идея алгоритма заключается в том, что близкие словоформы должны иметь большую долю совпадающих буквосочетаний. Предполагается анализ распределения всех буквосочетаний в рассматриваемой текстовой коллекции документов (при этом само число N выбиралось достаточно большим — 4 или 5). А так как морфологически инвариантные части словоформ (а именно уникальные корни) появляются реже, чем вариативные части (префиксы и суффиксы, такие как «ing» или «able»), то любая достаточно простая статистика, например IDF [91], может быть использована для их обнаружения.

В таблице 2.1, взятой из [101], показано распределение буквосочетаний (из 4 символов) слова «juggling» в зависимости от частоты их появления в коллекции документов (рассматривается коллекция CLEF 2002 [124]).

Таблица 2.1. Пример частот четырехбуквенных сочетаний слова

Четырехбуквенные сочетания	**Частота появления**
_jug	681
jugg	495
uggl	6775
ggli	3003
glin	4567
ling	55210
ing_	106463

Недостатком этого алгоритма является большой объем оперативной памяти, требуемый для хранения таблиц с частотами буквосочетаний. Достоинством можно считать тот факт, что алгоритм не зависит от выбора языка и является в этом смысле универсальным для решения задачи автоматического выделения основы.

Другие статистические алгоритмы аналитического выделения основы построены на основе концепции Скрытой Марковской Модели, СММ (*Hidden Markov Model, HMM*) [91].

Алгоритм, использующий в своей основе принципы СММ, был предложен М. Мелучи (Melucci) и Н. Орио (Orio) [102]. Преимущество такого подхода заключается в том, что для разработки алгоритма не требуется знать рассматриваемый язык и не требуется иметь созданную вручную обучающую выборку. Таким образом, так как вероятности перехода известны, можно определить (например, с помощью алгоритма Витерби [91]) наиболее вероятный путь в графе автомата.

В качестве состояний автомата СММ рассматриваются буквы, составляющие анализируемые словоформы. Все состояния разделены на две группы: корни и суффиксы; и на две категории: начальные (только корни) и терминальные (корни или суффиксы).

Последовательность переходов из одного состояния в другое представляет собой процесс порождения словоформы. Для любой поданной на вход алгоритма словоформы путь в графе из начального состояния в терминальное дает возможность определить точку, разделяющую словоформу на основу и окончание. Таким образом, часть словоформы, стоящая до этой точки, может быть выделена в качестве псевдоосновы.

Такая модель обладает большим потенциалом для дальнейшего развития, так как не нуждается в предварительном обучении и способна работать с

любым языком. Но она достаточно сложна для практического использования в системах, требующих высокой скорости обработки подаваемой на вход текстовой информации. Аналогичные идеи нашли отражение в работах [69, 70].

Проблему аналитического выделения основы можно рассматривать как задачу кластеризации с заранее не известным числом кластеров. Центром кластера является предполагаемая псевдооснова, а точнее словоформа, отстающая от других словоформ своего кластера на минимальное расстояние в смысле выбранной метрики. Основная проблема такого подхода заключается в выборе метрики расстояния между словоформами.

В работе [99] авторы предлагают алгоритм *YASS (Yet Another Suffix Stripper)*, который рассматривает несколько метрик, добавляющих значение для длинных совпадающих префиксов, и штрафует ранние несовпадения букв в сравниваемых словоформах. В наиболее простом варианте для двух рассматриваемых словоформ $\boldsymbol{X = x_0 x_1 \dots x_n}$ и $\boldsymbol{Y = y_0 y_1 \dots y_{n'}}$ штрафная функция определяется как:

$$\mathbf{p_i} = \begin{cases} \mathbf{0}, \mathbf{x_i = y_i}\ , \mathbf{0 \le i \le \min\ (n, n')} \\ \mathbf{1}, \text{иначе} \end{cases} \tag{2.1}$$

Если одна из рассматриваемых строк короче другой, то мы дополняем короткую строку пустыми символами для требуемой длины. Пусть теперь строки имеют длину $\boldsymbol{n + 1}$. Обозначим $\boldsymbol{m}$ позицию первого несовпадения $\boldsymbol{X}$ и $\boldsymbol{Y}$ (то есть, $\boldsymbol{x_0 = y_0, x_1 = y_1, x_{m-1} = y_{m-1}, x_m \neq y_m)}$. Тогда расстояние $\boldsymbol{D}$ может быть определено следующим образом:

$$\boldsymbol{D(X, Y) = \frac{n-m+1}{m} \sum_{i=m}^{n} \frac{1}{2^{i-m}}, m > 0} \tag{2.2}$$

иначе - ∞. Так для пары слов astronomer, astronomically $m = 8, n = 13$, таким образом $D_3 = \frac{6}{8}\left(\frac{1}{2^0} + \cdots + \frac{1}{2^{13-8}}\right) = 1.4766.$

Так как число кластеров заранее не определено, то кластеризация осуществляется с помощью иерархических методов. На практике чаще всего используют один из трех методов: метод одиночной связи, метод полной связи, метод средней связи [91].

Эксперименты, проведенные авторами на массивах текстов на английском и французском языках, показывают, что данный подход эффективен для языков флективного типа с преимущественно суффиксальным способом словоизменения. Однако кластеризацию таким методом можно производить, только если известны попарные расстояния между всеми словоформами. Этот факт не позволяет использовать данный подход для обработки текстовой информации в реальном времени.

Дальнейшее развитие алгоритмов аналитического выделения основы, базовым принципом работы которых является кластеризация словоформ, привело к появлению метода *GRAS (GRAph based Stemmer*) [108]. Данный алгоритм выглядит наиболее привлекательным для промышленного использования благодаря низким вычислительным ресурсам, требуемым для его работы, однако они все еще остаются достаточно высокими, по сравнению с алгоритмами, основанными на правилах грамматики.

Алгоритм *GRAS* состоит из двух этапов. На первом этапе по текстовому корпусу считается статистика суффиксов. Выделяют так называемые «сопутствующие суффиксы». Пара суффиксов $\langle \boldsymbol{s_1}, \boldsymbol{s_2} \rangle$ называется сопутствующей, если в текстовом корпусе существует такая пара различных словоформ $\langle \boldsymbol{w_i}, \boldsymbol{w_j} \rangle$, что $\boldsymbol{w_i} = \boldsymbol{r s_1}, \boldsymbol{w_j} = \boldsymbol{r s_2}$, при этом $\boldsymbol{r}$ — наибольший общий префикс словоформ $\boldsymbol{w}_\text{i}$ и $\boldsymbol{w_j}$. Таким образом, для каждой пары «сопутствующих

суффиксов» может быть определена частота их встречаемости в корпусе. Затем на основе этих данных строится ненаправленный граф $\boldsymbol{G} = (\boldsymbol{V}, \boldsymbol{E})$, в котором вершины — это словоформы текстового корпуса $\boldsymbol{L}$, а ребро между вершинами выставляется в том случае, если частота встречаемости их «сопутствующих суффиксов» больше заданного порога $\boldsymbol{\alpha}$, то есть $\boldsymbol{E} = \{(\boldsymbol{u}, \boldsymbol{v}) | \boldsymbol{w}(\boldsymbol{u}, \boldsymbol{v}) \geq \boldsymbol{\alpha}\}$. Теперь в этом графе выбираются классы близких словоформ. В качестве базовой выбирается вершина $\boldsymbol{u}$ с наибольшей степенью. Затем происходит итерирование по всем вершинам, непосредственно достижимым из выбранной вершины $\boldsymbol{u}$, и если их «связность» с базовой вершиной выше выбранного порога $\boldsymbol{\delta}$, то эта вершина попадет в класс вершины $\boldsymbol{u}$. После этого все вершины из построенного класса удаляются из графа $\boldsymbol{G}$. Процесс продолжается, пока в графе есть хотя бы одна вершина.

2.1.3. Смешанные методики выделения основ

Одна из основных проблем, с которой сталкиваются алгоритмы аналитического выделения основ, построенные на отсечении аффиксов (например, классический алгоритм Портера (Porter) или Ловинс (Lovins)), заключается в том, что они совершенно не берут во внимание семантику анализируемых словоформ. Так, словоформы «new» и «news» будут приведены к одной основе «new», хотя они относятся к разным парадигмам вне зависимости от контекста. Кроме того, смысл выделяемых основ может меняться в зависимости от корпуса анализируемых текстов.

В работе [121] Дж. Ксю (Xu) и У. Б. Крофт (Croft) был предложен метод, способный решить описанную проблему. Основная идея заключается в том, чтобы получить классы эквивалентности (множества словоформ, относящихся к одной выделенной основе) любым из существующих алгоритмов, а затем разделить словоформы из каждого класса в соответствии с их взаимным появлением в анализируемом корпусе. Аналогичная идея лежит в основе подхода, описанного в работе [83].

Авторы показали, что использование алгоритма Портера (Porter) в сочетании с триграммами для определения взаимного появления словоформ дает значительное повышение точности работы алгоритма.

Недостатком такого подхода является наличие дополнительно шага: проверки взаимного присутствия словоформ в корпусе. Это значительно снижает алгоритмическую эффективность данного метода и тем самым делает его плохо пригодным к использованию в реальных системах.

Среди статистических алгоритмов аналитического выделения основы можно выделить алгоритм, предложенный в работе [81]. Необычность изложенного в данной работе подхода заключается в том, что алгоритм использует статистическую модель в момент обработки пользовательского запроса, а не в момент индексации документов. Процедура аналитического выделения основы разделена на четыре этапа.

На первом этапе происходит построение кандидатов. Обычно для этого используется классический алгоритм Портера (Porter) [111], однако он не учитывает контекст анализируемой словоформы, что негативно влияет на качество работы данного алгоритма. Поэтому на данном этапе используется подход, анализирующий частоты взаимного появления анализируемых словоформ в корпусе [121]. В результате получается список кандидатов. Так, например, для словоформы develop список кандидатов будет следующим: developing, developed, develops, development, developer, developmental. На следующем этапе происходит выделение именных групп из запроса и выбирается главное слово в каждой выделенной группе. Теперь, когда имеются выделенные главные словоформы запроса, нужно определить, какое из его расширений будет наиболее подходящим вариантом. Для этого можно использовать статистическую естественно-языковую модель [91], позволяющую оценить вероятности взаимного появления словоформ во всем корпусе. На заключительном этапе происходит поиск наиболее вероятного

варианта расширения запроса в корпусе. Этот подход естественным образом отличается от классического подхода к поиску при использовании алгоритмов аналитического выделения основы, которые переводят исходный запрос к информационно-поисковой системе, например: (book OR books) and (store OR stores).

Очевидными недостатками такого подхода являются высокая вычислительная стоимость и сложная архитектура построения такого рода алгоритма аналитического выделения основы. Кроме того, на процедуру поиска в данном случае оказываем большое влияние алгоритм выделения именных групп.

Также существует алгоритм [80], с некоторым допущением относимый к статистическим, но при этом представляющий определенный теоретический интерес. Предлагаемый метод использует частоты последовательностей символов алфавита в тексте для определения псевдоосновы анализируемой словоформы. Так, если текст состоит из словоформ back, beach, body, backward, boy, то для того, чтобы определить так называемого «наследника разнообразия» для словоформы battle, нам требуется найти словоформы из текста, которые имеют с анализируемой словоформой наибольший общий префикс, и использовать статистику для ранжирования результатов.

Важно заметить, что все статистические методы аналитического выделения основы, несмотря на свои достоинства, в числе которых можно выделить отсутствие привязанности к конкретному языку и высокие показатели качества в ряде задач автоматического анализа текстовой информации, имеют два больших недостатка. Во-первых, для обеспечения высокого показателя качества алгоритм должен применяться к текстовым данным, которые будут иметь идентичную или достаточно близкую область, к которой они относятся. Так, нельзя применять алгоритм, обученный на литературных текстах, для

анализа сообщений с интернет-форумов, богатых специфической лексикой, и, соответственно, наоборот. Во-вторых, зачастую предлагаемые методы имеют крайне низкую скорость обработки анализируемых текстов, а то и вовсе не могут работать в реальном времени, позволяя осуществлять лишь обработку данных большими порциями или вообще целиком. В частности, это характерно для алгоритмов, основанных на кластеризации словоформ.

2.1.4. Реализации алгоритмов выделения основ

Рассматривая реализации алгоритмов аналитического выделения основы для различных языков, важно отметить проект М. Портера (Porter) Snowball [112], представляющий собой метаязык описания алгоритмов аналитического выделения основы. На официальном сайте проекта доступны реализации для английского, французского, испанского, португальского, итальянского, немецкого, нидерландского, шведского, норвежского (нюнорск), датского, русского и финского. Кроме этого доступны неофициальные реализации для литовского и словенского языков. Помимо собственно исходных файлов с описанием алгоритмов на метаязыке на сайте также представлены компиляторы в языки C++ и Java. Важно отметить, что сгенерированный компилятором код, в частности, на языке C++, трудночитаем и не оптимален в плане производительности.

Развитие алгоритма Портера детально рассмотрено в работе [120].

Подавляющее большинство алгоритмов аналитического выделения основы для индоевропейских языков базируется на правилах грамматики и представляет из себя процедуры отсечения аффиксов. Большой вклад в разработку этих алгоритмов был внесен швейцарским ученым Ж. Савоем (Savoy) [75]. Построены алгоритмы для словенского [110], французского [114, 115], латышского [95], финского [94], шведского [113], немецкого [72, 115], чешского [76, 100], греческого [103], венгерского [100], сербского [93],

болгарского [100, 116], латинского [117], польского [84] языков. Для скандинавских языков реализации приведены в работе [73].

Для языков агглютинативного морфологического типа, в частности для турецкого языка, был предложен алгоритм [79], который построен на основе правил грамматики языка, но последовательность его шагов представлена в виде конечного автомата [25]. Построение такого рода систем возможно для агглютинативных языков в виду особенностей морфемного состава словоформ и позволяет разрабатывать для этого класса языков быстрые алгоритмы аналитического выделения основы.

Для польского языка существует реализация алгоритма аналитического выделения основы, основанная на правилах [84]. Однако это не выписанные вручную лингвистами преобразования словоформ, а автоматически полученные команды вставки, удаления и замены. Подаваемая на вход словоформа проходит процедуру последовательных преобразований своих подстрок с целью получения псевдоосновы. Правила получаются на основе заранее подготовленной выборки вида словоформа – псевдооснова.

Для русского языка существует несколько реализаций алгоритмов аналитического выделения основы. На сайте проекта Snowball [112] приведена реализация алгоритма на метаязыке Snowball. Отличительной особенностью данного алгоритма является тот факт, что в нем учитывается и словообразовательная, и словоизменительная компонента языка, что может негативно влиять на корректность выделения псевдоосновы. Кроме того, правила в данном случае составлены не профессиональными лингвистами, что также негативным образом сказывается на качестве. В работе [77] предлагается алгоритм аналитического выделения основы, в основе которого лежит идея о том, что чаще всего в текстах на русском языке встречаются существительные и прилагательные. Таким образом, так как склонение данных частей речи носит регулярный характер, то можно выделить расширенные псевдофлексии,

включающие в себя в том числе словообразовательные суффиксы. В данном случае не учитывается словоизменение глаголов из-за трудоемкости составления правил. В своей работе авторы показывают возможность применения данного алгоритма к задаче построения информационно-поисковой системы, однако отмечают, что требуется использовать особую модель ранжирования.

Помимо этого, алгоритмы аналитического выделения основы для русского языка могут быть построены на основе статистических методов, наиболее перспективными из которых для решения этой задачи являются алгоритмы YASS и GRAS. Однако тут неминуемо возникнут проблемы, связанные с использованием статистических алгоритмов, о которых было сказано выше. Это и невозможность работы в реальном времени, и низкое качество выделения основ при анализе текстов, отличных по своей природе от тех, на которых осуществлялось обучение статистических моделей.

2.2. Оценки качества алгоритмов аналитического выделения основ

2.2.1. Методы оценки алгоритмов выделения основ

Для оценки качества разрабатываемых алгоритмов аналитического выделения основы могут быть использованы различные подходы. С одной стороны, можно оценивать качество всей информационно-поисковой системы в общем, а с другой — можно рассматривать показатели эффективности собственно алгоритмов аналитического выделения основы. Оба способа оценки качества несут полезный сигнал, на основе которого можно делать выводы о преимуществах использования того или иного алгоритма.

В случае построения на основе данных алгоритмов информационно-поисковых систем общего назначения мы могли бы оценить эффективность того или иного метода стандартным способом, описанным, например, в работе [104]. Измерив показатели полноты, точности и производных от них, можно было бы понять, насколько тот или иной алгоритм лучше или хуже своего аналога. Важно заметить, что для оценки влияния именно алгоритмов аналитического выделения основы на интегральные характеристики качества информационно-поисковых систем необходимо сохранять все компоненты, а именно структуру поискового индекса, формулу ранжирования, саму модель ранжирования т.д. в неизменном виде.

Такого рода эксперименты по оценке качества поиска требуют для своего осуществления наличия размеченной выборки, состоящей из запросов и «идеальных» ответов (лучшие 10, 20 или 30 результатов, упорядоченных по релевантности), которые должна выдавать информационно-поисковая система. Доступные в рамках различных форумов по оценке качества информационно-поисковых систем, таких как TREC [123], CLEF [124], РОМИП [66] и др., выборки включают слишком широкий спектр запросов, что не позволяет оценить качество информационно-поисковой системы специального

назначения, так как в выборках могут вообще отсутствовать запросы или документы по заданной предметной области, сформированные, например, с использованием специальной лексики. Помимо этого, правила участия некоторых форумов по оценке качества предполагают предоставление в полном объеме исходного кода информационно-поисковой системы. Также важной особенностью является тот факт, что массивы размеченных результатов приводятся для крайне ограниченного набора наиболее распространенных языков. Эта особенность заставляет строить вручную выборки для языков, не представленных в данных системах, что само по себе является достаточно трудоемкой задачей, требующей участия специалистов по рассматриваемому языку.

В работе [89] приводятся результаты экспериментов, на основе которых было показано, что использование алгоритмов аналитического выделения основы почти всегда оказывает положительный эффект и дает выигрыш в качестве. Исключение составляет использование этих алгоритмов на так называемых «цитатных запросах», когда нужно выдать документы, в которых данный запрос встречается в точной форме в полном объеме.

В работе [92] проводится детальный анализ эффективности аналитических и статистических методов аналитического выделения основы для нидерландского языка. Показывается целесообразность использования такого рода методов анализа при построении информационно-поисковых систем, однако подчеркивается факт снижения точности.

2.2.2. Показатели эффективности алгоритмов выделения основ

В отличие от методов, построенных на оценке качества информационно-поисковых систем, выделяют метод, получивший название «метода Пейса» (*Paice`s evaluation method*) [107]. Ошибкой алгоритма мы будем называть случай отнесения словоформы не к своей псевдооснове. При этом соответствие словоформа — псевдооснова устанавливается экспертом заранее. В

предлагаемом методе оценки качества алгоритмов аналитического выделения основы искомая характеристика определяется количеством обнаруживаемых ошибок, которое допускает тот или иной алгоритм.

Выделяются три класса отношений между двумя словоупотреблениями в тексте:

Тип 0. Два слова совпадают с точностью до словоформы;

Тип 1. Два слова представлены разными словоформами, но семантически эквивалентны;

Тип 2. Два слова представлены разными словоформами и семантически различны.

В терминах введенных типов отношений «хорошим» алгоритмом аналитического выделения основы можно назвать такой алгоритм, который сводит к одной псевдооснове максимально возможное количество пар слов Типа 1, при этом объединяя под одной псевдоосновой как можно меньше пар Типа 2. Численной величиной ошибок служат вводимые естественным образом характеристики ***Understemming Index*** (***UI***) и ***Overstemming Index*** (***OI***). ***UI*** определяется как относительное количество пар Типа 1, которые алгоритм аналитического выделения основы не привел в общей псевдооснове. ***OI*** определяется как относительное количество пар Типа 2, которые алгоритм аналитического выделения основы привел к одной псевдооснове.

Мы будем называть семантической группой совокупность словоформ, объединенных общим компонентом значения. В частности, в рамках семантической группы могут быть объединены все словоформы, составляющие парадигму некоторого слова. Кроме этого, допускается объединение в рамках одной семантической группы словоформ, соответствующих различным частям речи. Это позволяет в полной мере учитывать не только словоизменительные, но и словообразовательные аспекты языка, что допускается некоторыми вариантами алгоритмов аналитического выделения основы.

Если все словоупотребления текста разделены по семантическим группам, то для заданной семантической группы $\mathbf{g}$ можно определить величину $\boldsymbol{DMT_g}$, характеризующую количество пар, которое может быть получено из данной группы и которое будет являться ответом «идеального» алгоритма аналитического выделения основы:

$$\boldsymbol{DMT_g = \frac{1}{2} N_g (N_g - 1)} \quad (2.3)$$

где $\boldsymbol{N_g}$ — количество словоформ в семантической группе $\boldsymbol{g}$. Множитель $\frac{1}{2}$ вводится из-за того, что нас интересуют неупорядоченные пары словоформ из семантической группы. Просуммировав эту величину по всем семантическим группам текста мы получаем общий показатель:

$$\boldsymbol{GDMT = \sum_{i=1}^{n_g} DMT_{g_i}} \quad (2.4)$$

где $\boldsymbol{n_g}$ — общее количество семантических групп в тексте.

После применения алгоритма аналитического выделения основы все словоформы из текста будут преобразованы к псевдоосновам. При этом возможна ситуация, что в рамках одной семантической группы не все пары словоформ получат одинаковую псевдооснову. Эта невозможность алгоритма объединить все словоформы данной семантической группы g под одной псевдоосновой характеризуется показателем:

$$\boldsymbol{UMT_g = \frac{1}{2} \sum_{i=1}^{f_g} N_{gi} (N_g - N_{gi})} \quad (2.5)$$

где $\boldsymbol{f_g}$ — количество различных псевдооснов, выделенных в данной семантической группе $\boldsymbol{g}$, и $\boldsymbol{N_{gi}}$ — количество словоформ в семантической группе g, получивших псевдооснову $\boldsymbol{i}$. По аналогии с $\boldsymbol{GDMT}$ можно записать общую характеристику по всем группам:

$$\boldsymbol{GUMT = \sum_{i=1}^{n_g} UMT_{g_i}} \quad (2.6)$$

Используя выражения 2.4 и 2.6 $\boldsymbol{Understemming\ Index}$ ($\boldsymbol{UI}$) может быть записан как:

$$\boldsymbol{UI} = \frac{\boldsymbol{GUMT}}{\boldsymbol{GDMT}} \tag{2.7}$$

Эта характеристика показывает долю анализируемых словоформ, для которых должна была быть получена одна и та же псевдооснова, но в результате применения алгоритма были получены различные псевдоосновы.

Алгоритм аналитического выделения основы может относить к одной псевдооснове пары словоформ, принадлежащие разным семантическим группам. Таким образом, если все словоформы, получившие одну псевдооснову, были из одной семантической группы, то алгоритм не допустил ошибки. В противном случае можно оценить величину, характеризующую случаи неправильного выделения общей псевдоосновы для пары словоформ из различных семантических групп:

$$\boldsymbol{WMT_s} = \frac{\boldsymbol{1}}{\boldsymbol{2}} \sum_{\boldsymbol{i=1}}^{\boldsymbol{f_s}} \boldsymbol{N_{si}}(\boldsymbol{N_s} - \boldsymbol{N_{si}}), \tag{2.8}$$

где $\boldsymbol{N_s}$ — количество словоформ, отнесенных алгоритмом к псевдооснове $\boldsymbol{s}$, $\boldsymbol{f_s}$ — количество различных семантических групп, из представителей которых была получена псевдооснова $\boldsymbol{s}$, и $\boldsymbol{N_{si}}$ — количество псевдооснов, полученных в результате анализа словоформ из семантической группы $\boldsymbol{i}$. Общий показатель для всего анализируемого текста:

$$\boldsymbol{GWMT} = \sum_{\boldsymbol{i=1}}^{\boldsymbol{n_s}} \boldsymbol{WMT_{s_i}}, \tag{2.9}$$

где $\boldsymbol{n_s}$ — количество различных псевдооснов, выделенных в анализируемом тексте.

Каждая словоформа, не входящая в семантическую группу g, может быть объединена общей псевдоосновой со словоформами из данной семантической группы $\boldsymbol{g}$. Величина, характеризующая этот процесс:

$$\boldsymbol{DNT_g = \frac{1}{2} N_g (W - N_g)} \quad (2.10)$$

где $\boldsymbol{W}$ — общее количество словоупотреблений в рассматриваемом тексте. Просуммировав эту величину по всем семантическим группам текста мы получаем общий показатель:

$$\boldsymbol{GDNT = \sum_{i=1}^{n_g} DNT_{g_i}} \quad (2.11)$$

По аналогии с выражением 2.7, используя выражения 2.9 и 2.11 $\boldsymbol{Overstemming\ Index}$ ($\boldsymbol{OI}$) может быть записан как:

$$\boldsymbol{OI = \frac{GWMT}{GDNT}} \quad (2.12)$$

Эта характеристика показывает долю анализируемых словоформ, для которых должны были быть получены различные псевдоосновы, но в результате применения алгоритма были получены одинаковые псевдоосновы.

Отношение

$$\boldsymbol{SW = \frac{OI}{UI}} \quad (2.13)$$

называется «весом алгоритма аналитического выделения основы» и показывает, по сути, насколько сильно алгоритм учитывает словоизменительные и словообразовательные аспекты языка. Чем выше показатель $\boldsymbol{SW}$, тем в большей степени алгоритм учитывает словообразование и, соответственно, чаще сводит различные словоформы к одной псевдооснове.

Приведем пример вычисления описанных выше характеристик. В Табл. 2.2 представлены словоформы и основы, выделенные тривиальным алгоритмом, который оставляет первые n букв (взято n=5), а остальные отбрасывает. В Табл. 2.3 приведены результаты работы алгоритма для расчета коэффициента недостаточности отсечения.

Табл. 2.2. Результаты применения тривиального алгоритма выделения основы (n = 5).

Словоформа	**Основа**
divide	divid
dividing	divid
divided	divid
division	divis
divisor	divis
divine	divin
divination	divin

Табл. 2.3. Расчет коэффициента недостаточности отсечения UI.

	divide	dividing	divided	division	divisor	divine	divination
divide		+	+	-	-	x	x
dividing	+		+	-	-	x	x
divided	+	+		-	-	x	x
division	-	-	-		+	x	x
divisor	-	-	-	+		x	x
divine	x	x	x	x	x		+
divination	x	x	x	x	x	+	

Знаком «+» в таблице 2.3 отмечены правильные варианты получения основы, то есть те варианты, когда словоформы, относящиеся к одной парадигме, получают одинаковую основу; знаком «-» — варианты, когда словоформы, которые имеют одну парадигму, были отнесены к разным основам; знаком «x» отмечены варианты, которые не могут принимать участия в оценке показателя.

Так как всего знаков «+» получается 10, а всего знаков «+» и «-» 22, то значение коэффициента недостаточности отсечения равно:

$$\boldsymbol{UI = 1 - \frac{10}{22} = 0,545} \qquad (2.14)$$

В Табл. 2.4 приведены результаты работы алгоритма для расчета коэффициента избыточности отсечения.

Табл. 2.4. Расчет коэффициента избыточности отсечения $\boldsymbol{OI}$.

	divide	dividing	divided	division	divisor	divine	divination
divide		x	x	x	x	+	+
dividing	x		x	x	x	+	+
divided	x	x		x	x	+	+
division	x	x	x		x	+	+
divisor	x	x	x	x		+	+
divine	+	+	+	+	+		x
divination	+	+	+	+	+	x	

Знаком «+» в таблице 2.4 отмечены случаи, когда словоформы, не входящие в одну парадигму, получают разные основы; знаком «-» отмечены случаи, когда словоформы, не входящие в одну парадигму, получают одинаковые основы; знаком «x» отмечены варианты, которые не могут принимать участия в оценке показателя. Так как всего знаков «+» получается 20, а всего знаков «+» и «-» тоже 20, то значение коэффициента избыточности отсечения равно:

$$\boldsymbol{OI = 1 - \frac{20}{20} = 0} \qquad (1.4)$$

2.2.3. Оценка применимости алгоритмов выделения основ

Применимость того или иного алгоритма аналитического выделения основы в терминах количества допускаемых им ошибок в большой степени зависит от области применения. Для одних задач алгоритм с более низким показателем $\boldsymbol{OI}$ будет предпочтительнее, чем алгоритм с более низким показателем $\boldsymbol{UI}$, либо наоборот. Таким образом, требуется характеристика, способная одновременно учитывать оба противоположных по своей сути показателя $\boldsymbol{UI}$ и $\boldsymbol{OI}$. Кроме того, необходим некоторый базовый алгоритм аналитического выделения основы, относительно которого можно будет говорить о качестве предлагаемого алгоритма в терминах вводимой характеристики. На роль такого базового алгоритма очевидным образом подходит тривиальный алгоритм аналитического выделения основы, который

оставляет от анализируемой словоформы префикс заданной длины. Для каждого значения длины из выбранного диапазона можно определить показатели $\boldsymbol{UI}$ и $\boldsymbol{OI}$ соответствующего тривиального алгоритма аналитического выделения основы и поместить их на координатную плоскость $\boldsymbol{UI} \times \boldsymbol{OI}$. Затем, соединив указанные точки, мы получаем кривую, называемую «кривой усечения» (truncation line). Для оцениваемого алгоритма аналитического выделения основы мы также вычисляем его показатели $\boldsymbol{UI}$ и $\boldsymbol{OI}$ и помещаем соответствующую точку на ту же координатную плоскость. Обозначим данную точку P. Проведем прямую, проходящую через начало координат, точку O и данную точку P. Эта прямая пересечет кривую усечения в точке X. Тогда мы можем определить величину:

$$\boldsymbol{ERRT} = \frac{\boldsymbol{length\ OP}}{\boldsymbol{length\ OX}}, \tag{2.15}$$

где $\boldsymbol{length\ OP}$ — длина отрезка $\boldsymbol{OP}$, $\boldsymbol{length\ OX}$ — длина отрезка $\boldsymbol{OX}$. Иллюстрация процесса вычисления показателя $\boldsymbol{ERRT}$ приведена на рисунке 2.1.

В общем случае, чем меньше значение показателя $\boldsymbol{ERRT}$ для рассматриваемого алгоритма аналитического выделения основы, тем данный алгоритм лучше. Это объясняется тем, что наилучший алгоритм должен допускать как можно меньше ошибок обоих типов и, соответственно, иметь низкие показатели $\boldsymbol{UI}$ и $\boldsymbol{OI}$.

На рисунке 2.2 и в таблице 2.5 приведены описанные выше показатели качества для различных алгоритмов аналитического выделения основы для словоформ венгерского языка. Данные взяты из работы [118].

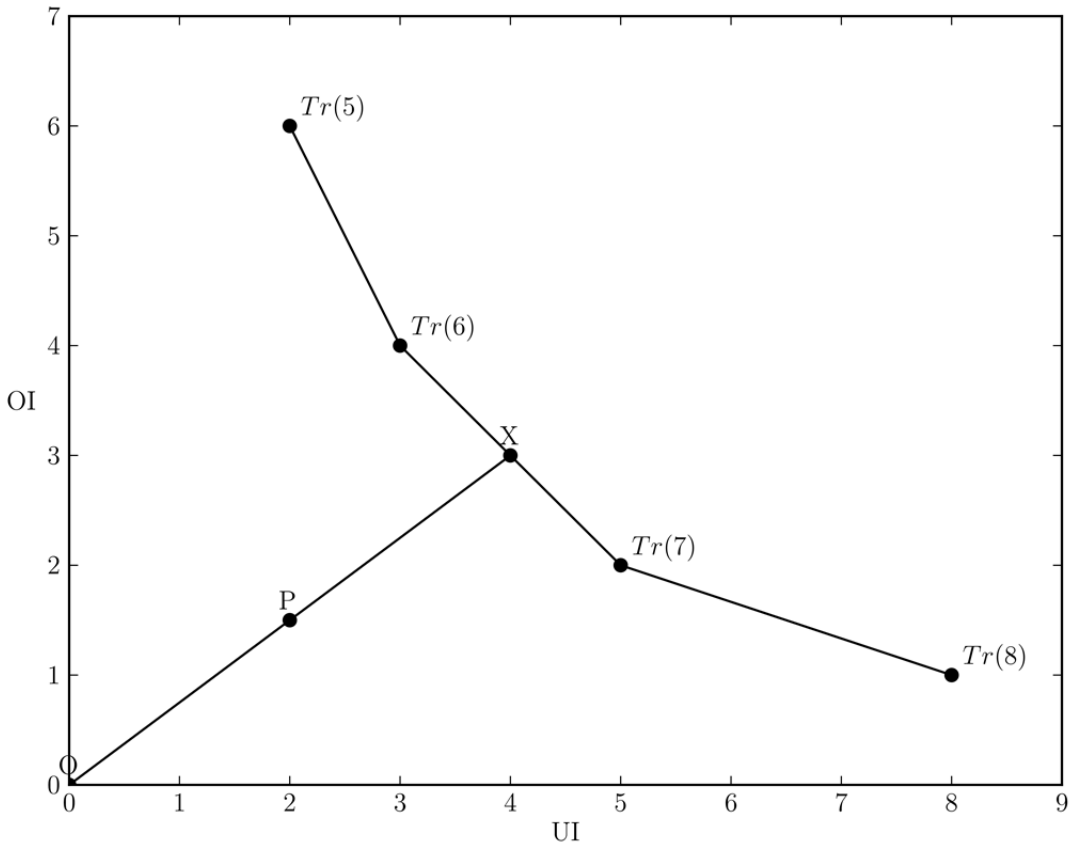

Рисунок 2.1. Вычисление ***ERRT***.

Таблица 2.5. Показатели качества алгоритмов аналитического выделения основы для словоформ венгерского языка

Алгоритм	***UI***	***OI***	***ERRT***
Light1	0.75	0.0000028	0.81
Light2	0.59	0.0000053	0.66
Medium	0.64	0.0000081	0.73
Heavy	0.53	0.0000134	0.65

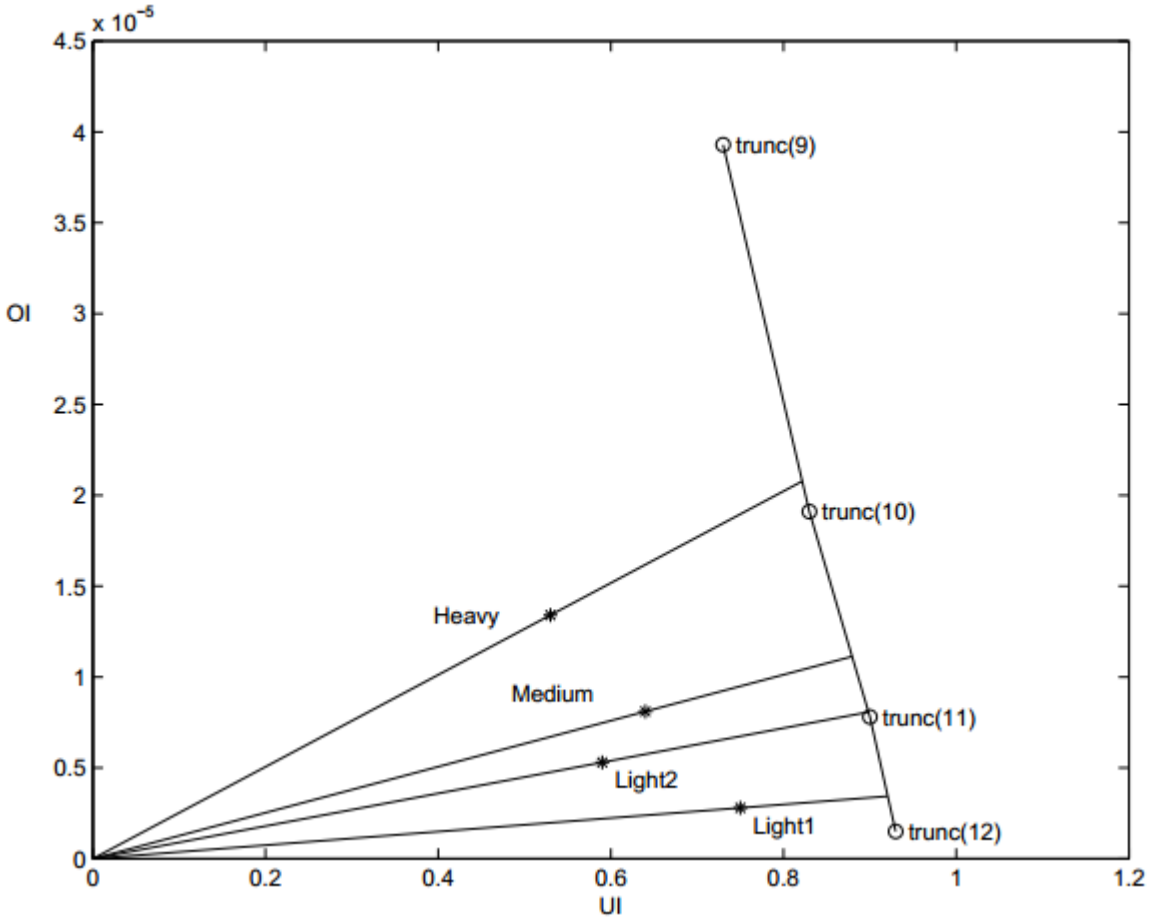

Рисунок 2.2. Показатель ***ERRT*** для различных алгоритмов аналитического выделения основы словоформ венгерского языка.

Помимо прямых показателей качества для аналитических алгоритмов выделения основ были предложены различные косвенные показатели, дающие при этом достаточно полезные сведения об эффективности разрабатываемых алгоритмов [81].

Количество слов, относящихся к одной парадигме:

$$\boldsymbol{MWC} = \frac{\boldsymbol{N}}{\boldsymbol{S}} \tag{2.16}$$

$\boldsymbol{N}$ — количество уникальных словоупотреблений до применения алгоритма,

$\boldsymbol{S}$ — количество уникальных основ, полученных после применения алгоритма.

Коэффициент сжатия индекса. Показывает, на сколько коллекция уникальных основ меньше коллекции уникальных словоформ:

$$\boldsymbol{ICF} = \frac{\boldsymbol{N-S}}{\boldsymbol{N}} \quad (2.17)$$

$\boldsymbol{N}$ — количество уникальных словоупотреблений до применения алгоритма,

$\boldsymbol{S}$ — количество уникальных основ, полученных после применения алгоритма.

Коэффициент нетривиальности. Показывает долю изменившихся словоформ из анализируемого текста:

$$\boldsymbol{NTI} = \frac{\boldsymbol{S_{NT}}}{\boldsymbol{N}} \quad (2.18)$$

$\boldsymbol{N}$ — количество уникальных словоупотреблений до применения алгоритма,

$\boldsymbol{S_{NT}}$ — количество уникальных основ, отличных от анализируемых словоформ, полученных после применения алгоритма.

Также важной для оценки качества алгоритмов аналитического выделения основы является величина, характеризующая среднюю степень модификации исходной словоформы по сравнению с полученной псевдоосновой. Для оценки такой величины подходит расстояние Левенштейна [18], определяемое как минимальное количество операций вставки одного символа, удаления одного символа и замены одного символа на другой, необходимое для преобразования одной строки (последовательности символов) в другую. Данную величину мы будем обозначать $\boldsymbol{MLD}$.

В реальных системах анализа текстовой информации, в частности, в информационно-поисковых системах, наибольший интерес для оценки представляют характеристики $\boldsymbol{ICF}$ и $\boldsymbol{MWC}$, так как они показывают реальное влияние алгоритма аналитического выделения основы на характеристики рассматриваемой системы. Остальные показатели, такие как $\boldsymbol{UI, OI, SW, ERRT}$, представляют в большей степени академический интерес и сложно применимы для оценки качества реальных систем, в первую очередь из-за необходимости ручного составления выборок, по которым будет производиться оценка указанных величин.

2.3. Обобщенная схема построения алгоритмов аналитического выделения основ

Для каждого языка можно выделить набор минимальных элементов — аффиксов, используемых для образования различных форм слова. Различают несколько типов аффиксов: префиксы, словообразовательные и словоизменительные суффиксы, флексии. Таким образом, любой алгоритм аналитического выделения основы построен в виде последовательности отсечения аффиксов при выполнении каких-либо условий.

Введем ряд определений, необходимых для формулирования обобщенной процедуры анализа словоформ.

Обозначим $\boldsymbol{\Sigma}$ — алфавит языка, $\boldsymbol{w} \in \boldsymbol{\Sigma}^*$ — слово на данном языке. Тогда

$$\boldsymbol{w = w_1 \dots w_n, w_i \in \Sigma} \tag{2.19}$$

В грамматиках языков могут быть определены правила удвоения согласных, снятия или добавления диакритических знаков, явления фузии на стыках морфем. Для учета всех этих явлений в обобщенную модель вводится понятие функции трансформации

$$\boldsymbol{f_T: \Sigma^* \to \Sigma^*} \tag{2.20}$$

Конкретизируем (2.20):

$$\mathbf{f_T(w_1 \dots w_{i-1} w_i w_{i+1} \dots w_n) = w_1 \dots w_{i-1} w_i' w_{i+1} \dots w_n, w_i \in \Sigma, w_i' \in \Sigma^*} \tag{2.21}$$

Конкретная функция трансформации преобразует один символ подаваемой на вход последовательности (слова). Преобразование произвольных символов может быть записано в виде композиции элементарных функций трансформации. Таким образом, можно определить множество функций трансформации

$$\boldsymbol{F_T} = \{\boldsymbol{f_{T_1}}, \dots, \boldsymbol{f_{T_P}}\} \quad (2.22)$$

Для удобства описания обобщенной схемы анализа словоформ мы будем представлять это множество в виде объединения двух множеств

$$\boldsymbol{F_T} = \boldsymbol{F_{TNC}} \cup \boldsymbol{F_{TC}}, \quad (2.23)$$

где $\boldsymbol{F_{TNC}}$ — множество функций трансформации, применение которых не зависит от выполнения каких-либо условий, и $\boldsymbol{F_{TC}}$ — множество функций трансформации, применение которых зависит от выполнения условий (ограничений). Также для функций трансформации из множества $\boldsymbol{F_{TNC}}$ можно определить обратные функции, то есть функции, совершающие обратные преобразования. Множество таких функций мы будем обозначать $\boldsymbol{F_{TNC}^{-1}}$.

Определим функцию проверки условия

$$\boldsymbol{f_C}: \boldsymbol{\Sigma}^* \to \{\mathbf{0}, \mathbf{1}\}, \quad (2.24)$$

которая по последовательности символов (слову) предоставляет ответ, удовлетворяет ли данная последовательность символов заданному условию. Определим множество функций проверки условия

$$\boldsymbol{F_C} = \left\{\boldsymbol{f_{C_1}}, \dots, \boldsymbol{f_{C_Q}}\right\} \quad (2.25)$$

Например, функция $\boldsymbol{f_C} \in \boldsymbol{F_C}$ может определять принадлежность буквы заданному набору букв.

Как было сказано ранее, для каждого языка можно выделить набор аффиксов, участвующих в словоизменительных явлениях. В предлагаемой обобщенной модели анализа мы рассматриваем только суффиксы и флексии, так как с помощью морфем этих типов происходит словоизменение во всех рассматриваемых нами языках. Таким образом, для данного языка можно определить множество аффиксов

$$\boldsymbol{T} = \{\boldsymbol{t_1}, \dots, \boldsymbol{t_L}\}, \boldsymbol{t_i} \in \boldsymbol{\Sigma}^*, \tag{2.26}$$

где $\boldsymbol{L}$ — общее количество аффиксов, определяемых для данного языка.

Определим функцию

$$\boldsymbol{f_A}: \boldsymbol{\Sigma}^* \times \boldsymbol{T} \to \boldsymbol{T} \cup \{\boldsymbol{\epsilon}\} \tag{2.27}$$

следующим образом:

$$\boldsymbol{f_A}(\boldsymbol{w}, \boldsymbol{T^I}) = \left\{\boldsymbol{t_i}: \forall \boldsymbol{j} = \mathbf{1}, \dots, \boldsymbol{k}, \boldsymbol{i} \neq \boldsymbol{j}\ |\boldsymbol{t_i}| > \left|\boldsymbol{t_j}\right|, \boldsymbol{t_i} \in \boldsymbol{T'}\right\} \cup \{\boldsymbol{\epsilon}\}, \tag{2.28}$$

где $\boldsymbol{T'} = \{\boldsymbol{t'_1}, \dots, \boldsymbol{t'_k}\}$, $\boldsymbol{t'_p} \in \boldsymbol{\Sigma}^*$ и словоформа $\boldsymbol{w}$ представима в виде $\boldsymbol{w} = \boldsymbol{w't'_p}$, и при этом $\boldsymbol{T'} \subset \boldsymbol{T^I}$. Данная функция по последовательности символов и множеству аффиксов выдает в качестве ответа самый длинный аффикс, на который оканчивается данная последовательность символов. Если такого аффикса не удалось найти среди элементов множества $\boldsymbol{T'}$, то возвращается пустая строка.

Обобщенную схему построения алгоритмов аналитического выделения основы можно сформулировать в форме следующего алгоритма:

<u>Обобщенный алгоритм аналитического выделения основы.</u>

Вход: слово $\boldsymbol{w} = \boldsymbol{w^0} = \boldsymbol{w_1} \dots \boldsymbol{w_n}, \boldsymbol{w_i} \in \boldsymbol{\Sigma}$

Выход: псевдооснова слова $\boldsymbol{w}$: $\mathbf{s} = \mathbf{s_1} \dots \mathbf{s_k}, \mathbf{k} \leq \mathbf{n}, \mathbf{s_i} \in \boldsymbol{\Sigma}$

1. На основе грамматики языка сформировать множества аффиксов $\boldsymbol{T^1}, \dots, \boldsymbol{T^N}$, отвечающих за различные словоизменительные аспекты данного языка. При

этом все допустимые аффиксы языка образуют множество $\boldsymbol{T} = \bigcup_{\boldsymbol{i=1}}^{\boldsymbol{N}} \boldsymbol{T^i}$, где $\boldsymbol{T^i} = \{\boldsymbol{t_1}, \dots, \boldsymbol{t_Q}\}, \boldsymbol{t_j} \in \boldsymbol{\Sigma^*}$.

2. На основе грамматики языка определить множества функций $\boldsymbol{F_{TNC}}, \boldsymbol{F_{TNC}^{-1}}$, при этом мощности этих множеств совпадают $|\boldsymbol{F_{TNC}}| = |\boldsymbol{F_{TNC}^{-1}}| = \boldsymbol{P}$. Также определить множества функций проверки условия $\boldsymbol{F_C}, |\boldsymbol{F_C}| = \boldsymbol{S}$ и функций трансформации $\boldsymbol{F_{TC}}, |\boldsymbol{F_{TC}}| = \boldsymbol{R}$.
3. $\forall \boldsymbol{i} = \boldsymbol{1}, \dots, \boldsymbol{P}\ \boldsymbol{w^i} = \boldsymbol{f_{TNC_i}}(\boldsymbol{w^{i-1}}), \boldsymbol{f_{TNC_i}} \in \boldsymbol{F_{TNC}}$. Здесь $\boldsymbol{w^i}$ — слово $\boldsymbol{w}$ после применения функции трансформации $\boldsymbol{f_{TNC_i}}$. Обозначим слово $\boldsymbol{w}$ после применения всех функций трансформации как $\boldsymbol{w^{[3]}}$, то есть $\boldsymbol{w^P} = \boldsymbol{w^{[3]}}$.
4. Обозначим $\boldsymbol{w^{[4]}} = \boldsymbol{w^{[3]}}$. Тогда $\forall \boldsymbol{i} = \boldsymbol{1}, \dots, \boldsymbol{N}: \boldsymbol{f_A}(\boldsymbol{w^{[4]}}, \boldsymbol{T^i}) = \boldsymbol{t_i}$, при этом $\boldsymbol{w^{[4]}} = \boldsymbol{w_1^{[4]}} \dots \boldsymbol{w_e^{[4]}} \boldsymbol{t_{i_1}} \dots \boldsymbol{t_{i_s}}$. Тогда, если $\boldsymbol{t_i} \neq \boldsymbol{\epsilon}$ и $\boldsymbol{f_{C_j}}(\boldsymbol{w^{[4]}}) = \boldsymbol{1}$, то $\boldsymbol{w^{[4]}} = \boldsymbol{w_1^{[4]}} \dots \boldsymbol{w_e^{[4]}}$. Также если $\exists \boldsymbol{f_{C_j}}(\boldsymbol{w^{[4]}}) = \boldsymbol{1}$, то применить соответствующую функцию трансформации $\boldsymbol{w^{[4]}} = \boldsymbol{f_{T_j}}(\boldsymbol{w^{[4]}})$.
5. $\boldsymbol{w^{[5]}} = \boldsymbol{w^{[4]}}\ \boldsymbol{u}\ \forall \boldsymbol{f_{TNC_i}^{-1}} \in \boldsymbol{F_{TNC}^{-1}}: \boldsymbol{w^{[5]}} = \boldsymbol{f_{TNC_i}^{-1}}(\boldsymbol{w^{[5]}})$
6. В качестве псевдоосновы вернуть $\boldsymbol{s} = \boldsymbol{w^{[5]}}$.

Таким образом, алгоритм аналитического выделения основы представляет собой процедуру последовательного отсечения аффиксов анализируемой словоформы при выполнении ряда условий (ограничений). Собственно отсечение найденного самого длинного аффикса происходит на шаге 4 обобщенного алгоритма анализа словоформы. Также на этом шаге осуществляется учет возможных фузионных явлений на стыках морфем (в данном случае аффиксов).

Шаги 3 и 5 являются вспомогательными. На них происходит модификация (и, соответственно, обратная модификация) словоформы с целью упрощения дальнейшей процедуры анализа. Например, на шаге 3 ряд гласных могут быть

заменены на согласные, потому что для некоторых языков при определенном сочетании букв они являются разделителями слогов в словоформе.

Использование различных структур данных и алгоритмов поиска в рамках обобщенной схемы анализа словоформ позволяет строить наиболее эффективные алгоритмы аналитического выделения основы, так как именно такой подход в наибольшей степени учитывает особенности конкретных языков. При этом, благодаря общей схеме анализа, оказываются разделены алгоритмический уровень и уровень конкретной реализации для заданного языка.

2.4. Классификация языков по методам реализации алгоритмов

На основании преобладания определенных морфологических явлений в каждом отдельно взятом языке можно разрабатывать специфические алгоритмы аналитического выделения основы. При этом важно подчеркнуть, что языки с преобладанием определенного морфологического типа обладают рядом общих характеристик, позволяющих формулировать алгоритмы для всех языков данного типа на основе общего подхода.

Для аналитических языков характерно словоизменение посредством служебных слов (например, предлогов). Таким образом, в языках данного типа используется малое количество словоизменительных аффиксов, что позволяет разработать для языков данного типа эффективные алгоритмы аналитического выделения основы.

Для языков флективного типа характерно наличие большого числа словоизменительных аффиксов и сложных правил употребления данных аффиксов. Зачастую в языках данного типа происходят фузионные явления на стыках морфем и чередования в корне слова. Напротив, в языках агглютинативного типа, несмотря на наличие большого числа аффиксов,

образование различных форм слова носит, как правило, регулярный характер. При этом действуют законы гармонии гласных и практически исключены чередования.

Все эти особенности морфологического строения языков различных типов находят непосредственное отражение в процедурах аналитического выделения основы для этих языков. При этом важно подчеркнуть, что языки определенного морфологического типа обладают рядом общих характеристик, позволяющих формулировать алгоритмы для всех языков данного типа на основе общего подхода.

Так как практически ни один язык нельзя однозначно отнести к определенному морфологическому типу, мы вводим собственную классификацию, которая позволяет нам на основе преобладающих явлений в языке формулировать алгоритмы аналитического выделения основы в терминах обобщенной схемы, описанной нами ранее. В результате мы получаем наиболее эффективные реализации алгоритмов для языков различных типов, учитывая при этом в максимально возможной степени морфологические явления каждого отдельно взятого языка.

В таблице 2.6 приведена классификация языков Евразии по типам алгоритмов выделения основ.

В соответствии с особенностями каждого языка, продиктованными конкретным морфологическим типом, требуется использование специализированных структур данных, обеспечивающих наиболее эффективное решение задачи автоматического выделения псевдоосновы анализируемой словоформы.

Для аналитических языков характерно наличие малого числа морфем, участвующих в словоизменении. Поэтому в алгоритме анализа использование древовидных структур данных оказывается неэффективным, так как требует

дополнительных затрат памяти и времени на построение. В итоге выбор делается в пользу массива с линейным или бинарным поиском элементов в нем.

Языки с преобладанием флективных свойств богаты разного рода словоизменительными аффиксами. Таким образом, для повышения быстродействия алгоритмов аффиксы хранятся в древовидной структуре данных типа бор. Так как в алгоритмах зачастую нужен поиск наиболее длинного аффикса анализируемой словоформы, то в боре хранится дополнительная служебная информация, например, о длине аффикса.

Агглютинативные языки отличаются наличием большого числа аффиксов (причем они могут быть выражены и суффиксами, и префиксами). Также особенностью данного типа языков является определенная схема словоизменения (хотя имеется ряд исключений, в целом схему можно считать достаточно регулярной). Таким образом, наиболее подходящим решением для реализации алгоритмов анализа словоформ агглютинативных языков является использование конечных автоматов. При этом в рамках обобщенной схемы изменяется только смысл поиска: вместо поиска по бору применяется поиск по исходящим из состояния дугам, что можно трактовать одинаково в терминах обобщенного алгоритма анализа словоформ.

Данная классификация (таблица 2.6) позволяет на основе преобладающих явлений в языке формулировать алгоритмы аналитического выделения основы в терминах обобщенной схемы, описанной выше (п. 2.4). В результате мы получаем наиболее эффективные реализации алгоритмов для языков различных типов, учитывая при этом в максимально возможной степени морфологические явления каждого отдельно взятого языка.

Таблица 2.6. Классификация языков по типам алгоритмов выделения основ.

№	Язык	Морфологический тип	Тип алгоритмической реализации
1	азербайджанский	агглютинативный	III
2	английский	аналитический	I
3	албанский	флективный	II
4	арабский	флективный	II
5	армянский	преимущественно агглютинативный с элементами аналитизма	III
6	башкирский	агглютинативный	III
7	белорусский	флективный	II
8	болгарский	аналитический	I
9	венгерский	агглютинативный с элементами флективности	III
10	вьетнамский	аналитический	I
11	греческий	флективный	II
12	грузинский	агглютинативный с элементами флективности	III
13	датский	аналитический	I
14	иврит	флективный с элементами аналитизма	II
15	исландский	флективный	II
16	испанский	флективно-аналитический	II
17	итальянский	флективно-аналитический	II
18	казахский	агглютинативный	III
19	каталанский	флективно-аналитический	II
20	киргизский	агглютинативный	III
21	корейский	агглютинативный	III
22	курдский	флективный	II
23	латышский	флективный	II
24	литовский	флективный	II
25	македонский	аналитический	I
26	молдавский	флективно-аналитический	II

27	монгольский	агглютинативный	III
28	немецкий	флективный с элементами аналитизма	II
29	нидерландский	аналитический	I
30	норвежский	аналитический	I
31	осетинский	агглютинативно-флективный	III
32	персидский	флективно-аналитический	II
33	польский	флективный	II
34	португальский	флективный с элементами аналитизма	II
35	румынский	флективно-аналитический	II
36	русский	флективный	II
37	сербскохорватский	флективный	II
38	словацкий	флективный	II
39	словенский	флективный	II
40	таджикский	флективно-аналитический	II
41	татарский	агглютинативный	III
42	турецкий	агглютинативный	III
43	туркменский	агглютинативный	III
44	узбекский	агглютинативный с элементами аналитизма	III
44	украинский	флективный	II
45	финский	агглютинативный с элементами флективности	III
46	французский	флективно-аналитический	II
47	чешский	флективный	II
48	шведский	аналитический	I
49	эстонский	агглютинативный с элементами флективности	III
50	японский	агглютинативный	III

2.5. Принипы реализации алгоритмов автоматического анализа словоформ I и II типов

Программные реализации алгоритмов аналитического выделения основы для языков I и II типов, в соответствии с классификацией, приведенной в таблице 2.6, во многом совпадают между собой.

Как следует из обобщенной схемы построения алгоритмов аналитического выделения основы, для рассматриваемого языка формируются списки аффиксов, участвующих в словоизменительных явлениях, например, в склонении имен существительных или спряжении глаголов. Кроме этого, формулируются функции трансформации (и обратные к ним), а также функции проверки свойств анализируемой словоформы. Для языков, относящихся к I и II типу, такие функции, как правило, представляют собой проверку схождения букв в заданную часть анализируемой словоформы. Также данные функции могут проверять различные свойства, связанные с длиной анализируемой словоформы.

Основное отличие в реализации алгоритмов для языков **I** и **II** типов заключается в использовании специализированных структур данных для хранения множества аффиксов. Так, для языков I типа, где, как правило, множество аффиксов по мощности меньше, чем множество аффиксов для языков II типа, нет необходимости в построении специализированных древовидных структур данных, обеспечивающих быстрый поиск при большом размере множества анализируемых аффиксов. Таким образом, для языков **I** типа множество аффиксов хранится в массиве, а выполнение функции $\boldsymbol{f_A}$ из обобщенной схемы представляет собой линейный поиск подходящего аффикса. В результате для языков **I** типа происходит экономия по памяти, так как не требуется специализированных структур данных, а также мы не получаем сильного проигрыша по производительности в силу малых размеров множеств аффиксов.

Для языков **II** типа, напротив, характерно наличие большого число словоизменительных аффиксов. Эта особенность заставляет нас хранить аффиксы каждого типа в древовидной структуре данных типа бор. В этом случае выполнение функции f_A сводится к одновременному проходу по анализируемой словоформе «справа налево» и проходу по бору от корня к листьям, так как аффиксы хранятся в боре в инвертированном виде. Такая процедура поиска позволяет без дополнительных затрат найти самый длинный аффикс из заданного множества. Однако в данном случае, по сравнению с алгоритмами для языков **I** типа, для анализа словоформы будет требоваться больше оперативной памяти, так как будет использована специализированная структура данных. Но это позволяет сократить время поиска, которое является линейным в случае перебора всех аффиксов соответствующего множества, как это сделано для языков **I** типа.

Таким образом, нами решается наиболее оптимальным образом задача построения алгоритмов аналитического выделения основы при условии потребления минимального объема оперативной памяти и максимального быстродействия в рамках выполнения поиска аффиксов в анализируемой словоформе, что делает предлагаемые алгоритмы применимыми в системах обработки текстовой информации, работающих в реальном масштабе времени с большими объемами данных.

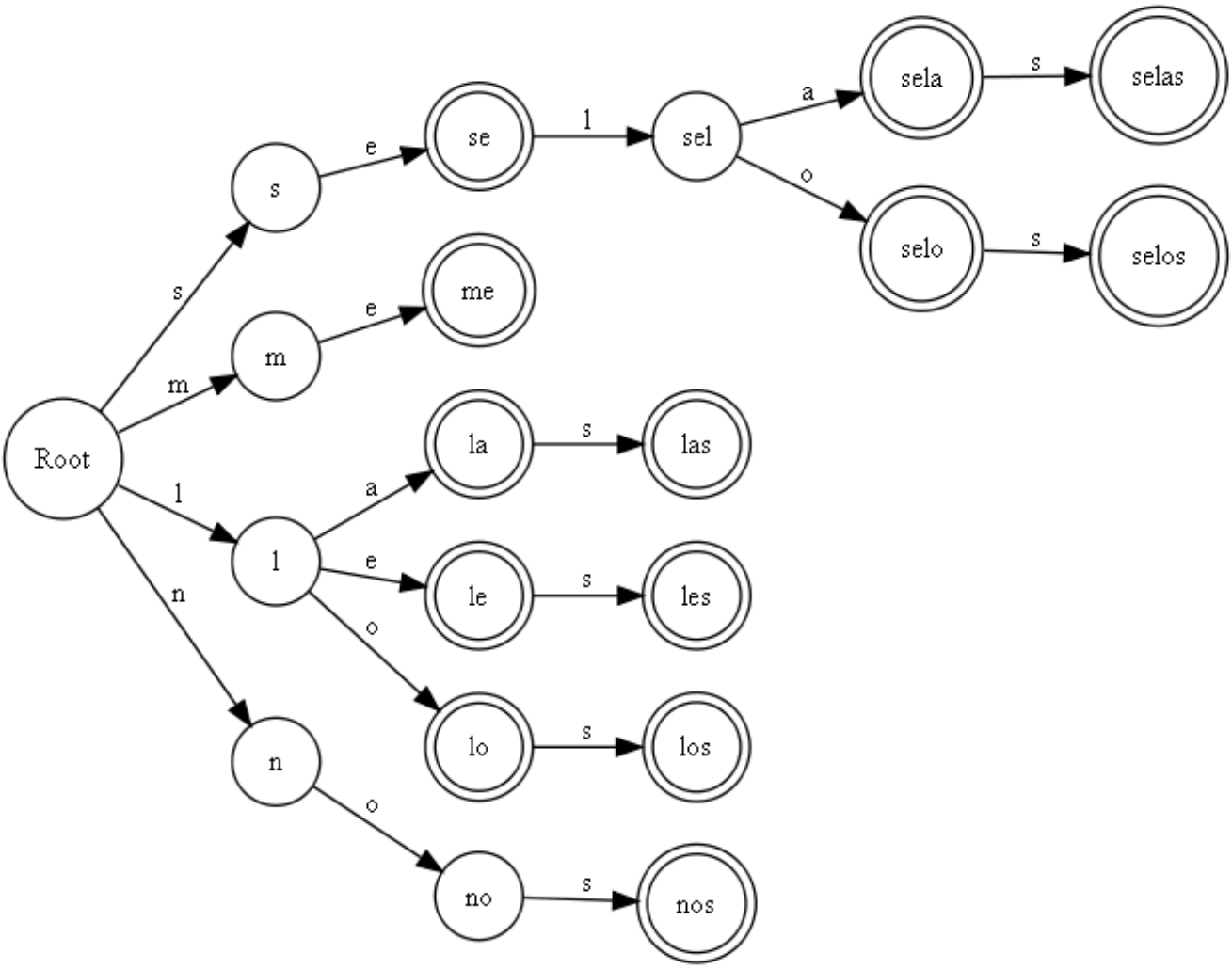

Рисунок 2.3. Бор для аффиксов испанского языка.

На рисунке 2.3 приведен бор для списка аффиксов T_1 из алгоритма для испанского языка (п. 3.3). Дважды обведены узлы бора, являющиеся терминальными, то есть в которых хранится конкретный аффикс. Корневой узел бора помечен Root.

2.6. Принипы реализации алгоритмов автоматического анализа словоформ III типа

Для языков, относимых нами к **III** типу, как было сказано ранее, характерно наличие большого количества словоизменительных и словообразовательных аффиксов. Кроме этого, правила, в соответствии с которыми в языках этого типа происходит и словообразование, и

словоизменение, носят регулярный характер и практически во всех случаях могут быть четко формализованы. Так, для финского языка, как показано в главе 5, можно четко сформулировать схему образования различных форм имени существительного. Конечно, в отдельных языках могут быть явления, связанные с требованиями согласованности гласных в частности или в общем с сочетаемостью букв. Такого рода условия могут быть учтены с помощью функций проверки условий и зависимых от них правил трансформации, определяемых в соответствии с обобщенной схемой построения алгоритмов аналитического выделения основы.

В результате мы приходим к тому, что процедура поиска и отсечения аффиксов, соответствующая выполнению функции $\boldsymbol{f_A}$ в обобщенной схеме и являющаяся основной в любом алгоритме аналитического выделения основы, может быть сформулирована в виде последовательности переходов между состояниями некоторого конечного автомата. На рисунках 2.4 и 2.5 приведены примеры таких конечных автоматов для финского и турецкого языков соответственно. Подробно их структура и использование для анализа словоформ приведены в главе 5.

Таким образом, в плане скорости обработки словоформ можно получить результаты, близкие к алгоритмам для анализа словоформ языков **I** типа. Это происходит за счет практически полного отсутствия проверок специализированных условий, накладываемых грамматикой языка на случаи присоединения аффиксов, что радикальным образом отличает языки **III** типа от языков **I** и, в особенности, **II** типа.

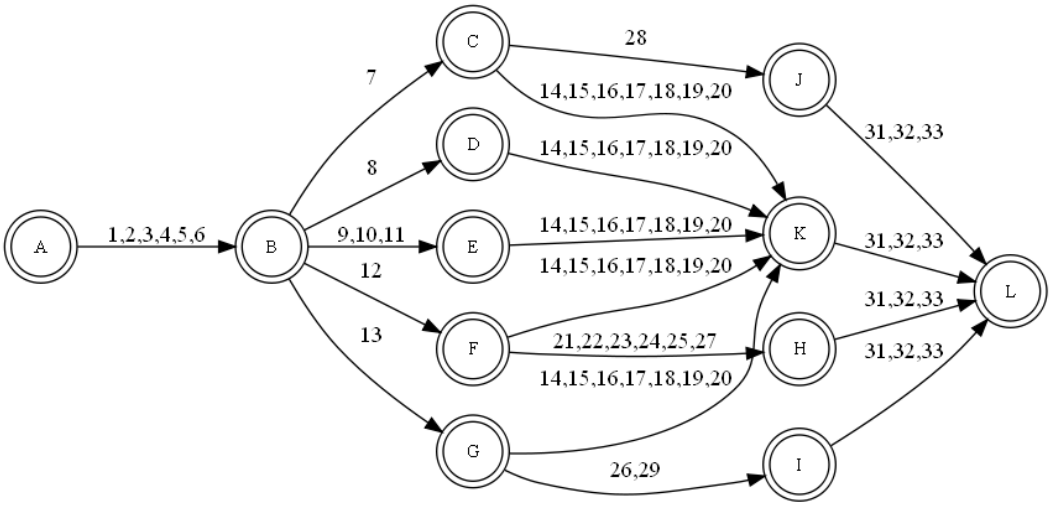

Рисунок 2.4. Конечный автомат для отсечения словоизменительных аффиксов имени существительного.

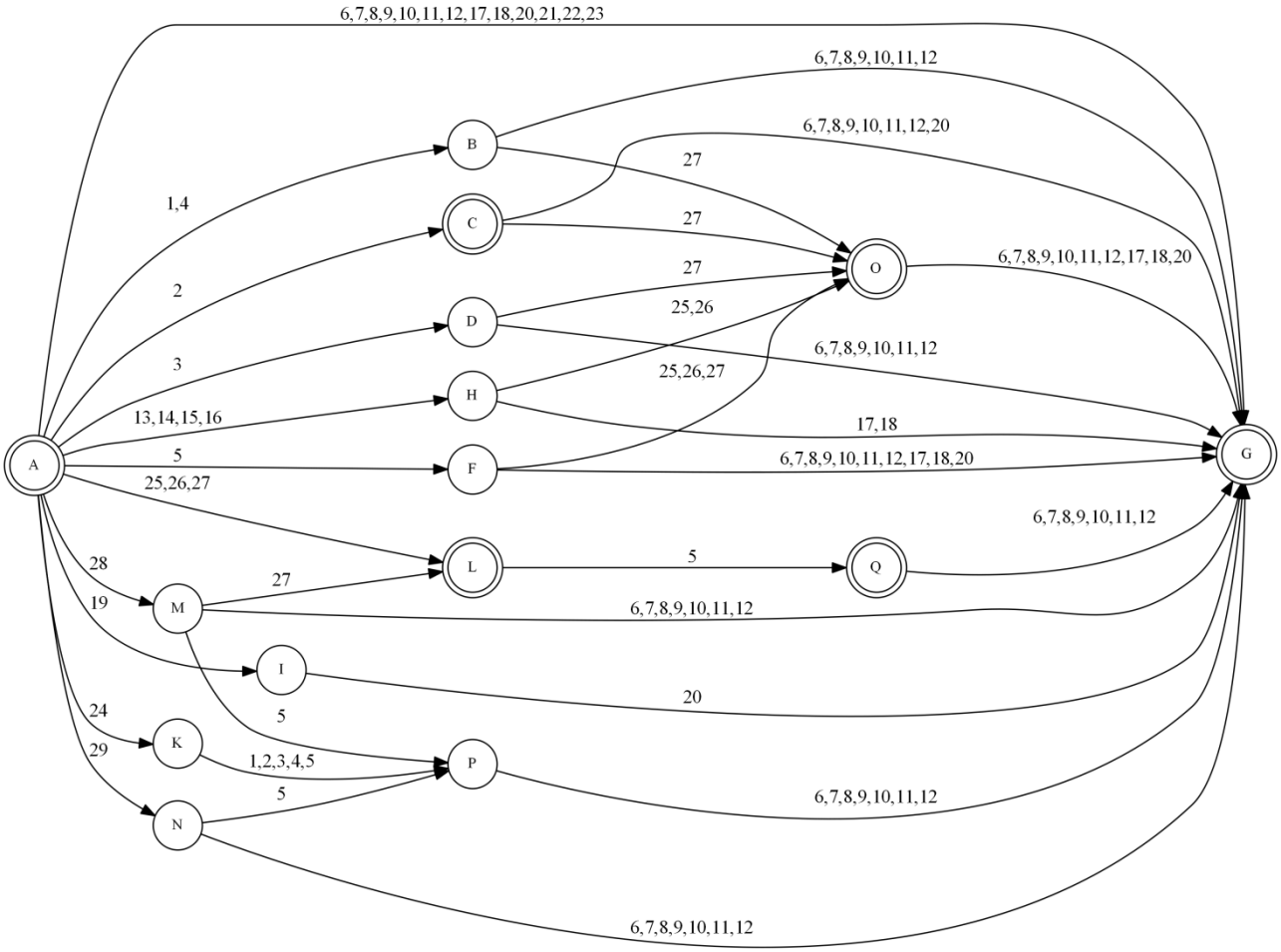

Рисунок 2.5. Конечный автомат для отсечения словоизменительных (лицо, время, наклонение) аффиксов глагола в турецком языке.

3. АЛГОРИТМЫ ВЫДЕЛЕНИЯ ОСНОВ I ТИПА

Требования к усвоению данного раздела учебного пособия:

Знать: иннновационные особенности различных видов печатных и электронных средств информации, основные технологические процессы производства печатных и электронных средств информации;

Уметь: использовать компьютерную технику в решении конкретных практических задач;

Владеть навыками: работы с прикладными программными средствами.

3.1. Английский язык

Современный английский язык представляет собой язык ярко выраженного аналитического типа. Для выражения семантико-грамматических отношений преимущественно используются служебные слова и порядок слов.

В английском языке существующие буквы являются гласными: a, e, i, o, u, y.

Алгоритм начинает работу с того, что переводит y, стоящую после гласной, в верхний регистр. Этот этап соответствует применению функций трансформации $f \in F_T$ в схеме обобщенного алгоритма анализа словоформы.

Определим R_1 как область, начинающуюся после первой согласной буквы, следующей за гласной. Если такой согласной нет, то область пустая. R_2 — область, начинающаяся после первой согласной буквы, следующей за гласной в области R_1. Аналогично, если нет такой согласной, то область пустая.

R_1 и R_2 первоначально настраиваются по умолчанию, но затем R_1 увеличивается так, чтобы область до него содержала минимум 3 буквы.

Определим список удвоенных согласных: bb, dd, ff, gg, mm, nn, pp, rr, tt.

Определим L-окончание как одну букву из списка: c, d, e, g, h, k, m, n, r, t.

Определим понятие «короткий слог» как гласную, за которой следует согласная, отличная от w, x либо Y и которой предшествует согласная, либо как гласную, стоящую в начале слова, за которой следует согласная.

Для английского языка в терминах обобщенного алгоритма определяются следующие функции $f \in F_C$:

- Буква является LI -окончанием;
- Буква (или последовательность букв) располагается в области R_1;
- Буква (или последовательность букв) располагается в области R_2.
- Буква является «коротким слогом».

В соответствии со схемой обобщенного алгоритма анализа будут сформированы списки аффиксов $T_1, \dots, T_{33}$:

- $T_1 = (', 's, 's')$;
- $T_2 = (sses)$;
- $T_3 = (ied, ies)$;
- $T_4 = (s)$;
- $T_5 = (us, ss)$;
- $T_6 = (eed, eedly)$;
- $T_7 = (ed, edly, ing, ingly)$;
- $T_8 = (tional)$;
- $T_9 = (enci)$;
- $T_{10} = (anci)$;
- $T_{11} = (abli)$;
- $T_{12} = (entli)$;
- $T_{13} = (izer, ization)$;
- $T_{14} = (ational, ation, ator)$;
- $T_{15} = (alism, aliti, alli)$;
- $T_{16} = (fulness)$;
- $T_{17} = (ousli, ousness)$;

- $T_{18} = (iveness, iviti)$;
- $T_{19} = (biliti, bli)$;
- $T_{20} = (ogli)$;
- $T_{21} = (fulli)$;
- $T_{22} = (lessli)$;
- $T_{23} = (li)$;
- $T_{24} = (tional)$;
- $T_{25} = (ational)$;
- $T_{26} = (alize)$;
- $T_{27} = (icate, iciti, ical)$;
- $T_{28} = (ful, ness)$;
- $T_{29} = (ative)$;
- $T_{30} = (al, ance, ence, er, ic, able, ible, ant, ement, ment, ent, ism, ate, iti, ous, ive, ize)$;
- $T_{31} = (ion)$;
- $T_{32} = (e)$;
- $T_{33} = (l)$;

Далее приводится описание алгоритма аналитического выделения основы.

Шаг 0:

Найти самый длинный аффикс из списка T_1 и удалить его.

Шаг 1a:

Найти самый длинный аффикс из списка $T_2' = T_2 \cup T_3 \cup T_4 \cup T_5$.

Если найденный аффикс принадлежит списку T_2, то заменить его на ss.

Если найденный аффикс принадлежит списку T_3 и ему предшествует более одной буквы, то заменить его на i, иначе заменить на ie.

Если найденный аффикс принадлежит списку T_4 и ему предшествует гласная, но не стоит непосредственно перед найденным аффиксом, то удалить его.

Если найденный аффикс принадлежит списку T_5, то ничего не совершать.

Шаг 1b:

Найти самый длинный аффикс из списка $T_6' = T_6 \cup T_7$.

Если найденный аффикс принадлежит списку T_6 и находится в области R_1, то заменить его на ee.

Если найденный аффикс принадлежит списку T_7 и ему предшествует гласная (не непосредственно), то удалить его. При этом, если псевдооснова оканчивается на at, bl, iz, то добавить e; если псевдооснова оканчивается на удвоенную согласную, то удалить последнюю букву; если псевдооснова оканчивается на «короткий слог», то добавить e.

Шаг 1c:

Заменить букву y либо Y, стоящую на конце псевдоосновы, на i, если ей предшествует согласная, не являющаяся первой буквой псевдоосновы.

Шаг 2:

Найти самый длинный аффикс из списка $T_8' = T_8 \cup T_9 \cup T_{10} \cup T_{11} \cup T_{12} \cup T_{13} \cup T_{14} \cup T_{15} \cup T_{16} \cup T_{17} \cup T_{18} \cup T_{19} \cup T_{20} \cup T_{21} \cup T_{22} \cup T_{23}$.

Если найденный аффикс принадлежит списку T_8 и находится в области R_1, то заменить его на $tion$.

Если найденный аффикс принадлежит списку T_9 и находится в области R_1, то заменить его на $ence$.

Если найденный аффикс принадлежит списку T_{10} и находится в области R_1, то заменить его на $ance$.

Если найденный аффикс принадлежит списку T_{11} и находится в области R_1, то заменить его на $able$.

Если найденный аффикс принадлежит списку T_{12} и находится в области R_1, то заменить его на ent.

Если найденный аффикс принадлежит списку T_{13} и находится в области R_1, то заменить его на ize.

Если найденный аффикс принадлежит списку T_{14} и находится в области R_1, то заменить его на ate.

Если найденный аффикс принадлежит списку T_{15} и находится в области R_1, то заменить его на al.

Если найденный аффикс принадлежит списку T_{16} и находится в области R_1, то заменить его на ful.

Если найденный аффикс принадлежит списку T_{17} и находится в области R_1, то заменить его на ous.

Если найденный аффикс принадлежит списку T_{18} и находится в области R_1, то заменить его на ive.

Если найденный аффикс принадлежит списку T_{19} и находится в области R_1, то заменить его на ble.

Если найденный аффикс принадлежит списку T_{20} и находится в области R_1, а также найденному аффиксу предшествует l, то заменить его на og.

Если найденный аффикс принадлежит списку T_{21} и находится в области R_1, то заменить его на ful.

Если найденный аффикс принадлежит списку T_{22} и находится в области R_1, то заменить его на $less$.

Если найденный аффикс принадлежит списку T_{23} и находится в области R_1, а также ему предшествует LI -окончание, то удалить его.

Шаг 3:

Найти самый длинный аффикс из списка $T'_{24} = T_{24} \cup T_{25} \cup T_{26} \cup T_{27} \cup T_{28} \cup T_{29}$.

Если найденный аффикс принадлежит списку T_{24} и находится в области R_1, то заменить его на $tion$.

Если найденный аффикс принадлежит списку T_{25} и находится в области R_1, то заменить его на ate.

Если найденный аффикс принадлежит списку T_{26} и находится в области R_1, то заменить его на al.

Если найденный аффикс принадлежит списку T_{27} и находится в области R_1, то заменить его на ic.

Если найденный аффикс принадлежит списку T_{28} и находится в области R_1, то удалить его.

Если найденный аффикс принадлежит списку T_{29} и находится в области R_2, то удалить его.

Шаг 4:

Найти самый длинный аффикс из списка $T'_{30} = T_{30} \cup T_{31}$.

Если найденный аффикс принадлежит списку T_{30} и находится в области R_2, то удалить его.

Если найденный аффикс принадлежит списку T_{31} и находится в области R_2, а также ему предшествует s или t, то удалить его.

Шаг 5:

Найти самый длинный аффикс из списка $T'_{32} = T_{32} \cup T_{33}$.

Если найденный аффикс принадлежит списку T_{32} и находится в области R_2, то удалить его.

Если найденный аффикс принадлежит списку T_{33} и находится в области R_2, а также ему предшествует l, то удалить его.

В заключение нужно вернуть оставшиеся после преобразований буквы Y в нижний регистр. Этот шаг соответствует применению функций трансформации $f \in F_T$, обратных к тем, что были применены на предварительном этапе.

Полученный результат является искомой псевдоосновой анализируемой словоформы.

3.2. Болгарский язык

Морфологический строй современного болгарского языка характеризуется тесным переплетением аналитических и синтетических способов выражения грамматических значений. Аналитические способы наиболее последовательно используются в системе имени. Характерно использование переходных образований: аналитико-флективных (постпозитивный артикль) и аналитико-агглютинативных (степени сравнения).

Для болгарского языка в терминах обобщенного алгоритма функции проверки условий (функции ограничений) $f \in F_C$ будут представлять из себя проверку длины получаемой в данный момент псевдоосновы. Например:

- Псевдооснова имеет длину менее 4 букв;
- Псевдооснова имеет длину более 5 букв;

и т.д.

В соответствии со схемой обобщенного алгоритма анализа будут сформированы списки аффиксов $T_1, \dots, T_{12}$:

- $T_1 = (\text{ища})$;
- $T_2 = (\text{ият})$;
- $T_3 = (\text{ът}, \text{то}, \text{те}, \text{та}, \text{ия})$;
- $T_4 = (\text{ят})$;
- $T_5 = (\text{овци}, \text{ове})$
- $T_6 = (\text{еве})$;
- $T_7 = (\text{ища}, \text{та})$;
- $T_8 = (\text{ци}, \text{зи})$;
- $T_9 = (\text{си})$;
- $T_{10} = (\text{и})$;
- $T_{11} = (\text{я}, \text{а}, \text{о}, \text{е}, \text{ен})$;
- $T_{12} = (\text{ен})$;

Далее приводится описание алгоритма аналитического выделения основы.

Шаг 0:

Инициализировать псевдооснову поданной на вход алгоритму словоформой.

Шаг 1:

Найти аффикс из списка T_1. Если псевдооснова имеет длину более 5 букв, то удалить аффикс. Если полученная псевдооснова имеет длину менее 4 букв, то результат получен и выполнение алгоритма останавливается.

Шаг 2:

Найти аффикс из списка T_2. Если псевдооснова имеет длину более 6 букв, то удалить аффикс.

Найти аффикс из списка T_3. Если псевдооснова имеет длину более 5 букв, то удалить аффикс.

Найти аффикс из списка T_4. Если псевдооснова имеет длину более 4 букв, то удалить аффикс.

Шаг 3:

Найти аффикс из списка T_5. Если псевдооснова имеет длину более 6 букв, то удалить последние три буквы.

Найти аффикс из списка T_6. Если псевдооснова имеет длину более 6 букв, то заменить найденный аффикс на й.

Найти аффикс из списка T_7. Если псевдооснова имеет длину более 5 букв, то удалить найденный аффикс.

Найти аффикс из списка T_8. Если псевдооснова имеет длину более 5 букв, то заменить найденный аффикс либо на к (если аффикс был ци), либо на г (если аффикс был зи).

Найти аффикс из списка T_9. Если псевдооснова имеет длину более 4 букв, то заменить найденный аффикс на х.

Найти аффикс из списка T_{10}. Если псевдооснова имеет длину более 4 букв, то удалить найденный аффикс.

Шаг 4:

Найти аффикс из списка T_{11}. Если псевдооснова имеет длину более 3 букв, то удалить найденный аффикс.

Найти аффикс из списка T_{12}. Если псевдооснова имеет длину более 4 букв, то заменить найденный аффикс на н.

Если псевдооснова имеет длину более 5 и если ее можно представить в виде $L_1L_2 \ldots L_{n-2}\text{ъ}L_n$, где L_i — любая буква болгарского алфавита, то удалить ъ.

Полученный результат является искомой псевдоосновой анализируемой словоформы.

3.3. Датский язык

Морфологический строй датского языка сочетает признаки аналитического и флективного типов. Ведущими грамматическими средствами являются служебные слова и порядок слов. При этом важную роль продолжают играть внешняя и внутренняя флексии.

Для датского языка вводится понятие области R_1, которая определяется следующим образом. Пусть -ая буква от начала слова является согласной, предшествующая ей буква — гласной. Если -ая буква — это первая буква от начала слова, удовлетворяющая рассмотренному условию, то часть слова после нее называется областью R_1, за исключением случаев:

- буквы, удовлетворяющей данному условию (первая буква, соответствующая согласному, которой предшествует буква, соответствующая гласному), в слове нет — тогда область R_1 считается пустой;

- часть слова, предшествующая полученной области R_1, насчитывает менее трех символов (т.е. два символа) — тогда начало области R_1 смещается так, чтобы предшествующая ей часть слова содержала три символа.

Для датского языка следующие буквы являются гласными: $a, e, i, o, u, y,$ æ, å, ø; следующие — согласными: $b, c, d, f, g, h, j, k, l, m, n, p, q, r,$ $s, t, v, w, x, z.$

Характерной особенностью словоизменения имен существительных является прибавление суффикса «s» к концу слова. Поэтому в датском языке вводится понятие -окончаний. Ими являются: $a, b, c, d, f, g, h, j, k, l, m, n, o, p,$ $r, t, v, y, z,$ å.

Таким образом, для датского языка в терминах обобщенного алгоритма определяются следующие функции $f \in F_C$:

- Буква является -окончанием;
- Буква (или последовательность букв) располагается в области R_1.

В соответствии со схемой обобщенного алгоритма анализа будут сформированы списки аффиксов $T_1, \dots, T_6$:

- $T_1 = (hed, ethed, ered, e, erede, ende, erende, ene, erne, ere, en,$ $heden, eren, er, heder, erer, heds, es, endes, erendes, enes, ernes,$ $eres, ens, hedens, erens, ers, ets, erets, et, eret)$;
- $T_2 = (s)$;
- $T_3 = (gd, dt, gt, kt)$;
- $T_4 = (igst)$;
- $T_5 = (ig, lig, elig, els)$;
- $T_6 = (løst)$.

Далее приводится описание алгоритма аналитического выделения основы.

Шаг 1:

Найти самый длинный аффикс из списка T_1, и если он находится в области R_1, то удалить его.

Найти аффикс из списка T_2. Если аффикс найден и ему предшествует буква, определенная как -окончание (при этом S-окончание может не входить в область R_1), то удалить аффикс.

Шаг 2:

Найти самый длинный аффикс из списка T_3, и если он находится в области R_1, то удалить последнюю букву.

Шаг 3:

Найти аффикс из списка T_4 и заменить его на ig.

Найти самый длинный аффикс из списка $T_5' = T_5 \cup T_6$.

Если найденный аффикс принадлежит списку T_5 и входит в область R_1, то удалить его и перейти к Шагу 2.

Если аффикс из списка T_6 и он находится в области R_1, то заменить его на аффикс $løs$.

Шаг 4:

Если слово оканчивается на удвоенный согласный в области R_1, то необходимо удалить последний согласный — этот шаг соответствует применению функций трансформации $f \in F_T$ в схеме обобщенного алгоритма анализа словоформы.

Полученный результат является псевдоосновой анализируемой словоформы.

Кроме этого, в датском языке определен ряд слов, не подлежащих обработке алгоритмом выделения основ. Обычно в роли этих слов выступают предлоги, местоимения и союзы.

3.4. Нидерландский язык

Нидерландский язык является аналитическим языком со следами синтетизма. Синтетичность языка выражена флективностью со значительными элементами вторичной агглютинации.

Определим R_1 как область, начинающуюся после первой согласной буквы, следующей за гласной. Если такой согласной нет, то область пустая. R_2 — область, начинающаяся после первой согласной буквы, следующей за гласной в области R_1. Аналогично, если нет такой согласной, то область пустая.

R_1 и R_2 первоначально настраиваются по умолчанию, но затем R_1 увеличивается так, чтобы область до него содержала минимум 3 буквы.

Следующие буквы являются гласными: $a, e, i, o, u, y,$ è.

Алгоритм начинает работу с того, что устанавливает буквы y и i между гласными в верхний регистр. Этот этап соответствует применению функций трансформации $f \in F_T$ в схеме обобщенного алгоритма анализа словоформы.

Определим S -окончание как любую согласную букву, отличную от j.

Определим EN -окончание как любую согласную букву, при этом, если это буква m, то она не должна входить в состав аффикса gem.

Таким образом, для нидерландского языка в терминах обобщенного алгоритма определяются следующие функции $f \in F_C$:

- Буква является S -окончанием;
- Буква является EN -окончанием;
- Буква (или последовательность букв) располагается в области R_1;
- Буква (или последовательность букв) располагается в области R_2.

В соответствии со схемой обобщенного алгоритма анализа будут сформированы списки аффиксов $T_1, \dots, T_{10}$:

- $T_1 = (heden)$;
- $T_2 = (en, ene)$;
- $T_3 = (s, se)$;
- $T_4 = (e)$;
- $T_5 = (heid)$;
- $T_6 = (end, ing)$;
- $T_7 = (ig)$;
- $T_8 = (lijk)$;
- $T_9 = (baar)$;
- $T_{10} = (bar)$;

Далее приводится описание алгоритма аналитического выделения основы.

Шаг 1:

Найти самый длинный аффикс из списка $T_1' = T_1 \cup T_2 \cup T_3$.

Если найденный аффикс принадлежит списку T_1 и находится в области R_1, то заменить его если на $heid$.

Если найденный аффикс принадлежит списку T_2 и находится в области R_1, а также ему предшествует -окончание, то удалить его. Если после этого псевдооснова оканчивается на kk, dd или tt, то удалить последнюю букву.

Если найденный аффикс принадлежит списку T_3 и находится в области R_1, а также ему предшествует -окончание, то удалить его.

Шаг 2:

Найти аффикс из списка T_4. Если найденный аффикс находится в области R_1 и ему предшествует согласная, то удалить его. Если после этого псевдооснова оканчивается на kk, dd или tt, то удалить последнюю букву.

Шаг 3a:

Найти аффикс из списка T_5. Если найденный аффикс находится в области R_2 и ему не предшествует c, то удалить его. Найти аффикс en и, если он находится в области R_1 и ему предшествует -окончание, то удалить его. Если после этого псевдооснова оканчивается на kk, dd или tt, то удалить последнюю букву.

Шаг 3b:

Найти самый длинный аффикс из списка $T_6' = T_6 \cup T_7 \cup T_8 \cup T_9 \cup T_{10}$.

Если найденный аффикс принадлежит списку T_6 и находится в области R_2, то удалить его. Если ему предшествует аффикс ig, расположенный в области R_2, и ему не предшествует e, то удалить указанный аффикс; иначе, если после этого псевдооснова оканчивается на kk, dd или tt, то удалить последнюю букву.

Если найденный аффикс принадлежит списку T_7 и находится в области R_2, а также ему не предшествует аффикс e, то удалить его.

Если найденный аффикс принадлежит списку T_8 и находится в области R_2, то удалить его и перейти к шагу 2.

Если найденный аффикс принадлежит списку T_9 и находится в области R_2, то удалить его.

Если найденный аффикс принадлежит списку T_{10} и находится в области R_2, а также если на шаге 2 был удален аффикс, содержащий e, то удалить его.

Шаг 4:

Если после осуществления шагов 1-3 окончание псевдоосновы представимо в виде CVD, где C — это согласная буква, V — удвоенная гласная a, e, o или u, D — это согласная или I, то удалить одну из гласных в V.

В заключении нужно вернуть буквы I и Y в нижний регистр. Этот шаг соответствует применению функций трансформации $f \in F_T$, обратных к тем, что были применены на предварительном этапе.

Полученный результат является псевдоосновой анализируемой словоформы.

3.5. Норвежский язык (Нюнорск)

Морфологический строй норвежского языка сочетает приметы аналитического и синтетического (флективного) типов. Аналитичность выражена тем, что ведущими грамматическими средствами являются служебные слова и порядок слов, но известную роль играет также внешняя и внутренняя флексия.

Для норвежского языка вводится понятие области R_1, которая определяется следующим образом. Пусть -ая буква от начала слова является согласной, предшествующая ей буква — гласной. Если -ая буква — это первая буква от начала слова, удовлетворяющая рассмотренному условию, то часть слова после нее называется областью R_1, за исключением случаев:

- буквы, удовлетворяющей данному условию (первая буква, соответствующая согласному, которой предшествует буква,

соответствующая гласному), в слове нет — тогда область R_1 считается пустой;

- часть слова, предшествующая полученной области R_1, насчитывает менее трех символов (т.е. два символа) — тогда начало области R_1 смещается так, чтобы предшествующая ей часть слова содержала три символа.

Для норвежского языка следующие буквы являются гласными: $a, e, i, o, u, y,$ æ, å, ø; следующие — согласными: $b, c, d, f, g, h, j, k, l, m, n, p, q, r,$

$s, t, v, w, x, z.$

Характерной особенностью словоизменения имен существительных является прибавление суффикса «s» к концу слова. Поэтому вводится понятие -окончаний, которыми являются: $b, c, d, f, g, h, j, l, m, n, o, p, r, t, v, y,$

z и k, если ей не предшествует гласная.

Для норвежского языка в терминах обобщенного алгоритма определяются следующие функции $f \in F_C$:

- Буква является -окончанием;
- Буква (или последовательность букв) располагается в области R_1.

В соответствии со схемой обобщенного алгоритма анализа будут сформированы списки аффиксов $T_1, \dots, T_5$:

- $T_1 = (a, e, ede, ande, ende, ane, ene, hetene, en, heten, ar, er,$ $heter, as, es, edes, endes, enes, hetenes, ens, hetens, ers, ets,$ $et, het, ast)$;
- $T_2 = (s)$;
- $T_3 = (erte, ert)$;
- $T_4 = (dt, vt)$;
- $T_5 = (leg, eleg, ig, eig, lig, elig, els, lov, elov, slov, hetslov)$.

Далее приводится описание алгоритма аналитического выделения основы.

Шаг 1:

Найти самый длинный аффикс из списка T_1, и если он находится в области R_1, то удалить его.

Найти аффикс из списка T_2. Если аффикс найден и находится в области R_1, а также ему предшествует буква, определенная как -окончание (при этом S-окончание может не входить в область R_1), то удалить аффикс.

Найти аффикс из списка T_3 и если он находится в области R_1, то заменить его аффиксом er.

Шаг 2:

Найти аффикс из списка T_4, и если он находится в области R_1, то удалить последнюю букву.

Шаг 3:

Найти самый длинный аффикс из списка T_5, и если он входит в область R_1, то удалить его.

Полученный результат является искомой псевдоосновой анализируемой словоформы.

3.6. Шведский язык

В грамматическом строе шведского языка сочетаются элементы аналитизма и широкое использование флективных и агглютинативных показателей.

Для шведского языка вводится понятие области R_1, которая определяется следующим образом. Пусть -ая буква от начала слова является согласной, предшествующая ей буква — гласной. Если -ая буква — это первая буква от начала слова, удовлетворяющая рассмотренному условию, то часть слова после нее называется областью R_1, за исключением случаев:

- буквы, удовлетворяющей данному условию (первая буква, соответствующая согласному, которой предшествует буква, соответствующая гласному), в слове нет — тогда область R_1 считается пустой;
- часть слова, предшествующая полученной области R_1, насчитывает менее трех символов (т.е. два символа) — тогда начало области R_1 смещается так, чтобы предшествующая ей часть слова содержала три символа.

В шведском языке следующие буквы являются гласными: a, e, i, o, u, y, ä, å, ö; следующие — согласными: $b, c, d, f, g, h, j, k, l, m, n, p, q, r,$ s, t, v, w, x, z.

Характерной особенностью словоизменения имен существительных является прибавление суффикса «s» к концу слова. Поэтому вводится понятие -окончаний, которыми являются: $b, c, d, f, g, h, j, k, l, m, n, o, p, r, t, v, y$.

Для шведского языка в терминах обобщенного алгоритма определяются следующие функции $f \in F_C$:

- Буква является -окончанием;
- Буква (или последовательность букв) располагается в области R_1.

В соответствии со схемой обобщенного алгоритма анализа будут сформированы списки аффиксов $T_1, \dots, T_5$:

- $T_1 = (a, arna, erna, heterna, orna, ad, e, ade, ande, arne, are, aste,$ $en, anden, aren, heten, ern, ar, er, heter, or, as, arnas, ernas, ornas,$

$es, ades, andes, ens, arens, hetens, erns, at, andet, het, ast$);

- $T_2 = (s)$;
- $T_3 = (dd, gd, nn, dt, gt, kt, tt)$;
- $T_4 = (lig, ig, els)$;
- $T_5 = (löst, fullt)$.

Далее приводится описание алгоритма аналитического выделения основы.

Шаг 1:

Найти самый длинный аффикс из списка T_1, и если он находится в области R_1, то удалить его.

Найти аффикс из списка T_2. Если аффикс найден и ему предшествует буква, определенная как -окончание (при этом S-окончание может не входить в область R_1), то удалить аффикс.

Шаг 2:

Найти самый длинный аффикс из списка T_3, и если он находится в области R_1, то удалить последнюю букву.

Шаг 3:

Найти самый длинный аффикс из списка T_4, и если он входит в область R_1, то удалить его.

Найти аффикс из списка T_5, и если он находится в области R_1, то удалить последнюю букву найденного аффикса.

Полученный результат является искомой псевдоосновой анализируемой словоформы.

4. АЛГОРИТМЫ ВЫДЕЛЕНИЯ ОСНОВ II ТИПА

Требования к усвоению данного раздела учебного пособия:

Знать: иннновационные особенности различных видов печатных и электронных средств информации, основные технологические процессы производства печатных и электронных средств информации;

Уметь: использовать компьютерную технику в решении конкретных практических задач;

Владеть навыками: работы с прикладными программными средствами.

4.1. Белорусский язык

Белорусский язык относится к языкам флективного строя. Будучи по своим основным характеристикам синтетическим языком, он использует ряд аналитических структур разного типа, например, формы степеней сравнения наречий.

Следующие буквы являются гласными: а, о, е, ё, і, э, ы, у, ю, я.

Определим R_1 как область, начинающуюся после первой согласной буквы следующей за гласной. Если такой согласной нет, то область пустая. R_2 — область, начинающуюся после первой согласной буквы следующей за гласной в области R_1. Аналогично, если нет такой согласной, то область пустая.

R_1 и R_2 первоначально настраиваются по умолчанию, но затем R_1 увеличивается так, чтобы область до него содержала минимум 3 буквы.

Определим RV как область, начинающуюся после первой гласной буквы.

Для белорусского языка в терминах обобщенного алгоритма определяются следующие функции $f \in F_C$:

- Буква (или последовательность букв) располагается в области R_2;
- Буква (или последовательность букв) располагается в области RV.

В соответствии со схемой обобщенного алгоритма анализа будут сформированы списки аффиксов $T_1, \dots, T_6$:

- $T_1 =$ (шы, ючы, учы, ачы, ячы);
- $T_2 =$ (ся, си, сь);
- $T_3 =$ (ы, ае, ая, ага, ай, аму, ую, аю, ым, ымі, ых, і, яе, яя, ага, яй, яму, ою, ім, яю, ія, іх, імі);
- $T_4 =$ (у, еш, эш, аш, е, ем, эм, ам, еце, эце, аце, уць, ыш, іш, іць, ыць, ім, ым, іце, ыце, яць, аць, яце, ям);
- $T_5 =$ (а, я, ы, і, ямі, амі, у, ю, ой, ою, ёй, ёю, е, ам, ям, ах, ях, о, ом, ем, ём, ей, оў, ёў, аў, эй);
- $T_6 =$ (ост, ость);

Далее приводится описание алгоритма аналитического выделения основы.

Шаг 1:

Найти самый длинный аффикс из списка T_1, и если он находится в области RV, то удалить его и перейти к шагу 2.

Найти самый длинный аффикс из списка T_2, и если он находится в области RV, то удалить его и перейти к шагу 2.

Найти самый длинный аффикс из списка T_3, и если он находится в области RV, то удалить его и перейти к шагу 2.

Найти самый длинный аффикс из списка T_4, и если он находится в области RV, то удалить его и перейти к шагу 2.

Найти самый длинный аффикс из списка T_5, и если он находится в области RV, то удалить его и перейти к шагу 2.

Шаг 2:

Если псевдооснова оканчивается на и, то удалить последнюю букву.

Шаг 3:

Найти самый длинный аффикс из списка T_6, и если он находится в области R_2, то удалить его.

Шаг 4:

Если псевдооснова оканчивается на нн, то удалить последнюю букву.

Если псевдооснова оканчивается на ейш, ейше, то удалить соответствующий аффикс и удалить последнюю букву, если псевдооснова оканчивается на нн.

Если псевдооснова оканчивается на ь, то удалить последнюю букву.

Полученный результат является искомой псевдоосновой анализируемой словоформы.

4.2. Греческий язык

Морфологический строй современного греческого языка преимущественно синтетический (флективный). При этом в небольшой степени представлены аналитические формы, выраженные преимущественно в глаголах.

В греческом языке следующие буквы являются гласными: $\alpha, \varepsilon, \eta, \iota, o, \upsilon, \omega$.

Определим R_1 как часть слова за первой негласной (согласной или заглавной гласной), следующей за гласной, или конец слова, если такой негласной нет.

Для сербскохорватского языка в терминах обобщенного алгоритма определяются следующие функции $f \in F_C$:

- Буква (или последовательность букв) располагается в области R_1.

В соответствии со схемой обобщенного алгоритма анализа будут сформированы списки аффиксов $T_1, \ldots, T_{27}$:

- $T_1 = (\alpha\delta\varepsilon\sigma, \alpha\delta\omega\nu)$;
- $T_2 = (\varepsilon\delta\varepsilon\sigma, \varepsilon\delta\omega\nu)$;
- $T_3 = (ου\delta\varepsilon\sigma, ου\delta\omega\nu)$;
- $T_4 = (\varepsilon\omega\sigma, \varepsilon\omega\nu)$;
- $T_5 = (\iota\alpha, \iota ου, \iota\omega\nu)$;
- $T_6 = (\iota\kappa\alpha, \iota\kappa ο, \iota\kappa ου, \iota\kappa\omega\nu)$;
- $T_7 = (\alpha\gamma\alpha\mu\varepsilon, \eta\sigma\alpha\mu\varepsilon, \eta\kappa\alpha\mu\varepsilon, ου\sigma\alpha\mu\varepsilon, \eta\theta\eta\kappa\alpha\mu\varepsilon)$;
- $T_8 = (\alpha\mu\varepsilon)$;
- $T_9 = (\alpha\gamma\alpha\nu\varepsilon, \eta\sigma\alpha\nu\varepsilon, ο\tau\alpha\nu\varepsilon, \eta\kappa\alpha\nu\varepsilon, \iota ου\nu\tau\alpha\nu\varepsilon, \iota ο\nu\tau\alpha\nu\varepsilon, ου\nu\tau\alpha\nu\varepsilon,$ $\eta\theta\eta\kappa\alpha\nu\varepsilon, \iota ο\tau\alpha\nu\varepsilon, ο\nu\tau\alpha\nu\varepsilon, ου\sigma\alpha\nu\varepsilon)$;
- $T_{10} = (\alpha\nu\varepsilon)$;
- $T_{11} = (\eta\sigma\varepsilon\tau\varepsilon)$;
- $T_{12} = (\varepsilon\tau\varepsilon)$;
- $T_{13} = (ο\nu\tau\alpha\sigma, \omega\nu\tau\alpha\sigma)$;
- $T_{14} = (ο\mu\alpha\sigma\tau\varepsilon, \iota ο\mu\alpha\sigma\tau\varepsilon)$;
- $T_{15} = (\iota\varepsilon\sigma\tau\varepsilon)$;
- $T_{16} = (\varepsilon\sigma\tau\varepsilon)$;
- $T_{17} = (\eta\theta\eta\kappa\alpha, \eta\theta\eta\kappa\varepsilon, \eta\theta\eta\kappa\varepsilon\sigma)$;
- $T_{18} = (\eta\kappa\alpha, \eta\kappa\varepsilon, \eta\kappa\varepsilon\sigma)$;
- $T_{19} = (ου\sigma\alpha, ου\sigma\varepsilon, ου\sigma\varepsilon\sigma)$;
- $T_{20} = (\alpha\gamma\alpha, \alpha\gamma\varepsilon, \alpha\gamma\varepsilon\sigma)$;
- $T_{21} = (\eta\sigma\varepsilon, \eta\sigma\alpha, \eta\sigma ου)$;
- $T_{22} = (\eta\sigma\tau\varepsilon)$;
- $T_{23} = (\eta\sigma ου\nu\varepsilon, \eta\theta ου\nu\varepsilon, ου\nu\varepsilon)$;
- $T_{24} = (ου\mu\varepsilon, \eta\sigma ου\mu\varepsilon, \eta\theta ου\mu\varepsilon)$;
- $T_{25} = (\mu\alpha\tau\alpha, \mu\alpha\tau\omega\nu, \mu\alpha\tau ο\sigma)$;

- $T_{26} = (\iota\omicron\nu\tau\omicron\upsilon\sigma\alpha\nu, \iota\omicron\mu\alpha\sigma\tau\alpha\nu, \iota\omicron\sigma\alpha\sigma\tau\alpha\nu, \iota\omicron\upsilon\mu\alpha\sigma\tau\varepsilon, \omicron\nu\tau\omicron\upsilon\sigma\alpha\nu,$ $\iota\varepsilon\mu\alpha\sigma\tau\varepsilon, \iota\varepsilon\sigma\alpha\sigma\tau\varepsilon, \iota\omicron\mu\omicron\upsilon\nu\alpha, \iota\omicron\sigma\alpha\sigma\tau\varepsilon, \iota\omicron\sigma\omicron\upsilon\nu\alpha, \iota\omicron\upsilon\nu\tau\alpha\iota, \iota\omicron\upsilon\nu\tau\alpha\nu,$ $\eta\theta\eta\kappa\alpha\tau\varepsilon, \omicron\mu\alpha\sigma\tau\alpha\nu, \omicron\sigma\alpha\sigma\tau\alpha\nu, \omicron\upsilon\mu\alpha\sigma\tau\varepsilon, \iota\omicron\mu\omicron\upsilon\nu, \iota\omicron\nu\tau\alpha\nu, \iota\omicron\sigma\omicron\upsilon\nu,$ $\eta\theta\varepsilon\iota\tau\varepsilon, \eta\theta\eta\kappa\alpha\nu, \omicron\mu\omicron\upsilon\nu\alpha, \omicron\sigma\alpha\sigma\tau\varepsilon, \omicron\sigma\omicron\upsilon\nu\alpha, \omicron\upsilon\nu\tau\alpha\iota, \omicron\upsilon\nu\tau\alpha\nu, \omicron\upsilon\sigma\alpha\tau\varepsilon,$ $\alpha\gamma\alpha\tau\varepsilon, \iota\varepsilon\mu\alpha\iota, \iota\varepsilon\tau\alpha\iota, \iota\varepsilon\sigma\alpha\iota, \iota\omicron\tau\alpha\nu, \iota\omicron\upsilon\mu\alpha, \eta\theta\varepsilon\iota\sigma, \eta\theta\omicron\upsilon\nu, \eta\kappa\alpha\tau\varepsilon, \eta\sigma\alpha\tau\varepsilon,$ $\eta\sigma\omicron\upsilon\nu, \omicron\mu\omicron\upsilon\nu, \omicron\nu\tau\alpha\iota, \omicron\nu\tau\alpha\nu, \omicron\sigma\omicron\upsilon\nu, \omicron\upsilon\mu\alpha\iota, \omicron\upsilon\sigma\alpha\nu, \alpha\gamma\alpha\nu, \alpha\mu\alpha\iota, \alpha\sigma\alpha\iota,$ $\alpha\tau\alpha\iota, \varepsilon\iota\tau\varepsilon, \varepsilon\sigma\alpha\iota, \varepsilon\tau\alpha\iota, \eta\delta\varepsilon\sigma, \eta\delta\omega\nu, \eta\theta\varepsilon\iota, \eta\kappa\alpha\nu, \eta\sigma\alpha\nu, \eta\sigma\varepsilon\iota, \eta\sigma\varepsilon\sigma, \omicron\mu\alpha\iota,$ $\omicron\tau\alpha\nu, \alpha\varepsilon\iota, \varepsilon\iota\sigma, \eta\theta\omega, \eta\sigma\omega, \omicron\upsilon\nu, \omicron\upsilon\sigma, \alpha\nu, \alpha\sigma, \alpha\omega, \varepsilon\iota, \varepsilon\sigma, \eta\sigma, \omicron\iota, \omicron\sigma, \omicron\upsilon, \upsilon\sigma,$ $\omega\nu)$;
- $T_{27} = (\varepsilon\sigma\tau\varepsilon\rho, \varepsilon\sigma\tau\alpha\tau, \omicron\tau\varepsilon\rho, \omicron\tau\alpha\tau, \upsilon\tau\varepsilon\rho, \upsilon\tau\alpha\tau, \omega\tau\varepsilon\rho, \omega\tau\alpha\tau)$;

Далее приводится описание алгоритма аналитического выделения основы.

Шаг 1:

Найти самый длинный аффикс из списка T_1, и если он находится в области R_1, то удалить его. Если после этого псевдооснова оканчивается на аффикс из списка $\omicron\kappa, \mu\alpha\mu, \mu\alpha\nu, \mu\pi\alpha\mu\pi, \pi\alpha\tau\varepsilon\rho, \gamma\iota\alpha\gamma\iota, \nu\tau\alpha\nu\tau, \kappa\upsilon\rho, \theta\varepsilon\iota, \pi\varepsilon\theta\varepsilon\rho$, то приписать к псевдооснове $\alpha\delta$.

Шаг 2:

Найти самый длинный аффикс из списка T_2, и если он находится в области R_1, то удалить его. Если после этого псевдооснова оканчивается на аффикс из списка $\omicron\pi, \iota\pi, \varepsilon\mu\pi, \upsilon\pi, \gamma\eta\pi, \delta\alpha\pi, \kappa\rho\alpha\sigma\pi, \mu\iota\lambda$, то приписать к псевдооснове $\varepsilon\delta$.

Шаг 3:

Найти самый длинный аффикс из списка T_3, и если он находится в области R_1, то удалить его. Если после этого псевдооснова оканчивается на аффикс из списка

$\alpha\rho\kappa, \kappa\alpha\lambda\iota\alpha\kappa, \pi\varepsilon\tau\alpha\lambda, \lambda\iota\chi, \pi\lambda\varepsilon\xi, \sigma\kappa, \sigma, \varphi\lambda, \varphi\rho, \beta\varepsilon\lambda, \lambda\sigma\upsilon\lambda, \chi\nu, \sigma\pi, \tau\rho\alpha\gamma, \varphi\varepsilon,$ то приписать к псевдооснове $\sigma\upsilon\delta$.

Шаг 4:

Найти самый длинный аффикс из списка T_4, и если он находится в области R_1, то удалить его. Если после этого псевдооснова оканчивается на аффикс из списка $\theta, \delta, \varepsilon\lambda, \gamma\alpha\lambda, \nu, \pi, \iota\delta, \pi\alpha\rho$, то приписать к псевдооснове ε.

Шаг 5:

Найти самый длинный аффикс из списка T_5, и если он находится в области R_1, то удалить его. Если после этого псевдооснова оканчивается на гласную, то приписать к псевдооснове ι.

Шаг 6:

Найти самый длинный аффикс из списка T_6, и если он находится в области R_1, то удалить его. Если после этого псевдооснова оканчивается на аффикс из списка $\alpha\lambda, \alpha\delta, \varepsilon\nu\delta, \alpha\mu\alpha\nu, \alpha\mu\mu\sigma\chi\alpha\lambda, \eta\theta, \alpha\nu\eta\theta, \gamma\varepsilon\rho, \varepsilon\xi\omega\delta, \kappa\alpha\lambda\pi,$

$\kappa\alpha\lambda\lambda\iota\nu, \kappa\alpha\tau\alpha\delta, \mu\sigma\upsilon\lambda, \mu\pi\alpha\nu, \mu\pi\alpha\gamma\iota\alpha\tau, \mu\pi\sigma\lambda, \mu\pi\sigma\sigma, \nu\iota\tau, \xi\iota\kappa, \sigma\upsilon\nu\sigma\mu\eta\lambda, \pi\varepsilon\tau\sigma, \pi\iota\tau\sigma,$

$\pi\iota\kappa\alpha\nu\tau, \pi\lambda\iota\alpha\tau\sigma, \pi\sigma\sigma\tau\varepsilon\lambda\nu, \pi\rho\omega\tau\sigma\delta, \sigma\varepsilon\rho\tau, \sigma\upsilon\nu\alpha\delta, \tau\sigma\alpha\mu, \upsilon\pi\sigma\delta, \varphi\iota\lambda\sigma\nu, \varphi\upsilon\lambda\sigma\delta, \chi\alpha\sigma,$
или на гласную, то приписать к псевдооснове $\iota\kappa$.

Шаг 7:

Если на данном шаге псевдооснова представляет из себя $\alpha\gamma\alpha\mu\varepsilon$, то вернуть в качестве результата работы алгоритма псевдооснову $\alpha\gamma\alpha\mu$.

Найти самый длинный аффикс из списка T_7, и если он находится в области R_1, то удалить его.

Найти самый длинный аффикс из списка T_8, и если он находится в области R_1, то удалить его. Если после этого псевдооснова оканчивается на аффикс из списка $αναπ, αποθ, αποκ, αποστ, βουβ, ξεθ, ουλ, πεθ, πικρ, ποτ,$

$σιχ, χ$, то приписать к псевдооснове $αμ$.

Шаг 8:

Найти самый длинный аффикс из списка T_9, и если он находится в области R_1, то удалить его. Если после этого псевдооснова оканчивается на аффикс из списка $τρ, τσ$, то приписать к псевдооснове $αγαν$.

Найти самый длинный аффикс из списка T_{10}, и если он находится в области R_1, то удалить его. Если после этого псевдооснова оканчивается на аффикс из списка $βετερ, βουλκ, βραχμ, γ, δραδουμ, θ, καλπουζ, καστελ,$

$κορμορ, λαοπλ, μωαμεθ, μ, μουσουλμ, ν, ουλ, π, πελεκ, πλ, πολισ, πορτολ,$

$σαρακατσ, σουλτ, τσαρλατ, ορφ, τσιγγ, τσοπ, φωτοστεφ, χ, ψυχοπλ, αγ, ορφ,$

$γαλ, γερ, δεκ, διπλ, αμερικαν, ουρ, πιθ, πουριτ, σ, ζωντ, ικ, καστ, κοπ, λιχ,$

$λουθηρ, μαιντ, μελ, σιγ, σπ, στεγ, τραγ, τσαγ, φ, ερ, αδαπ, αθιγγ, αμηχ, ανικ,$

$ανοργ, απηγ, απιθ, ατσιγγ, βασ, βασκ, βαθυγαλ, βιομηχ, βραχυκ, διατ, διαφ,$

$ενοργ, θυσ, καπνοβιομηχ, καταγαλ, κλιβ, κοιλαρφ, λιβ, μεγλοβιομηχ,$

$μικροβιομηχ, νταβ, ξηροκλιβ, ολιγοδαμ, ολογαλ, πενταρφ, περηφ, περιτρ,$

$πλατ, πολυδαπ, πολυμηχ, στεφ, ταβ, τετ, υπερηφ, υποκοπ, χαμηλοδαπ, ψηλο$ или на гласную, то приписать к псевдооснове $αν$.

Шаг 9:

Найти самый длинный аффикс из списка T_{11}, и если он находится в области R_1, то удалить его.

Найти самый длинный аффикс из списка T_{12}, и если он находится в области R_1, то удалить его. Если после этого псевдооснова оканчивается на аффикс из списка $\alpha\beta\alpha\rho, \beta\varepsilon\nu, \varepsilon\nu\alpha\rho, \alpha\beta\rho, \alpha\delta, \alpha\theta, \alpha\nu, \alpha\pi\lambda, \beta\alpha\rho\sigma\nu, \nu\tau\rho, \sigma\kappa, \kappa\sigma\pi,$

$\mu\pi\sigma\rho, \nu\iota\varphi, \pi\alpha\gamma, \pi\alpha\rho\alpha\kappa\alpha\lambda, \sigma\varepsilon\rho\pi, \sigma\kappa\varepsilon\lambda, \sigma\upsilon\rho\varphi, \tau\sigma\kappa, \upsilon, \delta, \varepsilon\mu, \theta\alpha\rho\rho, \theta, \sigma\delta, \alpha\iota\rho, \varphi\sigma\rho,$

$\tau\alpha\theta, \delta\iota\alpha\theta, \sigma\chi, \varepsilon\nu\delta, \varepsilon\upsilon\rho, \tau\iota\theta, \upsilon\pi\varepsilon\rho\theta, \rho\alpha\theta, \varepsilon\nu\theta, \rho\sigma\theta, \sigma\theta, \pi\upsilon\rho, \alpha\iota\nu, \sigma\upsilon\nu\delta, \sigma\upsilon\nu, \sigma\upsilon\nu\theta,$

$\chi\omega\rho, \pi\sigma\nu, \beta\rho, \kappa\alpha\theta, \varepsilon\upsilon\theta, \varepsilon\kappa\theta, \nu\varepsilon\tau, \rho\sigma\nu, \alpha\rho\kappa, \beta\alpha\rho, \beta\sigma\lambda, \omega\varphi\varepsilon\lambda$ или на гласную, то приписать к псевдооснове $\varepsilon\tau$.

Шаг 10:

Найти самый длинный аффикс из списка T_{13}, и если он находится в области R_1, то удалить его. Если после этого псевдооснова оканчивается на аффикс из списка $\alpha\rho\chi$, то приписать к псевдооснове $\sigma\nu\tau$; если оканчивается на $\kappa\rho\varepsilon$, то приписать к псевдооснове $\omega\nu\tau$.

Шаг 11:

Найти самый длинный аффикс из списка T_{14}, и если он находится в области R_1, то удалить его. Если после этого псевдооснова оканчивается на аффикс из списка $\sigma\nu$, то приписать к псевдооснове $\sigma\mu\alpha\sigma\tau$.

Шаг 12:

Найти самый длинный аффикс из списка T_{15}, и если он находится в области R_1, то удалить его. Если после этого псевдооснова оканчивается на аффикс из списка $\pi, \alpha\pi, \sigma\upsilon\mu\pi, \alpha\sigma\upsilon\mu\pi, \alpha\kappa\alpha\tau\alpha\pi, \alpha\mu\varepsilon\tau\alpha\mu\varphi$, то приписать к псевдооснове $\iota\varepsilon\sigma\tau$.

Найти самый длинный аффикс из списка T_{16}, и если он находится в области R_1, то удалить его. Если после этого псевдооснова оканчивается на аффикс из списка $\alpha\lambda, \alpha\rho, \varepsilon\kappa\tau\varepsilon\lambda, \zeta, \mu, \xi, \pi\alpha\rho\alpha\kappa\alpha\lambda, \pi\rho\sigma, \nu\iota\sigma$, то приписать к псевдооснове $\varepsilon\sigma\tau$.

Шаг 13:

Найти самый длинный аффикс из списка T_{17}, и если он находится в области R_1, то удалить его.

Найти самый длинный аффикс из списка T_{18}, и если он находится в области R_1, то удалить его. Если после этого псевдооснова оканчивается на аффикс из списка $διαθ, θ, παρακαταθ, προσθ, συνθ, σκωλ, σκουλ, ναρθ, σφ,$

$οθ, πιθ$, то приписать к псевдооснове $ηκ$.

Шаг 14:

Найти самый длинный аффикс из списка T_{19}, и если он находится в области R_1, то удалить его. Если после этого псевдооснова оканчивается на аффикс из списка $φαρμακ, χαδ, αγκ, αναρρ, βρομ, εκλιπ, λαμπιδ, λεχ, μ, πατ,$

$ρ, λ, μεδ, μεσαζ, υποτειν, αμ, αιθ, ανηκ, δεσποζ, ενδιαφερ, δε, δευτερευ,$

$καθαρευ, πλε, τσα, ποδαρ, βλεπ, πανταχ, φρυδ, μαντιλ, μαλλ, κυματ, λαχ, ληγ,$

$φαγ, ομ, πρωτ$ или на гласную, то приписать к псевдооснове $ουσ$.

Шаг 15:

Найти самый длинный аффикс из списка T_{20}, и если он находится в области R_1, то удалить его. Если после этого псевдооснова оканчивается на аффикс из списка $αβαστ, πολυφ, αδηφ, παμφ, ρ, ασπ, αφ, αμαλ, αμαλλι,$

$ανυστ, απερ, ασπαρ, αχαρ, δερβεν, δροσοπ, ξεφ, νεοπ, νομοτ, ολοπ, ομοτ,$

$προστ, προσωποπ, συμπ, συντ, τ, υποτ, χαρ, αειπ, αιμοστ, ανυπ, αποτ, αρτιπ,$

$διατ, εν, επιτ, κροκαλοπ, σιδηροπ, λ, ναυ, ουλαμ, ουρ, π, τρ, μ, οφ, πελ, χορτ, λλ,$

$σφ, ρπ, φρ, πρ, λοχ, σμην$ и не оканчивается на аффикс из списка $ψοφ, ναυλοχ, κολλ$, то приписать к псевдооснове $αγ$.

Шаг 16:

Найти самый длинный аффикс из списка T_{21}, и если он находится в области R_1, то удалить его. Если после этого псевдооснова оканчивается на аффикс из списка $\nu, \chi\varepsilon\rho\sigma\sigma\nu, \delta\omega\delta\varepsilon\kappa\alpha\nu, \varepsilon\rho\eta\mu\sigma\nu, \mu\varepsilon\gamma\alpha\lambda\sigma\nu, \varepsilon\pi\tau\alpha\nu$, то приписать к псевдооснове $\eta\sigma$.

Шаг 17:

Найти самый длинный аффикс из списка T_{22}, и если он находится в области R_1, то удалить его. Если после этого псевдооснова оканчивается на аффикс из списка $\alpha\sigma\beta, \sigma\beta, \alpha\chi\rho, \chi\rho, \alpha\pi\lambda, \alpha\varepsilon\iota\mu\nu, \delta\upsilon\sigma\chi\rho, \varepsilon\upsilon\chi\rho, \kappa\sigma\iota\nu\sigma\chi\rho, \pi\alpha\lambda\iota\mu\psi$, то приписать к псевдооснове $\eta\sigma\tau$.

Шаг 18:

Найти самый длинный аффикс из списка T_{23}, и если он находится в области R_1, то удалить его. Если после этого псевдооснова оканчивается на аффикс из списка $\nu, \rho, \sigma\pi\iota, \sigma\tau\rho\alpha\beta\sigma\mu\sigma\upsilon\tau\sigma, \kappa\alpha\kappa\sigma\mu\sigma\upsilon\tau\sigma, \varepsilon\xi\omega\nu$, то приписать к псевдооснове $\sigma\upsilon\nu$.

Шаг 19:

Найти самый длинный аффикс из списка T_{24}, и если он находится в области R_1, то удалить его. Если после этого псевдооснова оканчивается на аффикс из списка $\pi\alpha\rho\alpha\sigma\sigma\upsilon\sigma, \varphi, \chi, \omega\rho\iota\sigma\pi\lambda, \alpha\zeta, \alpha\lambda\lambda\sigma\sigma\upsilon\sigma, \alpha\sigma\sigma\upsilon\sigma$, то приписать к псевдооснове $\sigma\upsilon\mu$.

Шаг 20:

Найти самый длинный аффикс из списка T_{25}, и если он находится в области R_1, то заменить его на $\mu\alpha$.

Шаг 21:

Найти самый длинный аффикс из списка T_{26}, и если он находится в области R_1, то удалить его

Шаг 22:

Найти самый длинный аффикс из списка T_{27}, и если он находится в области R_1, то удалить его.

Полученный результат является искомой псевдоосновой анализируемой словоформы.

4.3. Исландский язык

Исландский язык является языком флективного типа. Он обладает наиболее развитой среди всех германских языков морфологической системой. Наиболее продуктивными морфологическими средствами являются внешняя и внутренняя флексии. Большую роль играют служебные слова.

В исландском языке следующие буквы являются гласными: $a, e, i, o, u,$ ö, á, é, í, ó, ú, $y,$ ý.

Определим R_1 как часть слова за первой негласной (согласной или заглавной гласной), следующей за гласной, или конец слова, если такой негласной нет. R_2 — часть слова за первой негласной, следующей за гласной, находящейся в R_1, или конец слова, если такой негласной нет.

Для исландского языка в терминах обобщенного алгоритма определяются следующие функции $f \in F_C$:

- Буква (или последовательность букв) располагается в области R_1.

В соответствии со схемой обобщенного алгоритма анализа будут сформированы списки аффиксов T_1, T_2:

- $T_1 = (s, ar, jar, ir, ur, r, a, u, ll, nn, inn, llinn, urinn, nninn, arnir, in, an, irnar, urnar, ið, ta, te, the, acha, eacha, anna, eanna, um, jum, rum, ja)$;
- $T_2 = (ri, ari, ara, astur, stur, ust, ast, asti, asta, lega)$;

Далее приводится описание алгоритма аналитического выделения основы.

Шаг 1:

Найти самый длинный аффикс из списка $T_1' = T_1 \cup T_2$, и если он находится в области R_1, то удалить его.

Шаг 2:

Если псевдооснова оканчивается на гласную букву, то удалить последнюю букву.

Полученный результат является искомой псевдоосновой анализируемой словоформы.

4.4. Испанский язык

Испанский язык относится к смешанному флективно-аналитическому типу с элементами агглютинации. Флективный способ выражения грамматических значений характерен для глагола, в то время как в имени преобладает агглютинация. В глагольной системе, наряду с флективными, имеется большое количество аналитических форм.

Буквы в испанском языке включают следующие формы с акцентом: á, é, í, ó, ú, ü, ñ.

Следующие буквы гласные: a, e, i, o, u, á, é, í, ó, ú, ü.

Определим R_1 как часть слова за первой негласной (согласной или заглавной гласной), следующей за гласной, или конец слова, если такой негласной нет. R_2 — часть слова за первой негласной, следующей за гласной, находящейся в R_1, или конец слова, если такой негласной нет.

Область RV определяется следующим образом: если в слове вторая буква согласная, то RV — это часть слова после следующей гласной; если первые две буквы гласные, то RV — часть слова после следующей согласной буквы, иначе, если первая — согласная, а вторая — гласная, то RV — это область слова после третьей буквы. Но в данных условиях RV не может начинаться в конце слова.

Для испанского языка в терминах обобщенного алгоритма определяются следующие функции $f \in F_C$:

- Буква (или последовательность букв) располагается в области R_1;
- Буква (или последовательность букв) располагается в области R_2;
- Буква (или последовательность букв) располагается в области RV.

В соответствии со схемой обобщенного алгоритма анализа будут сформированы списки аффиксов $T_1, \dots, T_{15}$:

- $T_1 = (me, se, sela, selo, selas, selos, la, le, lo, las, les, los, nos)$;
- $T_2 = (anza, anzas, ico, ica, icos, icas, ismo, ismos, able, ables, ible, ibles, ista, istas, oso, osa, osos, osas, amiento, amientos, imiento, imientos)$;
- $T_3 = (adora, ador, ación, adoras, adores, aciones, ante, antes, ancia, ancias)$;
- $T_4 = (logía, logías)$;
- $T_5 = (ución, uciones)$;
- $T_6 = (encia, encias)$;
- $T_7 = (amente)$;
- $T_8 = (mente)$;

- $T_9 = (idad, idades)$;
- $T_{10} = (iva, ivo, ivas, ivos)$;
- $T_{11} = (ya, ye, yan, yen, yeron, yendo, yo,$ yó$, yas, yes, yais, yamos)$;
- $T_{12} = (en, es,$ é$is, emos)$;
- $T_{13} = ($ar$ían, ar$ías$, ar$án$, ar$ás$, ar$íais$, ar$ía$, ar$éis$, ar$íamos$, $aremos$, ar$á$, ar$é$, er$ían$, er$ías$, er$án$, er$ás$, er$íais$, er$ía$, er$éis$, er$íamos$, $eremos$, er$á$, er$é$, ir$ían$, ir$ías$, ir$án$, ir$ás$, ir$íais$, ir$ía$, ir$éis$, ir$íamos$, $iremos$, ir$á$, ir$é$, aba, ada, ida, ía, ara, $iera$, ad, ed, id, ase, $iese$, $aste$, $iste$, an, $aban$, ían, $aran$, $ieran$, $asen$, $iesen$, $aron$, $ieron$, ado, ido, $ando$, $iendo$, i$ó$, ar, er, ir, as, $abas$, $adas$, $idas$, ías, $aras$, $ieras$, $ases$, $ieses$, ís, áis, $abais$, íais, $arais$, $ierais$, $aseis$, $ieseis$, $asteis$, $isteis$, $ados$, $idos$, $amos$, á$bamos$, í$amos$, $imos$, á$ramos$, i$éramos$, i$ésemos$, á$semos$);
- $T_{14} = (os, a, o,$ á, í, ó$)$;
- $T_{15} = (e,$ é$)$;

Далее приведем описание работы алгоритма по шагам. Шаги 0 и 1 выполняются всегда.

Шаг 0:

Найти самый длинный аффикс из списка T_1.

Если найденный аффикс принадлежит списку T_1 и находится в области RV, то удалить его, если перед найденным аффиксом стоит один из аффкисов: i$éndo$, ándo, ár, ér, ír, $ando$, $iendo$, ar, er, ir, либо $yendo$, перед которым стоит u.

Шаг 1:

Найти самый длинный аффикс из списка $T_2' = T_2 \cup T_3 \cup T_4 \cup T_5 \cup T_6 \cup T_7 \cup T_8 \cup T_9 \cup T_{10}$.

Если найденный аффикс принадлежит списку T_2 и находится в области R_2, то удалить его.

Если найденный аффикс принадлежит списку T_3 и находится в области R_2, то удалить его. При этом, если аффиксу предшествует ic, то удалить ic, если данный аффикс находится в области R_2;

Если найденный аффикс принадлежит списку T_4 и находится в области R_2, то заменить его на log.

Если найденный аффикс принадлежит списку T_5 и находится в области R_2, то заменить его на u.

Если найденный аффикс принадлежит списку T_6 и находится в области R_2, то заменить его на $ente$.

Если найденный аффикс принадлежит списку T_7 и находится в области R_1, то удалить его. При этом, если данному аффиксу предшествует iv, то удалить iv, если тот находится в области R_2 (если аффиксу предшествует at, то удалить at, если тот находится в области R_2); если найденный аффикс следует за os, ic или ad, то удалить os, ic или ad, если тот находится в области R_2.

Если найденный аффикс принадлежит списку T_8 и находится в области R_2, то удалить его. При этом, если найденный аффикс следует за $ante, able$ или $ible$, то удалить $ante, able$ или $ible$, если тот находится в области R_2.

Если найденный аффикс принадлежит списку T_9 и находится в области R_2, то удалить его. При этом, если найденный аффикс следует за $abil, ic$ или iv, то удалить $abil, ic$ или iv, если тот находится в области R_2.

Если найденный аффикс принадлежит списку T_{10} и находится в области R_2, то удалить его. При этом, если найденный аффикс следует за at, то удалить at, если тот находится в области R_2.

Если ни один аффикс не был удален на Шаге 1, перейти к выполнению Шага 2a.

Шаг 2a:

Найти самый длинный аффикс из списка T_{11}.

Если найденный аффикс находится в области RV и следует за u, то удалить его. При этом предшествующая найденному аффиксу u может не находиться в области RV.

Если на Шаге 2a не был удален ни один аффикс, то перейти к Шагу 2b.

Шаг 2b:

Найти самый длинный аффикс из списка $T'_{12} = T_{12} \cup T_{13}$.

Если найденный аффикс принадлежит списку T_{12} и находится в области RV, то удалить его, если он следует за gu. При этом также удалить u из аффикса gu.

Если найденный аффикс принадлежит списку T_{13} и находится в области RV, то удалить его.

Шаг 3:

Найти самый длинный аффикс из списка $T'_{14} = T_{14} \cup T_{15}$.

Если найденный аффикс принадлежит списку T_{14} и находится в области RV, то удалить его.

Если найденный аффикс принадлежит списку T_{15} и находится в области RV, то удалить его, если он следует за gu. При этом также удалить u из аффикса gu.

В заключение нужно преобразовать все формы с акцентом к соответствующим им формам без акцета. Полученный результат является искомой псевдоосновой анализируемой словоформы.

4.5. Итальянский язык

Итальянский язык относится к смешанному, синтетическо-аналитическому типу. Категории рода и числа выражаются флективно, причем для итальянского языка характерен синтез категорий в одной флексии. Падежи в системе имени отсутствуют, падежные значения выражаются аналитически.

В итальянском языке определены следующие буквы с акцентом: á, é, í, ó, ú, à, è, ì, ò, ù.

Гласными буквами являются: $a, e, i, o, u,$ à, è, ì, ò, ù.

В начале работы необходимо заменить все акуты на грависы, поместить u после q, а буквы u, i между гласными перевести в верхний регистр. Этот этап соответствует применению функций трансформации $f \in F_T$ в схеме обобщенного алгоритма анализа словоформы.

Определим область RV следующим образом: если в слове вторая буква согласная, то RV — это часть слова после следующей гласной; если первые две буквы гласные, то RV — часть слова после следующей согласной буквы, иначе, если первая — согласная, а вторая — гласная, то RV — это область слова после третьей буквы. Но в данных условиях RV не может начинаться в конце слова.

Определим R_1 как часть слова за первой негласной (согласной или заглавной гласной), следующей за гласной, или конец слова, если такой негласной нет. R_2 — часть слова за первой негласной, следующей за гласной, находящейся в R_1, или конец слова, если такой негласной нет. R_1 может содержать в себе RV, и RV может содержать в себе R_1.

Для итальянского языка в терминах обобщенного алгоритма определяются следующие функции $f \in F_C$:

- Буква (или последовательность букв) располагается в области R_1;
- Буква (или последовательность букв) располагается в области R_2;
- Буква (или последовательность букв) располагается в области RV.

В соответствии со схемой обобщенного алгоритма анализа будут сформированы списки аффиксов $T_1, \dots, T_{12}$:

- $T_1 = (ci, gli, la, leli, lo, mi, ne, si, ti, vi, sene, gliela, gliele, glieli, glielo, gliene, mela, mele, meli, melo, mene, tela, tele, teli, telo, tene, cela, cele, celi, celo, cene, vela, vele, veli, velo, vene)$;
- $T_2 = (anza, anze, ico, ici, ica, ice, iche, ichi, ismo, ismi, abile, abili, ibile, ibili, ista, iste, isti, istà, istè, istì, oso, osi, osa, ose, mente, atrice, atrici, ante, anti)$;
- $T_3 = (azione, azioni, atore, atori)$;
- $T_4 = (logia, logie)$;
- $T_5 = (uzione, uzioni, usione, usioni)$;
- $T_6 = (enza, enze)$;
- $T_7 = (amento, amenti, imento, imenti)$;
- $T_8 = (amente)$;
- $T_9 = (ità)$;
- $T_{10} = (ivo, ivi, iva, ive)$;
- $T_{11} = (ammo, ando, ano, are, arono, asse, assero, assi, assimo, ata, ate, ati, ato, ava, avamo, avano, avate, avi, avo, emmo, enda, ende, endi, endo, erà, erai, eranno, ere, erebbe, erebbero, erei, eremmo, eremo, ereste, eresti, erete, erò, erono, essero, ete, eva, evamo, evano, evate, evi, evo, Yamo, iamo, immo, irà, irai, iranno, ire, irebbe, irebbero, irei, iremmo, iremo, ireste, iresti, irete, irò, irono, isca, iscano, isce, isci, isco, iscono, issero, ita, ite, iti, ito, iva, ivamo,$

$ivano, ivate, ivi, ivo, ono, uta, ute, uti, uto, ar, ir$);

- T_{12} = (a, e, i, o, à, è, ì, ò);

Далее приведено описание алгоритма аналитического выделения основы (шаги алгоритма 0 и 1 выполняются всегда).

Шаг 0:

Найти самый длинный аффикс из списка T_1 в области RV, которому предшествует:

- $ando, endo$ — в этом случае удалить аффикс;
- ar, er, ir — в этом случае заменить найденный аффикс на e.

Шаг 1:

Найти самый длинный аффикс из списка $T_2' = T_2 \cup T_3 \cup T_4 \cup T_5 \cup T_6 \cup T_7 \cup T_8 \cup T_9 \cup T_{10}$.

Если найденный аффикс принадлежит списку T_2 и находится в области R_2, то удалить его.

Если найденный аффикс принадлежит списку T_3 и если он находится в области R_2, то удалить его. Если перед найденным аффиксом находится аффикс ic, то удалить его, если он находится в области R_2.

Если найденный аффикс принадлежит списку T_4 и если он находится в области R_2, то заменить его на log.

Если найденный аффикс принадлежит списку T_5 и если он находится в области R_2, то заменить его на u.

Если найденный аффикс принадлежит списку T_6 и если он находится в области R_2, то заменить его на $ente$.

Если найденный аффикс принадлежит списку T_7 и если он находится в области RV, то удалить его.

Если найденный аффикс принадлежит списку T_8 и если он находится в области R_1, то удалить его. Если найденному аффиксу предшествует аффикс iv, то удалить его, если он находится в области R_2 (если при этом iv предшествует аффикс at, то удалить его, если тот находится в области R_2); если суффиксу предшествует один из аффиксов os, ic или $abil$, то удалить его, если тот находится в области R_2.

Если найденный аффикс принадлежит списку T_9 и если он находится в области R_2, то удалить его. Если найденному аффиксу предшествует один из аффиксов $abil, ic$ или iv, то удалить его, если тот находится в области R_2.

Если найденный аффикс принадлежит списку T_{10} и если он находится в области R_2, то удалить его. Если найденному аффиксу предшествует аффикс at, то удалить at, если он находится в области R_2 (если при этом at предшествует аффикс ic, то удалить ic, если тот находится в области R_2).

Если ни один из аффиксов не был удален на Шаге 1, то перейти к Шагу 2. Иначе — перейти к Шагу 3a.

Шаг 2:

Найти самый длинный аффикс из списка T_{11}, и если он находится в области RV, то удалить его.

Шаг 3a:

Удалите аффикс из списка T_{12}, если он находится в области RV, и предшествующую ему букву i, если она находится в области RV.

Шаг 3b:

Заменить последний аффикс ch (или gh) на c (или g), если он находится в области RV.

В заключение нужно вернуть I и U в нижний регистр — этот шаг соответствует применению функций трансформации $f \in F_T$, обратных к тем, что были применены на предварительном этапе.

Полученный результат является искомой псевдоосновой анализируемой словоформы.

4.6. Каталанский язык

Каталанский язык является смешанным синтетическо-аналитическим языком. Флективное словоизменение имени и глагола сопровождается многообразными морфонологическими чередованиями. В то же время в глагольной системе имеется большое количество аналитических форм.

Буквы в каталанском языке включают следующие формы с акцентом: á, à, é, è, í, ï, ó, ò, ú, ü.

Следующие буквы гласные: $a, e, i, o, u,$ á, à, é, è, í, ï, ó, ò, ú, ü.

Определим R_1 как часть слова за первой негласной (согласной или заглавной гласной), следующей за гласной, или конец слова, если такой негласной нет. R_2 — часть слова за первой негласной, следующей за гласной, находящейся в R_1, или конец слова, если такой негласной нет.

Для каталанского языка в терминах обобщенного алгоритма определяются следующие функции $f \in F_C$:

- Буква (или последовательность букв) располагается в области R_1;
- Буква (или последовательность букв) располагается в области R_2;

В соответствии со схемой обобщенного алгоритма анализа будут сформированы списки аффиксов $T_1, \dots, T_{10}$:

- $T_1 = ('s, 'hi, 'ho, 'l, 'ls, -ls, -la, -les, -li, vos, se, nos, -nos, -us, us,$

$'n, 'ns, -n, -ns, 'm, -me, -m, -te, 't, li, lo, los, me, sela, selo, selas, selos, le, la, las, les, ens, ho, hi$);

- $T_2 = (ar, atge, formes, icte, ictes, ell, ells, ella, es, es, esc, essa, et, ets, eta, eres, eries, ers, ina, ines, able, ls, io, itat, itats, itzar, iva, ives, ivisme, ius, fer, ment, amen, ament, aments, ments, ot, sfera, al, als, era, ana, iste, aire, eria, esa, eses, esos, or, icia, icies, icis, ici, ici, icis, aria, aries, alla, cio, cions, nca, nces, o, dor, all, il, istic', enc, enca, is, issa, issos, issem, issiu, issem, isseu, isseu, os, osa, dora, dores, dors, adura, ble, bles, ivol, ivola, dis, egar, ejar, ificar, itar, ables, adors, idores, idors, adora, acio, doras, dur, dures, allengues, ant, ants, ancia, ncies, atoria, atories, tori, toris, ats, ions, ota, isam, ors, ora, ores, isament, bilitat, bilitats, ivitat, ivitats, ari, aris, ionisme, ionista, ionistes, ialista, ialistes, ialisme, ialismes, ud, uts, uds, encia, encies, encia, encies, itat, itats, atiu, atius, atives, ativa, ativitat, ativitats, ible, ibles, assa, asses, assos, ent, ents, issim, issima, issims, issimes, issem, isseu, issin, ims, ima, imes, isme, ista, ismes, istes, inia, inies, iinia, inies, ita, ites, triu, trius, oses, osos, ient, otes, ots$);
- $T_3 = (acions, ada, ades)$;
- $T_4 = (logia, logies, logia, logies, logi, logis, logica, logics, logiques)$;
- $T_5 = (ic, ica, ics, iques)$;
- $T_6 = (quissim, quissims, quissimes, quissima)$;
- $T_7 = (ador, adoraadors, adores, re, ie, ent, ents, udes, ara, eren, ara, arian, arias, aran, aras, ariais, aria, arian, arien, aries, aras, aria, areis, ariamos, aremos, ara, are, ares, erian, erias, eran, eras, eriais, eria, ereis, eriamos, eremos, era, ere, er, erau, erass, irian, irias, iran, iras, iriais, iria, ireis, iriamos, iremos, ira, ire, irem, ireu, ieu, ia, ies, iem, ieu, ien, at, ut, uda, ava, aves, avem, avem, avem,$

$aveu, aveu, aven, au, ats, asseu, esseu, eresseu, asseu, assem, assim,$ $assiu, essen, esses, assen, asses, assim, assiu, essen, esseuessim,$ $essiu, essem, i, ares, arem, areu, aren, ariem, arieu, areu, aren, ant,$ $im, iu, es, ien, en, es, em, am, ams, ia, ies, dre, eix, eixer, tzar, eixes,$ $ides, ides, it, it, ida, aba, ada, ades, ida, ia, iera, ad, ed, its, id, idsase,$ $iese, aste, iste, an, aban, ian, aran, ieran, asen, iesen, aron, ieron,$ $ado, ido, iendo, io, ar, ir, as, ieu, ii, io, ia, ess, essin, essisass, assin,$ $assis, essim, essim, essiu, abas, adas, idas, ias, aras, ieras, ases,$ $ieses, is, ais, abais, iais, arais, ieraisaseis, ieseis, asteis, isteis,$ $ados, idos, amos, abamos, iamos, imos, ques, aramos, ieramos,$ $iesemos, asemos, ira, iran, irem, iren, ires, ireu, iria, irien, iries, ira,$ $iras, ire, iriem, irieu, isquen, iguem, igueu, esqui, esquin, esquis,$ $eixi, eixin, eixis, eixen, eixo, isin, isisesques, sis, sin, int, iriem, irieu,$ $isc, atges, esca, esquen, issen, isses, issin, issis, isca, issiu, issim, isc,$ $isca, issin, issiu, issim, issis, iguem, igueu, ira, iren, ires, isquen,$ $isques, issen, isses, ixo, ixen, ixes, ix, ixo, ixen, ixes, ix, ixa, inin,$ $inis, ini, ineu, itza, itzi, itzeu, itzis, itzo, itz, itza, arem, in, as, ii, iin,$ iis);

- $T_8 = (ando)$;
- $T_9 = (os, a, o, a, a, i, o, e, e, eu, iu, is, i, ir, s, i, itz, i, in, is, it)$;
- $T_{10} = (iqu)$;

Далее приводится описание алгоритма аналитического выделения основы.

Шаг 1:

Найти самый длинный аффикс из списка T_1, и если он находится в области R_1, то удалить его.

Шаг 2:

Найти самый длинный аффикс из списка $T_2' = T_2 \cup T_3 \cup T_4 \cup T_5 \cup T_6$.

Если найденный аффикс принадлежит списку T_2 и находится в области R_1, то удалить его.

Если найденный аффикс принадлежит списку T_3 и находится в области R_2, то удалить его.

Если найденный аффикс принадлежит списку T_4 и находится в области R_2, то заменить его на log.

Если найденный аффикс принадлежит списку T_5 и находится в области R_2, то заменить его на ic.

Если найденный аффикс принадлежит списку T_6 и находится в области R_1, то заменить его на c

Если на шаге 2 не было совершено ни одного преобразования псевдоосновы, то перейти к шагу 3, иначе перейти сразу к шагу 4.

Шаг 3:

Найти самый длинный аффикс из списка $T_7' = T_7 \cup T_8$.

Если найденный аффикс принадлежит списку T_7 и находится в области R_1, то удалить его.

Если найденный аффикс принадлежит списку T_8 и находится в области R_2, то удалить его.

Шаг 4:

Найти самый длинный аффикс из списка $T_9' = T_9 \cup T_{10}$.

Если найденный аффикс принадлежит списку T_9 и находится в области R_1, то удалить его.

Если найденный аффикс принадлежит списку T_{10} и находится в области R_1, то заменить его на ic.

В заключение нужно преобразовать все формы с акцентом к соответствующим им формам без акцета.

Полученный результат является искомой псевдоосновой анализируемой словоформы.

4.7. Латышский язык

Латышский язык является преимущественно флективным языком с элементами аналитизма.

В латышском языке следующие буквы являются гласными: $a, e, i, o, u, \bar{a}, \bar{\imath}, \bar{e}, \bar{u}$.

Определим R_1 как часть слова за первой негласной (согласной или заглавной гласной), следующей за гласной, или конец слова, если такой негласной нет.

Для латышского языка в терминах обобщенного алгоритма определяются следующие функции $f \in F_C$:

- Буква (или последовательность букв) располагается в области R_1;

В соответствии со схемой обобщенного алгоритма анализа будут сформированы списки аффиксов $T_1, \dots, T_6$:

- $T_1 = (ajiem, ajai)$;
- $T_2 = (ajam, aj\bar{a}m, ajos, aj\bar{a}s, aj\bar{a}, ais, ai, ei)$;
- $T_3 = (iem)$;
- $T_4 = (\bar{a}m, am, \bar{e}m, \bar{\imath}m, im, um, as, \bar{a}s, es, ij, \bar{\imath}s, \bar{e}s, is, ie, e, \bar{a}, \bar{e}, \bar{\imath}, \bar{u}, o)$;
- $T_5 = (os, us, u, a, i)$;
- $T_6 = (s, \check{s})$;

Далее приводится описание алгоритма аналитического выделения основы.

Шаг 1:

Найти самый длинный аффикс из списка T_1, и если он находится в области R_1, а также анализируемая словоформа содержит более трех гласных букв, то удалить найденный аффикс.

Шаг 2:

Найти самый длинный аффикс из списка T_2, и если он находится в области R_1, а также анализируемая словоформа содержит более двух гласных букв, то удалить найденный аффикс.

Шаг 3:

Найти самый длинный аффикс из списка T_3, и если он находится в области R_1, а также анализируемая словоформа содержит более двух гласных букв, то удалить найденный аффикс.

В случае, если аффикс был найден и удален, применить следующее преобразование полученной после удаления аффикса псевдоосновы:

- Если удаленный аффикс был u, то заменить kš(ņņ) на конце псевдоосновы на $kst(nn)$ соответственно;
- Если псевдооснова оканчивается на pj, bj, mj, vj, то удалить последнюю букву;
- Если псевдооснова оканчивается на šņ, žņ, šļ, žļ, ļņ, ļļ, č, ļ, ņ, то заменить соответственно на $sn, zn, sl, zl, ln, ll, c, l, n$.

Шаг 4:

Найти самый длинный аффикс из списка T_4, и если он находится в области R_1, а также анализируемая словоформа содержит более одной гласной буквы, то удалить найденный аффикс.

Шаг 5:

Найти самый длинный аффикс из списка T_3, и если он находится в области R_1, а также анализируемая словоформа содержит более одной гласной

буквы, то удалить найденный аффикс. Также выполнить преобразования, описанные в шаге 3.

Шаг 6:

Найти самый длинный аффикс из списка T_4, и если он находится в области R_1, а также анализируемая словоформа содержит хотя бы одну гласную букву, то удалить найденный аффикс.

Полученный результат является искомой псевдоосновой анализируемой словоформы.

4.8. Литовский язык

Литовский язык представляет собой флективный язык синтетического типа с элементами агглютинации и аналитизма.

В литовском языке следующие буквы являются гласными: $a, e, i, y, o, u, ą, ę, į, ų, ė, ū$.

Определим R_1 как часть слова за первой негласной (согласной или заглавной гласной), следующей за гласной, или конец слова, если такой негласной нет.

Для литовского языка в терминах обобщенного алгоритма определяются следующие функции $f \in F_C$:

- Буква (или последовательность букв) располагается в области R_1;

В соответствии со схемой обобщенного алгоритма анализа будут сформированы списки аффиксов $T_1, \dots, T_5$:

- $T_1 = (tis, tas)$;
- $T_2 = (uosiuose, aisiais, iesiems, osioms, ėjausi, uosius, usios, usius, amasi, osios, osiom, iomis, iajam, ajame, ianti, ajam, asis, amės, uoju, uoji, uoja, usis, omis, ieji, ioji, iaus, iais, iant, anti, inti, iuose,$

$iasi, iams, ioje, iose, uose, iems, ėmis, umas, tųjų, iame, toje, ios, ąjį, ąją, iąs, oms, iau, asi, ose, oje, ojo, oji, yse, yje, iai, ias, ies, ius, ėse, ėje, ers, ens, ais, ams, aus, iui, umu, ėms, iam, imo, yti, uos, ant, osi, om, oj, os, uo, us, ys, ia, ią, ės, es, ei, as, ai, ui, iu, io, ūs, ti, ąs, ęs, is, ių, ų, u, o, į, i, ę, ė, e, ą, a, s$);

- $T_3 = (č, dž)$;
- $T_4 = (uomen, inink, okšn, iaus, enyb, ojim, ėjim, imas, avus, ulyt, iuoj, iant, iul, ant, uos, uoj, iuk, dam, dav, auj, ain, išk, eiv, int, uol, tok, toj, tas, iam, iau, iav, atv, yst, ien, ykl, tuv, tuk, ikl, esn, ekl, sen, tyn, izm, jim, ėt, ia, ėz, šė, už, yt, el, av, uk, oj, ėn, om, oč, ėj, ij, iuo, uo, um, yk, ės, in, ūn, ik, ėl, yb, sm, es, ym)$;
- $T_5 = (iant, iūkšt)$;

Далее приводится описание алгоритма аналитического выделения основы.

Шаг 1:

Найти самый длинный аффикс из списка T_1, и если он находится в области R_1, то удалить его.

Если на данном шаге произошла модификации псевдоосновы, то перейти к шагу 4, иначе перейти к шагу 2.

Шаг 2:

Найти самый длинный аффикс из списка T_2, и если он находится в области R_1, то удалить его.

Шаг 3:

Найти самый длинный аффикс из списка T_3, и если он находится в области R_1, то заменить его на t, если найденный аффикс был $č$, либо на d в противном случае.

Шаг 4:

Найти самый длинный аффикс из списка T_4, и если он находится в о в области R_1, то удалить его.

Повторить данный шаг не более трех раз.

Шаг 5:

Если на шаге 3 псевдооснова была модифицирована, то найти самый длинный аффикс из списка T_5, и если он находится в о в области R_1, то удалить его.

Шаг 6:

Если на шаге 2 псевдооснова была модифицирована, то повторить действия, описанные на шаге 3.

Полученный результат является искомой псевдоосновой анализируемой словоформы.

4.9. Немецкий язык

Немецкий язык является языком флективного типа. В современном немецком языке внешняя и внутренняя флексии сочетаются с аналитизмом. Синтетический способ представлен в основах множественного числа существительных, степенях сравнения прилагательных и наречий. При этом синтаксическая зависимость существительного в основном маркируется аналитически — формой артикля.

В немецком языке существуют три буквы с акцентом: ä, ö, ü, а также специальная буква ß, которая является эквивалентом удвоенной s.

Следующие буквы являются гласными: a, e, i, o, u, y, ä, ö, ü.

Алгоритм начинает работу с того, что заменяет ß на ss и устанавливает буквы u и y между гласными в верхний регистр. Этот этап соответствует применению функций трансформации $f \in F_T$ в схеме обобщенного алгоритма анализа словоформы.

Определим R_1 как область, начинающуюся после первой согласной буквы следующей за гласной. Если такой согласной нет, то область пустая. R_2 — область, начинающаяся после первой согласной буквы, следующей за гласной в области R_1. Аналогично, если нет такой согласной, то область пустая.

R_1 и R_2 первоначально настраиваются по умолчанию, но затем R_1 увеличивается так, чтобы область до него содержала минимум 3 буквы.

Определим S -окончание как одну букву из списка: $b, d, f, g, h, k, l, m, n, r, t$.

Определим ST -окончание, как букву из того же списка, за исключением буквы r.

Для немецкого языка в терминах обобщенного алгоритма определяются следующие функции $f \in F_C$:

- Буква является S -окончанием;
- Буква является ST -окончанием;
- Буква (или последовательность букв) располагается в области R_1;
- Буква (или последовательность букв) располагается в области R_2.

В соответствии со схемой обобщенного алгоритма анализа будут сформированы списки аффиксов $T_1, \dots, T_6$:

- $T_1 = (em, ern, er, e, en, es, s)$;
- $T_2 = (en, er, est, st)$;
- $T_3 = (end, ung)$;
- $T_4 = (ig, ik, isch)$;
- $T_5 = (lich, heit)$;
- $T_6 = (keit)$;

Далее приводится описание алгоритма аналитического выделения основы.

Шаг 1:

Найти самый длинный аффикс из списка T_1, и если он находится в области R_1, то удалить его. Если найденный аффикс — это аффикс s, то ему должно предшествовать -окончание.

Шаг 2:

Найти самый длинный аффикс из списка T_2, и если он находится в области R_1, то удалить его. Если найденный аффикс — это аффикс st, то ему должно предшествовать -окончание, перед которым, в свою очередь, должно быть минимум три буквы.

Шаг 3:

Найти самый длинный аффикс из списка $T_3' = T_3 \cup T_4 \cup T_5 \cup T_6$.

Если найденный аффикс из списка T_3 и он находится в области R_2, то удалить его. При этом, если найденному аффиксу предшествует ig, притом ig находится в области R_2 и ему не предшествует e, то ig также необходимо удалить.

Если найденный аффикс из списка T_4, то удалить его, если он находится в области R_2 и ему не предшествует e.

Если найденный аффикс из списка T_5, то удалить его, если он находится в области R_2. Если найденному аффиксу предшествует либо аффикс er, либо аффикс en, находящийся в области R_1, то его также следует удалить.

Если найденный аффикс из списка T_6, то удалить его, если он находится в области R_2. Если найденному аффиксу предшествует либо аффикс $lich$, либо аффикс ig, находящийся в области R_2, то его также следует удалить.

В заключение нужно вернуть буквы U и Y в нижний регистр и удалить умляут над буквами a, o и u. Этот шаг соответствует применению функций

трансформации $f \in F_T$, обратных к тем, что были применены на предварительном этапе.

Полученный результат является искомой псевдоосновой анализируемой словоформы.

4.10. Польский язык

Польский язык принадлежит к флективным языкам синтетического типа, что реализуется, в частности, в системе падежно-числового формообразования существительных. Отдельные формы словоизменения носят аналитический характер (будущее время изъявительного наклонения, страдательный залог). Также присутствуют элементы агглютинации.

В польском языке следующие буквы являются гласными: $a, e, i, o, u, y,$ ą, ę, ó.

Определим R_1 как часть слова за первой негласной (согласной или заглавной гласной), следующей за гласной, или конец слова, если такой негласной нет.

Для польского языка в терминах обобщенного алгоритма определяются следующие функции $f \in F_C$:

- Буква (или последовательность букв) располагается в области R_1.

В соответствии со схемой обобщенного алгоритма анализа будут сформированы списки аффиксов T_1, T_2:

- $T_1 = (owie, y, i, e, a, u,$ ów$, owi, om, em, ami, ach, e!, u!)$;
- $T_2 = ($ę$, emy, ecie, e, esz,$ ą$, y, imy, isz, icie, i, am, amy, asz, acie, a,$ ają$, em,$ ają$, m,$ śmy$,$ ś$,$ ście$)$;

Далее приводится описание алгоритма аналитического выделения основы.

Шаг 1:

Найти самый длинный аффикс из списка $T_1' = T_1 \cup T_2$, и если она находится в области R_1, то удалить его.

Полученный результат является искомой псевдоосновой анализируемой словоформы.

4.11. Португальский язык

Португальский язык относится к смешанному, синтетическо-аналитическому типу. Связи между словами осуществляются с помощью флексий, служебных слов, порядка слов.

В португальском языке следующие буквы с акцентом: á, é, í, ó, ú, â, ê, ô, ç, ã, õ, ü, ñ.

Следующие буквы являются гласными: a, e, i, o, u, á, é, í, ó, ú, â, ê, ô.

Существуют две назальные (носовые) формы гласных: ã, õ, — которые следует рассматривать как гласные, за которыми следуют согласные. Поэтому ã и õ в слове заменяются на $a\sim$ и $o\sim$, где $\sim$ — разделительный символ, который следует рассматривать как согласную букву. Этот этап соответствует применению функций трансформации $f \in F_T$ в схеме обобщенного алгоритма анализа словоформы.

Определим R_1 как часть слова за первой негласной (согласной или заглавной гласной), следующей за гласной, или конец слова, если такой

негласной нет. R_2 — часть слова за первой негласной, следующей за гласной, находящейся в R_1, или конец слова, если такой негласной нет.

Область RV определяется следующим образом: если в слове вторая буква согласная, то RV — это часть слова после следующей гласной; если первые две буквы гласные, то RV — часть слова после следующей согласной буквы, иначе, если первая — согласная, а вторая — гласная, то RV — это область слова после третьей буквы. Но в данных условиях RV не может начинаться в конце слова.

Для португальского языка в терминах обобщенного алгоритма определяются следующие функции $f \in F_C$:

- Буква (или последовательность букв) располагается в области R_1;
- Буква (или последовательность букв) располагается в области R_2;
- Буква (или последовательность букв) располагается в области RV.

В соответствии со схемой обобщенного алгоритма анализа будут сформированы списки аффиксов $T_1, \dots, T_{11}$:

- $T_1 = (eza, ezas, ico, ica, icos, icas, ismo, ismos, \acute{a}vel, \acute{i}vel, ista, istas, oso, osa, osos, osas, amento, amentos, imento, imentos, adora, ador, aça{\sim}o, adoras, adores, aço{\sim}es, ante, antes, \hat{a}ncia)$;
- $T_2 = (log\acute{i}a, log\acute{i}as)$;
- $T_3 = (uci\acute{o}n, uciones)$;
- $T_4 = (\hat{e}ncia, \hat{e}ncias)$;
- $T_5 = (amente)$;
- $T_6 = (mente)$;
- $T_7 = (idade, idades)$;
- $T_8 = (iva, ivo, ivas, ivos)$;
- $T_9 = (ira, iras)$;
- $T_{10} = (ada, ida, ia, aria, eria, iria, ar\acute{a}, ara, er\acute{a}, era, ir\acute{a}, ava, asse, esse, isse, aste, este, iste, ei, arei, erei, irei, am, iam,$

$ariam$, $eriam$, $iriam$, $aram$, $eram$, $iram$, $avam$, em, $arem$, $erem$, $irem$, $assem$, $essem$, $issem$, ado, ido, $ando$, $endo$, $indo$, $ara\sim o$, $era\sim o$, $ira\sim o$, ar, er, ir, as, $adas$, $idas$, ias, $arias$, $erias$, $irias$, $ará s$, $aras$, $erá s$, $eras$, $irá s$, $avas$, es, $ardes$, $erdes$, $irdes$, $ares$, $eres$, $ires$, $asses$, $esses$, $isses$, $astes$, $estes$, $istes$, is, ais, eis, $íeis$, $aríeis$, $eríeis$, $iríeis$, $áreis$, $areis$, $éreis$, $ereis$, $íreis$, $ireis$, $ásseis$, $ésseis$, $ísseis$, $áveis$, $ados$, $idos$, $ámos$, $amos$, $íamos$, $aríamos$, $eríamos$, $iríamos$, $áramos$, $éramos$, $íramos$, $ávamos$, $emos$, $aremos$, $eremos$, $iremos$, $ássemos$, $êssemos$, $íssemos$, $imos$, $armos$, $ermos$, $irmos$, eu, iu, ou, ira, $iras$);

- $T_{11} = (os, a, i, o, á, í, ó)$;

Далее приводится описание алгоритма аналитического выделения основы.

Шаг 1:

Найти самый длинный аффикс из списка $T_1' = T_1 \cup T_2 \cup T_3 \cup T_4 \cup T_5 \cup T_6 \cup T_7 \cup T_8 \cup T_9$.

Если найденный аффикс принадлежит списку T_1 и находится в области R_2, то удалить его.

Если найденный аффикс принадлежит списку T_2 и находится в области R_2, то заменить его на log.

Если найденный аффикс принадлежит списку T_3 и находится в области R_2, то заменить его на u.

Если найденный аффикс принадлежит списку T_4 и находится в области R_2, то заменить его на $ente$.

Если найденный аффикс принадлежит списку T_5 и находится в области R_2, то удалить его. При этом, если найденному аффиксу предшествует iv, то удалить его, если он находится в области R_2 (если аффиксу предшествует at, то удалить at, если тот находится в области R_2); если найденный аффикс следует за os, ic или ad, то удалить os, ic или ad, если тот находится в области R_2.

Если найденный аффикс принадлежит списку T_6 и находится в области R_2, то удалить его. При этом, если найденный аффикс следует за $ante, able$ или $ible$, то удалить $ante, avel$ или $ível$, если тот находится в области R_2.

Если найденный аффикс принадлежит списку T_7 и находится в области R_2, то удалить его. При этом, если найденный аффикс следует за $abil, ic$ или iv, то удалить $abil, ic$ или iv, если тот находится в области R_2.

Если найденный аффикс принадлежит списку T_8 и находится в области R_2, то удалить его. При этом, если найденный аффикс следует за at, то удалить at, если тот находится в области R_2.

Если найденный аффикс принадлежит списку T_9 и находится в области RV, а также ему предшествует аффикс e, то заменить найденный аффикс на if.

Если на Шаге 1 не был удален ни один аффикс, то перейти к Шагу 2.

Шаг 2:

Найти самый длинный аффикс из списка T_{10} и удалить его, если он находится в области RV.

Если псевдооснова была преобразована либо на Шаге 1, либо на Шаге 2, то перейти к Шагу 3, иначе перейти к Шагу 4.

Шаг 3:

Удалить суффикс i, если он находится в области RV и ему предшествует c. Перейти к Шагу 5.

Шаг 4:

Найти самый длинный аффикс из списка T_{11} и удалить его, если он находится в области RV.

Шаг 5:

Если слово оканчивается на одну из букв $e, \acute{e}, \hat{e}$ и эта буква находится в области RV, то удалить ее; если ей предшествует аффикс gu (или ci), причем u (или i) из последнего находятся в области RV, то удалить u (или i).

Если слово заканчивается на ç, то удалить седиль.

В заключение нужно преобразовать $a\sim$ и $o\sim$ обратно в ã и õ соответственно. Этот шаг соответствует применению функций трансформации $f \in F_T$, обратных к тем, что были применены на предварительном этапе.

Полученный результат является искомой псевдоосновой анализируемой словоформы.

4.12. Румынский язык

Румынский язык относится к флективно-аналитическому типу. В области формообразования наблюдается тесное переплетение тенденций к синтетизму и аналитизму, что находит выражение в сложном построении именных и глагольных синтагм. Особенностью словоизменения является наличие значительного количества чередований гласных и согласных в основе.

В румынском языке следующие буквы с акцентом: ă, â, î, ş, ţ.

Следующие буквы являются гласными: a, e, i, o, u, ă, â, î.

В начале работы необходимо перевести в верхний регистр буквы u, i, стоящие между гласными. Этот этап соответствует применению функций трансформации $f \in F_T$ в схеме обобщенного алгоритма анализа словоформы.

Символы, переведенные в верхний регистр, не являются гласными в дальнейшем описании алгоритма.

Определим R_1 как часть слова за первой негласной (согласной или заглавной гласной), следующей за гласной, или конец слова, если такой негласной нет. R_2 — часть слова за первой негласной, следующей за гласной, находящейся в R_1, или конец слова, если такой негласной нет.

Область RV определяется следующим образом: если в слове вторая буква согласная, то RV — это часть слова после следующей гласной; если первые две буквы гласные, то RV — часть слова после следующей согласной буквы, иначе, если первая — согласная, а вторая — гласная, то RV — это область слова после третьей буквы. Но в данных условиях RV не может начинаться в конце слова.

Для румынского языка в терминах обобщенного алгоритма определяются следующие функции $f \in F_C$:

- Буква (или последовательность букв) располагается в области R_1;
- Буква (или последовательность букв) располагается в области R_2;
- Буква (или последовательность букв) располагается в области RV.

В соответствии со схемой обобщенного алгоритма анализа будут сформированы списки аффиксов $T_1, \dots, T_{19}$:

- $T_1 = (ul, ului)$;
- $T_2 = (aua)$;
- $T_3 = (ea, ele, elor)$;
- $T_4 = (ii, iua, iei, iile, iilor, ilor)$;
- $T_5 = (ile)$;
- $T_6 = (atei)$;

- $T_7 = (aţie, aţia)$;
- $T_8 = (abilitate, abilitati, abilităi, abilităţi)$;
- $T_9 = (ibilitate)$;
- $T_{10} = (ivitate, ivitati, ivităi, ivităţi)$;
- $T_{11} = (icitate, icitati, icităi, icităţi, icator, icatori, iciv, iciva, icive, icivi, icivă, ical, icala, icale, icali, icală)$;
- $T_{12} = (ativ, ativa, ative, ativi, ativă, aţiune, atoare, ator, atori, ătoare, ător, ători)$;
- $T_{13} = (itiv, itiva, itive, itivi, itivă, iţiune, itoare, itor, itori)$;
- $T_{14} = (at, ata, ată, ati, ate, ut, uta, ută, uti, ute, it, ita, ită, iti, ite, ic, ica, ice, ici,ică, abil, abila, abile, abili, abilă, ibil, ibila, ibile, ibili, ibilă, oasa, oasă, oase, os, osi, oşi, ant, anta, ante, anti, antă, ator, atori, itate, itati, ităi, ităţi, iv, iva, ive, ivi, ivă)$;
- $T_{15} = (iune, iuni)$;
- $T_{16} = (ism, isme, ist, ista, iste, isti, istă, işti)$;
- $T_{17} = (are, ere, ire, âre, ind, ând, indu, ându, eze, ească, ez, ezi, ează,$
- $esc, eşti, eşte, ăsc, ăşti, ăşte, am, ai, au, eam, eai, ea, eaţi, eau, iam, iai, ia, iaţi, iau, ui, aşi, arăm, arăţi, ară, uşi, urăm, urăţi, ură, işi, irăm, irăţi, ură, âi, âşi, ârăm, ârăţi, âră, asem, aseşi, ase, aserăm, aserăţi, aseră, isem, iseşi, ise, iserăm, iserăţi, iseră, âsem, âseşi, âse, âserăm, âserăţi, âseră, usem, useşi, use, userăm, userăţi, useră)$;
- $T_{18} = (ăm, aţi, em, eţi, im, iţi, âm, âţi, seşi, serăm, serăţi, seră, sei, se, sesem, seseşi, sese, seserăm, seserăţi, seseră)$;
- $T_{19} = (a, e, i, ie, ă)$;

Далее приводится описание алгоритма аналитического выделения основы.

Шаг 0:

Найти самый длинный аффикс из списка $T_1' = T_1 \cup T_2 \cup T_3 \cup T_4 \cup T_5 \cup T_6 \cup T_7$.

Если найденный аффикс принадлежит списку T_1 и находится в области R_1, то удалить его.

Если найденный аффикс принадлежит списку T_2 и находится в области R_1, то заменить его на a.

Если найденный аффикс принадлежит списку T_3 и находится в области R_1, то заменить его на e.

Если найденный аффикс принадлежит списку T_4 и находится в области R_1, то заменить его на i.

Если найденный аффикс принадлежит списку T_5 и находится в области R_1, то заменить его на i, если ему не предшествует ab.

Если найденный аффикс принадлежит списку T_6 и находится в области R_1, то заменить его на at.

Если найденный аффикс принадлежит списку T_7 и находится в области R_1, то заменить его на $aţi$.

Шаг 1:

Найти самый длинный аффикс из списка $T_8' = T_8 \cup T_9 \cup T_{10} \cup T_{11} \cup T_{12} \cup T_{13}$.

Если найденный аффикс принадлежит списку T_8 и находится в области R_1, то заменить его на $abil$.

Если найденный аффикс принадлежит списку T_9 и находится в области R_1, то заменить его на $ibil$.

Если найденный аффикс принадлежит списку T_{10} и находится в области R_1, то заменить его на iv.

Если найденный аффикс принадлежит списку T_{11} и находится в области R_1, то заменить его на ic.

Если найденный аффикс принадлежит списку T_{12} и находится в области R_1, то заменить его на at.

Если найденный аффикс принадлежит списку T_{13} и находится в области R_1, то заменить его на it.

Повторять данный шаг до тех пор, пока происходит модификация псевдоосновы.

Шаг 2:

Найти самый длинный суффикс из представленных и выполнить требуемое действие, если он находится в R_2:

Найти самый длинный аффикс из списка $T'_{14} = T_{14} \cup T_{15} \cup T_{16}$.

Если найденный аффикс принадлежит списку T_{14} и находится в области R_2, то удалить его.

Если найденный аффикс принадлежит списку T_{15} и находится в области R_2, то удалить его, если ему предшествует ţ. При этом заменить ţ на t.

Если найденный аффикс принадлежит списку T_{16} и находится в области R_2, то заменить его на ist.

Шаг 3:

Данный шаг выполняется в том случае, если после шагов 1 и 2 псевдооснова не была изменена.

Найти самый длинный аффикс из списка $T'_{17} = T_{17} \cup T_{18}$.

Если найденный аффикс принадлежит списку T_{17} и находится в области RV, то удалить его, если в области RV найденному аффиксу предшествует согласная или u.

Если найденный аффикс принадлежит списку T_{18} и находится в области RV, то удалить его.

Шаг 4:

Найти самый длинный аффикс из списка T_{19}, и если он находится в области RV, то удалить его.

В заключение нужно вернуть I, U в нижний регистр — этот шаг соответствует применению функций трансформации $f \in F_T$, обратных к тем, что были применены на предварительном этапе.

Полученный результат является искомой псевдоосновой анализируемой словоформы.

4.13. Сербскохорватский язык

Сербскохорватский язык относится к языкам флективного морфологического строя. Черты аналитизма незначительны и наблюдаются в диалектах, пограничных с болгарским и македонским языками.

В сербскохорватском языке следующие буквы являются гласными: а, е, и, о, у.

Определим R_1 как часть слова за первой негласной (согласной или заглавной гласной), следующей за гласной, или конец слова, если такой негласной нет.

Для сербскохорватского языка в терминах обобщенного алгоритма определяются следующие функции $f \in F_C$:

- Буква (или последовательность букв) располагается в области R_1.

В соответствии со схемой обобщенного алгоритма анализа будут сформированы списки аффиксов T_1, T_2:

- $T_1 = (\text{о}, \text{е}, \text{и}, \text{а}, \text{у}, \text{има}, \text{ом}, \text{ем}, \text{ама}, \text{на}, \text{та}, \text{ну}, \text{ту}, \text{ном}, \text{том}, \text{нима}, \text{тима})$;
- $T_2 = (\text{и}, \text{а}, \text{о}, \text{е}, \text{ог}, \text{их}, \text{ом}, \text{oj}, \text{им}, \text{у})$;

Далее приводится описание алгоритма аналитического выделения основы.

Шаг 1:

Найти самый длинный аффикс из списка T_1, и если он находится в области R_1, то удалить его.

Шаг 2:

Если на шаге 1 не была произведена модификация псевдоосновы, то найти самый длинный аффикс из списка T_2 и если он находится в области R_1, то удалить его.

Полученный результат является искомой псевдоосновой анализируемой словоформы.

4.14. Словацкий язык

Словацкий язык является по преимуществу языком флективного морфологического типа. Отмечаются элементы агглютинативного характера, что проявляется в тенденции к унификации окончаний в основе. В глаголе отмечаются черты аналитизма.

В словацком языке следующие буквы являются гласными: a, e, i, o, u, y, á, ä, é, í, ó, ô, ú, ý.

Определим R_1 как часть слова за первой негласной (согласной или заглавной гласной), следующей за гласной, или конец слова, если такой негласной нет.

Область RV определяется следующим образом: если в слове вторая буква согласная, то RV — это часть слова после следующей гласной; если первые две буквы гласные, то RV — часть слова после следующей согласной буквы, иначе, если первая — согласная, а вторая — гласная, то RV — это область слова после третьей буквы. Но в данных условиях RV не может начинаться в конце слова.

Для словацкого языка в терминах обобщенного алгоритма определяются следующие функции $f \in F_C$:

- Буква (или последовательность букв) располагается в области R_1;
- Буква (или последовательность букв) располагается в области RV.

В соответствии со схемой обобщенного алгоритма анализа будут сформированы списки аффиксов $T_1, \dots, T_5$:

- $T_1 = (atách, om, atám, ách, ých, ové, ými, ata, aty, ama, ami, ovi, at, ám, os, us, ým, mi, ou, u, y, a, o, á, é, ý)$;
- $T_2 = (ech, ich, ích, ého, ámi, ému, áte, áti, ího, ími, emi, iho, imu, ém, ím, es, e, i, í)$;
- $T_3 = (em)$;
- $T_4 = (ov)$;
- $T_5 = (in)$;

Далее приводится описание алгоритма аналитического выделения основы.

Шаг 1:

Найти самый длинный аффикс из списка $T_1' = T_1 \cup T_2 \cup T_3$.

Если найденный аффикс принадлежит списку T_1, то удалить его.

Если найденный аффикс принадлежит списку T_2, то удалить его. Также выполнить процедуру «смягчения аффикса». В рамках данной процедуры выполнить следующие действия:

- Если псевдооснова оканчивается на $ci, ce, \check{c}i, \check{c}$, то заменить аффикс на k.
- Если псевдооснова оканчивается на $zi, ze, \check{z}i, \check{z}e$, то заменить аффикс на h.
- Если псевдооснова оканчивается на $\check{c}t\acute{a}, \check{c}ti, \check{c}t\acute{e}$, то заменить аффикс на ck.
- Если псевдооснова оканчивается на $\check{s}t\acute{a}, \check{s}ti, \check{s}t\acute{e}$, то заменить аффикс на sk.

Если найденный аффикс принадлежит списку T_3, то заменить его на e и выполнить процедуру «смягчения аффикса», описанную выше.

Шаг 2:

Найти самый длинный аффикс из списка $T_4' = T_4 \cup T_5$.

Если найденный аффикс принадлежит списку T_4 и находится в области RV, то удалить его.

Если найденный аффикс принадлежит списку T_5, то удалить его и выполнить процедуру «смягчения аффикса», описанную выше.

Полученный результат является искомой псевдоосновой анализируемой словоформы.

4.15. Словенский язык

Словенский язык принадлежит к числу флективных языков, синтетические способы выражения грамматических значений преобладают над аналитическими. В именных категориях словоизменение носит синтетический характер (элементы аналитизма возникают у существительных за счет употребления предлогов).

В словенском языке следующие буквы являются гласными: a, e, i, o, u.

Определим R_1 как часть слова за первой негласной (согласной или заглавной гласной), следующей за гласной, или конец слова, если такой негласной нет.

Для словенского языка в терминах обобщенного алгоритма определяются следующие функции $f \in F_C$:

- Буква (или последовательность букв) располагается в области R_1;

В соответствии со схемой обобщенного алгоритма анализа будут сформированы списки аффиксов $T_1, \dots, T_6$:

- $T_1 = (ovski, evski, anski)$;
- $T_2 = (stvo, štvo)$;
- $T_3 = (šen, ski, ček, ovm, ega, ovi, ijo, ija, ema, ste, ejo, ite, ila, šče, ški, ost, ast, len, ven, vna, čan, iti)$;
- $T_4 = (al, ih, iv, eg, ja, je, em, en, ev, ov, jo, ma, mi, eh, ij, om, do, oč, ti, il, ec, ka, in, an, at, ir)$;
- $T_5 = (š, m, c, a, e, i, o, u)$;
- $T_6 = (a, e, i, o, u)$;

Далее приводится описание алгоритма аналитического выделения основы.

Последовательность шагов 1-7 выполняется до тех пор, пока псевдооснова не перестанет модифицироваться, но не более четырех раз.

Шаг 1:

Найти самый длинный аффикс из списка T_1, и если он находится в области R_1, то удалить его.

Шаг 2:

Найти самый длинный аффикс из списка T_2, и если он находится в области R_1, то удалить его.

Шаг 3:

Найти самый длинный аффикс из списка T_3, и если он находится в области R_1, то удалить его.

Шаг 4:

Найти самый длинный аффикс из списка T_4, и если он находится в области R_1, то удалить его.

Шаг 5:

Найти самый длинный аффикс из списка T_5, и если он находится в области R_1, то удалить его.

Шаг 6:

Если псевдооснова оканчивается на согласную, то удалить последнюю букву.

Шаг 7:

Найти самый длинный аффикс из списка T_6, и если он находится в области R_1, то удалить его.

Полученный результат является искомой псевдоосновой анализируемой словоформы.

4.16. Украинский язык

Украинский язык относится к языкам флективного морфологического типа. Флексии являются средством образования грамматических форм слова, а также основным средством выражения синтаксических связей.

Следующие буквы являются гласными: а, е, є, и, і, ї, о, у, ю, я.

Определим R_1 как область, начинающуюся после первой согласной буквы следующей за гласной. Если такой согласной нет, то область пустая. R_2 — область, начинающаяся после первой согласной буквы, следующей за гласной в области R_1. Аналогично, если нет такой согласной, то область пустая.

R_1 и R_2 первоначально настраиваются по умолчанию, но затем R_1 увеличивается так, чтобы область до него содержала минимум 3 буквы.

Определим RV как область, начинающуюся после первой гласной буквы.

Для украинского языка в терминах обобщенного алгоритма определяются следующие функции $f \in F_C$:

- Буква (или последовательность букв) располагается в области R_2;
- Буква (или последовательность букв) располагается в области RV.

В соответствии со схемой обобщенного алгоритма анализа будут сформированы списки аффиксов $T_1, \dots, T_6$:

- $T_1 =$ (ив, ивши, ившись, ыв, ывши, ывшись, ав, авши, авшись, яв, явши, явшись);
- $T_2 =$ (ся, си, сь);
- $T_3 =$ (ими, ій, ий, а, е, ова, ове, ів, є, їй, єє, еє, я, ім, ем, им, ім, их, іх, ою, йми, іми, у, ю, ого, ому, ої);
- $T_4 =$ (сь, ся, ив, ать, ять, у, ю, ав, али, учи, ячи, вши, ши, е, ме, ати, яти, є);
- $T_5 =$ (а, ев, ов, е, ями, ами, еи, и, ей, ой, ий, й, имя, ям, ием, ем, ам, ом, о, у, ах, иях, ях, ы, ь, ию, ью, ю, ия, ья, я, і, ові, ї, ею, єю, ою, є, еві, ем, єм, ів, їв);
- $T_6 =$ (ост, ость);

Далее приводится описание алгоритма аналитического выделения основы.

Шаг 1:

Найти самый длинный аффикс из списка T_1, и если он находится в области RV, то удалить его и перейти к шагу 2.

Найти самый длинный аффикс из списка T_2, и если он находится в области RV, то удалить его и перейти к шагу 2.

Найти самый длинный аффикс из списка T_3, и если он находится в области RV, то удалить его и перейти к шагу 2.

Найти самый длинный аффикс из списка T_4 ,и если он находится в области RV, то удалить его и перейти к шагу 2.

Найти самый длинный аффикс из списка T_5, и если он находится в области RV, то удалить его и перейти к шагу 2.

Шаг 2:

Если псевдооснова оканчивается на и, то удалить последнюю букву.

Шаг 3:

Найти самый длинный аффикс из списка T_6, и если он находится в области R_2, то удалить его.

Шаг 4:

Если псевдооснова оканчивается на нн, то удалить последнюю букву.

Если псевдооснова оканчивается на ейш, ейше, то удалить соответствующий аффикс и удалить последнюю букву, если псевдооснова оканчивается на нн.

Если псевдооснова оканчивается на ь, то удалить последнюю букву.

Полученный результат является искомой псевдоосновой анализируемой словоформы.

4.17. Французский язык

Французский язык относится к смешанному флективно-аналитическому морфологическому типу. Флексии выражаются в формировании рода и числа имени, некоторых глагольных форм. Показателем аналитизма являются ограниченные словообразовательные и словоизменительные возможности французского языка. Отсутствие флексий компенсируется фиксированным порядком слов.

Во французском алфавите определены следующие буквы с акцентом: â, à, ç, ë, é, ê, è, ï, î, ô, û, ù.

Следующие буквы являются гласными: a, e, i, o, u, y, â, à, ë, é, ê, è, ï, î, ô, û, ù.

В начале работы необходимо перевести в верхний регистр буквы u, i, y, стоящие между гласными. Также в верхний регистр нужно перевести букву u, стоящую после q. Этот этап соответствует применению функций трансформации $f \in F_T$ в схеме обобщенного алгоритма анализа словоформы.

Символы, переведенные в верхний регистр, не являются гласными в дальнейшем описании алгоритма.

Если слово начинается с двух гласных, то RV — это часть слова, начинающаяся с четвертой буквы, иначе часть после первой гласной, не

стоящей первой в слове, в противном случае — конец слова. Исключениями являются слова, начинающиеся на par, col или tap. В этом случае RV — это часть слова, начинающаяся с четвертой буквы.

Определим R_1 как часть слова за первой негласной (согласной или заглавной гласной), следующей за гласной, или конец слова, если такой негласной нет. R_2 — часть слова за первой негласной, следующей за гласной, находящейся в R_1, или конец слова, если такой негласной нет. R_1 может содержать в себе RV, и RV может содержать в себе R_1.

Для французского языка в терминах обобщенного алгоритма определяются следующие функции $f \in F_C$:

- Буква (или последовательность букв) располагается в области R_1;
- Буква (или последовательность букв) располагается в области R_2;
- Буква (или последовательность букв) располагается в области RV.

В соответствии со схемой обобщенного алгоритма анализа будут сформированы списки аффиксов $T_1, \dots, T_{23}$:

- $T_1 = (ance, iqUe, isme, able, iste, eux, ances, iqUes, ismes, ables, istes)$;
- $T_2 = (atrice, ateur, ation, atrices, ateurs, ations)$;
- $T_3 = (logie, logies)$;
- $T_4 = (usion, ution, usions, utions)$;
- $T_5 = (ence, ences)$;
- $T_6 = (ement, ements)$;
- $T_7 = (ité, ités)$;
- $T_8 = (if, ive, ifs, ives)$;
- $T_9 = (eaux)$;
- $T_{10} = (aux)$;
- $T_{11} = (euse, euses)$;

- $T_{12} = (issement, issements)$;
- $T_{13} = (amment)$;
- $T_{14} = (emment)$;
- $T_{15} = (ment, ments)$;
- $T_{16} = (\text{î}mes, \text{î}t, \text{î}tes, i, ie, ies, ir, ira, irai, iraIent, irais, irait, iras, irent, irez, iriez, irions, irons, iront, is, issaIent, issais, issait, issant, issante, issantes, issants, isse, issent, isses, issez, issiez, issions, issons, it)$;
- $T_{17} = (ions)$;
- $T_{18} = (\text{é}, \text{é}e, \text{é}es, \text{é}s, \text{è}rent, er, era, erai, eraIent, erais, erait, eras, erez, eriez, erions, erons, eront, ez, iez)$;
- $T_{19} = (\text{â}mes, \text{â}t, \text{â}tes, a, ai, aIent, ais, ait, ant, ante, antes, ants, as, asse, assent, asses, assiez, assions)$;
- $T_{20} = (ion)$;
- $T_{21} = (ier, i\text{è}re, Ier, I\text{è}re)$;
- $T_{22} = (e)$;
- $T_{23} = (\text{ë})$;

Далее приводится описание алгоритма аналитического выделения основы.

Шаг 1:

Найти самый длинный аффикс из списка $T_1' = T_1 \cup T_2 \cup T_3 \cup T_4 \cup T_5 \cup T_6 \cup T_7 \cup T_8 \cup T_9 \cup T_{10} \cup T_{11} \cup T_{12} \cup T_{13} \cup T_{14} \cup T_{15}$.

Если найденный аффикс принадлежит списку T_1 и находится в области R_2, то удалить его.

Если найденный аффикс принадлежит списку T_2 и находится в области R_2, то удалить его. Если при этом аффиксу предшествует ic, то удалить данный аффикс, если он находится в области R_2, в противном случае заменить на iqU;

Если найденный аффикс принадлежит списку T_3 и находится в области R_2, то заменить его на log.

Если найденный аффикс принадлежит списку T_4 и находится в области R_2, то заменить его на u.

Если найденный аффикс принадлежит списку T_5 и находится в области R_2, то заменить его на ent.

Если найденный аффикс принадлежит списку T_6 и находится в области RV, то удалить его. При этом, если аффиксу предшествует iv, то удалить данный аффикс, если он находится в области R_2 (и если ему предшествует at, то также удалить данный аффикс, если он находится в области R_2); если аффиксу предшествует eus, то удалить данный аффикс, если тот находится в области R_2, иначе заменить на eux, если он находится в области R_1; если аффиксу предшествует abl или iqU, то удалить данный аффикс, если тот находится в области R_2; если аффиксу предшествует $i\grave{e}r$ или $I\grave{e}r$, заменить на i, если соответствующий аффикс находится в области RV.

Если найденный аффикс принадлежит списку T_7 и находится в области R_2, то удалить его. При этом, если аффиксу предшествует $abil$, то удалить данный аффикс, если тот находится в области R_2, в противном случае заменить на abl; если аффиксу предшествует ic, то удалить данный аффикс, если тот находится в области R_2, в противном случае заменить на iqU; если аффиксу предшествует iv, то удалить данный аффикс, если тот находится в области R_2.

Если найденный аффикс принадлежит списку T_8 и находится в области R_2, то удалить его. При этом, если аффиксу предшествует at, то удалить данный аффикс, если тот находится в области R_2 (и если ему предшествует ic, то удалить данный аффикс, если тот находится в области R_2, в противном случае заменить его на iqU).

Если найденный аффикс принадлежит списку T_9, то заменить его на eau.

Если найденный аффикс принадлежит списку T_{10} и находится в области R_1, то заменить его на al.

Если найденный аффикс принадлежит списку T_{11} и находится в области R_2, то удалить его.

Если найденный аффикс принадлежит списку T_{11} и находится в области R_1, то заменить его на eux.

Если найденный аффикс принадлежит списку T_{12} и находится в области R_1, а также ему предшествует согласная, то удалить данный аффикс.

Если найденный аффикс принадлежит списку T_{13} и находится в области RV, то заменить его на ant.

Если найденный аффикс принадлежит списку T_{14} и находится в области RV, то заменить его на ent.

Если найденный аффикс принадлежит списку T_{15} и находится в области RV, а также ему предшествует гласная, то удалить данный аффикс.

Далее в Шагах 2a и 2b все проверки осуществляются в области RV.

Шаг 2a выполняется, если ни один из суффиксов не был удален на шаге 1, либо найденный аффикс был одним из списка: $amment, emment, ment, ments$.

Шаг 2a:

Найти самый длинный аффикс из списка T_{16}, и если найденному аффиксу предшествует согласная, то удалить данный аффикс (согласная также должна находиться в области RV).

Шаг 2b выполняется, если после выполнения Шага 2a не был удален ни один суффикс.

Шаг 2b:

Найти самый длинный аффикс из списка $T'_{17} = T_{17} \cup T_{18} \cup T_{19}$.

Если найденный аффикс принадлежит списку T_{17} и находится в области R_1, то удалить его.

Если найденный аффикс принадлежит списку T_{18}, то удалить его.

Если найденный аффикс принадлежит списку T_{19}, то удалить его. При этом, если аффиксу предшествует e, то удалить эту букву, если она находится в области RV.

Если на одном из шагов 1, 2a или 2b анализируемая словоформа была изменена, то перейти к Шагу 3.

Шаг 3:

Заменить Y на конце слова на i или ç на конце слова на c.

Если на данном шаге псевдооснова не была изменена, то перейти к шагу 4.

Шаг 4:

Если слово оканчивается на s, которому не предшествует $a, i, o, u,$ è или s, то удалить s.

Все проверки далее осуществляются в области RV.

Найти самый длинный аффикс из списка $T'_{20} = T_{20} \cup T_{21} \cup T_{22} \cup T_{23}$.

Если найденный аффикс принадлежит списку T_{20} и находится в области R_2, а также ему предшествует s или t, то удалить найденный аффикс.

Если найденный аффикс принадлежит списку T_{21}, то заменить его на i.

Если найденный аффикс принадлежит списку T_{22}, то удалить его.

Если найденный аффикс принадлежит списку T_{23}, а также ему предшествует gu, то удалить его.

Шаг 5:

Если слово оканчивается на enn, onn, ett, ell или на $eill$, то удалить последнюю букву.

Шаг 6:

Если слово заканчивается на é или è, за которыми следует как минимум одна негласная, то заменить буквы на e.

В заключении нужно вернуть I, U, Y в нижний регистр — этот шаг соответствует применению функций трансформации $f \in F_T$, обратных к тем, что были применены на предварительном этапе.

Полученный результат является искомой псевдоосновой анализируемой словоформы.

4.18. Чешский язык

Чешский язык является флективным языком с элементами аналитизма в спряжении глагола (формы прошедшего и будущего времени, а также страдательного залога).

В чешском языке следующие буквы являются гласными: $a, e, i, o, u, y,$ á, é, ě, í, ó, ú, ů, ý.

Определим R_1 как часть слова за первой негласной (согласной или заглавной гласной), следующей за гласной, или конец слова, если такой негласной нет.

Область RV определяется следующим образом: если в слове вторая буква согласная, то RV — это часть слова после следующей гласной; если первые две буквы гласные, то RV — часть слова после следующей согласной буквы, иначе, если первая — согласная, а вторая — гласная, то RV — это область слова после третьей буквы. Но в данных условиях RV не может начинаться в конце слова.

Для чешского языка в терминах обобщенного алгоритма определяются следующие функции $f \in F_C$:

- Буква (или последовательность букв) располагается в области R_1;
- Буква (или последовательность букв) располагается в области RV.

В соответствии со схемой обобщенного алгоритма анализа будут сформированы списки аффиксов $T_1, \dots, T_5$:

- $T_1 = (atech, ětem, atům, ách, ých, ové, ými, ata, aty, ama, ami, ovi, at, ám, os, us, ým, mi, ou, u, y, ů, a, o, á, é, ý)$;
- $T_2 = (ech, ich, ích, ého, ěmi, ému, ěte, ěti, ího, ími, emi, iho, imu, ém, ím, es, e, i, í, ě)$;
- $T_3 = (em)$;
- $T_4 = (ov, ův)$;
- $T_5 = (in)$;

Далее приводится описание алгоритма аналитического выделения основы.

Шаг 1:

Найти самый длинный аффикс из списка $T_1' = T_1 \cup T_2 \cup T_3$.

Если найденный аффикс принадлежит списку T_1, то удалить его.

Если найденный аффикс принадлежит списку T_2, то удалить его. Также выполнить процедуру «смягчения аффикса». В рамках данной процедуры выполнить следующие действия:

- Если псевдооснова оканчивается на $ci, ce, \check{c}i, \check{c}$, то заменить аффикс на k.
- Если псевдооснова оканчивается на $zi, ze, \check{z}i, \check{z}e$, то заменить аффикс на h.
- Если псевдооснова оканчивается на $\check{c}t\check{e}, \check{c}ti, \check{c}t\acute{e}$, то заменить аффикс на ck.
- Если псевдооснова оканчивается на $\check{s}t\check{e}, \check{s}ti, \check{s}t\acute{e}$, то заменить аффикс на sk.

Если найденный аффикс принадлежит списку T_3, то заменить его на e и выполнить процедуру «смягчения аффикса», описанную выше.

Шаг 2:

Найти самый длинный аффикс из списка $T_4' = T_4 \cup T_5$.

Если найденный аффикс принадлежит списку T_4 и находится в области RV, то удалить его.

Если найденный аффикс принадлежит списку T_5, то удалить его и выполнить процедуру «смягчения аффикса», описанную выше.

Полученный результат является искомой псевдоосновой анализируемой словоформы.

5. АЛГОРИТМЫ ВЫДЕЛЕНИЯ ОСНОВ III ТИПА

Требования к усвоению данного раздела учебного пособия:

Знать: иннновационные особенности различных видов печатных и электронных средств информации, основные технологические процессы производства печатных и электронных средств информации;

Уметь: использовать компьютерную технику в решении конкретных практических задач;

Владеть навыками: работы с прикладными программными средствами.

5.1. Азербайджанский язык

По своему морфологическому типу азербайджанский язык является агглютинативно-синтетическим языком.

Для азербайджанского языка в терминах обобщенного алгоритма функция проверки условия $f \in F_C$ будет соответствовать проверке того, что выделяемая псевдооснова не пуста, а также данная функция накладывает ограничение на количество возможных преобразований словоформы для получения ее псевдоосновы.

В соответствии со схемой обобщенного алгоритма анализа будет сформирован единый список аффиксов T_1:

- $T_1 = (lar, lər, m, ım, im, um, $ üm$, mız, miz, muz, m$üz$, ımız, imiz,$
  $umuz,$ ümüz$, n, ın, in, un,$ ün$, nız, niz, nuz, n$üz$, ınız, iniz, unuz,$ ünüz$,$
  $ız, iz, uz,$ üz$, ı, i, u,$ ü$, sı, si, su, s$ü$, ları,ləri, ın, in, un,$ ün$, nın, nin, nun,$
  $n$ün$, a, ə, ya, yə, da, də, dan, dən, ı, i, u,$ ü$, nı, ni, nu, n$ü$, am, əm, yam,$
  $yəm, ıq, ik, uq,$ üq$, yıq, yik, yuq, y$ük$, san, sən, sınız, siniz, sunuz,$
  $s$ünüz$, sız, siz, suz, s$üz$, dır, dir, dur, d$ür$, dırlar, dirlər, durlar,$
  $d$ürlər$, am, əm, yam, yəm, ıq, ik, uq,$ üq$, yıq, yik, yuq, y$ük$, maq, mək,$
  $ın, in, un,$ ün$, yın, yin, yun, y$ün$, sa, sə, ma, mə, r, ar, ər, yar, yər, ır, ir,$
  $ur,$ ür$, yır, yir, yur, y$ür$, acaq, əcək, yacaq, yəcək, mış, miş, muş, m$üş$,$
  $dı, di, du, d$ü$, mı, mi, mu, m$ü$, maz, məz, m, n, z, m, am, əm, yam, yəm,$
  $aq, ək, yaq, yək, ası, əsi, malı, məli, maqda, məkdə, sa, sə, raq, rək,$

$dan, dən, ca, cə, ılır, araq, la, lə, lidir$);

Далее приводится описание алгоритма аналитического выделения основы.

Шаг 1:

Найти самый длинный аффикс из списка T_1 и удалить его. Повторить шаг 1, пока происходит модификация псевдоосновы, но не более трех раз.

Полученный результат является искомой псевдоосновой анализируемой словоформы.

5.2. Армянский язык

Морфологический строй армянского языка преимущественно агглютинативный, но встречаются отдельные элементы аналитизма.

В армянском языке следующие буквы являются гласными: *ա, է, ի, օ, ı, ե, ո, ը*.

Определим R_1 как часть слова за первой негласной (согласной или заглавной гласной), следующей за гласной, или конец слова, если такой негласной нет. R_2 — часть слова за первой негласной, следующей за гласной, находящейся в R_1, или конец слова, если такой негласной нет.

Область RV определяется следующим образом: если в слове вторая буква согласная, то RV — это часть слова после следующей гласной; если первые две буквы гласные, то RV — часть слова после следующей согласной буквы, иначе, если первая — согласная, а вторая — гласная, то RV — это область слова после третьей буквы. Но в данных условиях RV не может начинаться в конце слова.

Для армянского языка в терминах обобщенного алгоритма определяются следующие функции $f \in F_C$:

- Буква (или последовательность букв) располагается в области R_2;
- Буква (или последовательность букв) располагается в области RV.

В соответствии со схемой обобщенного алгоритма анализа будут сформированы списки аффиксов $T_1, \dots, T_4$:

- $T_1 = ($*բար, պես, որէն, ովին, ակի, լայն, րորդ, երորդ, ալի, որակ, եղ, ական, արան, են, եկեն, երեն, ին, գին, վուն, իվ, ատ, ավետ, կոտ,*$)$;
- $T_2 = ($*ա, ացա, եցա, վե, ացրի, ացի, եցի, վեցի, ալ, ըալ, անալ, ենալ, ացնալ, ել, ըել, նել, ցնել, եցնել, չել, վել, ացվել, եցվել, տել, ատել, ոտել, կոտել, վածˊ, ում, վում, ան, ցան, ացան, ացրին, ացին, եցին, վեցին, ալիս, ելիս, ավ, ացավ, եցավ, ալով, ելով, ար, ացար, եցար, ացրիր, ացիր, եցիր, վեցիր, աց, եց, ացրեց, ալուց, ելուց, ալու, ելու, աք, ցաք, ացաք, ացրիք, ացիք, եցիք, վեցիք, անք, ցանք, ացանք, ացրինք, ացինք, եցինք, վեցինք,*$)$;
- $T_3 = ($*որդ, ույթ, ուհի, ցի, իլ, ակ, յակ, անակ, իկ, ուկ, ան, պան, ստան, արան, յուն, ությունˊ, ածո, իչ, ուս, ուստ, զար, վոր, ավոր, ոց, անօց, ու, ք, չէք, իք, ալիք, անիք, վածք, ույք, ենք, ոնք, ունք, մունք, իչք, արք, եղէն,*$)$;
- $T_4 = ($*սա, վա, ամբ, դ, անդ, ությանդ, վանդ, ոջդ, երդ, ներդ, ուդ, ը, անը, ությանը, վանը, ոջը, երը, ները, ի, վի, երի, ների, անում, երում, ներում, ն, ան, ության, ան, ին, երին, ներին, ությանն, երն, ներն, ուն, ոջ, ությանս, վանս, ոջս, ով, անով, վով, երով, ներով, եր, ներ, ց, ից, վանից, ոջից, վից, երից, ներից, ցից, ոց, ուց,*$)$;

Далее приводится описание алгоритма аналитического выделения основы.

Шаг 1:

Найти самый длинный аффикс из списка T_1, и если он находится в области RV, то удалить его.

Шаг 2:

Найти самый длинный аффикс из списка T_2, и если он находится в области RV, то удалить его.

Шаг 3:

Найти самый длинный аффикс из списка T_3, и если он находится в области RV, то удалить его.

Шаг 4:

Найти самый длинный аффикс из списка T_2, и если он находится в области RV и в области R_2, то удалить его.

Полученный результат является искомой псевдоосновой анализируемой словоформы.

5.3. Венгерский язык

По своему морфологическому типу венгерский язык является агглютинативным языком, но ему присущи и некоторые элементы флективности. Аналитические структуры в венгерском языке представлены ограниченно.

Буквы в венгерском языке включают следующие формы с акцентом: á, é, í, ó, ö, õ, ú, ü, û.

Следующие буквы гласные: a, á, e, é, i, í, o, ó, ö, õ, u, ú, ü, û.

Определим R_1 как часть слова за первой негласной (согласной или заглавной гласной), следующей за гласной, или конец слова, если такой негласной нет.

Определим для венгерского языка список удвоенных согласных: $bb, cc, ccs, dd, ff, gg, ggy, jj, kk, ll, lly, mm, nn, nny, pp, rr, ss, ssz, tt, tty, vv,$

zz, zzs.

Для венгерского языка в терминах обобщенного алгоритма определяются следующие функции $f \in F_C$:

- Буква (или последовательность букв) располагается в области R_1;
- Буква (или последовательность букв) располагается после удвоенной согласной из соответствующего списка.

В соответствии со схемой обобщенного алгоритма анализа будут сформированы списки аффиксов $T_1, \dots, T_{20}$:

- $T_1 = (al, el)$;
- $T_2 = (ban, ben, ba, be, ra, re, nak, nek, val, vel, tól, től, ról, ről, ból,$ $ből, hoz, hez, höz, nál, nél, ig, at, et, ot, öt, ért, képp, képpen, kor, ul,$ $ül, vá, vé, onként, enként, anként, ként, en, on, an, ön, n, t)$;
- $T_3 = (án, ánként)$;
- $T_4 = (én)$;
- $T_5 = (astul, estül, stul, stül)$;
- $T_6 = (ástul)$;
- $T_7 = (éstül)$;
- $T_8 = (á, é)$;
- $T_9 = (oké, öké, aké, eké, ké, éi, é)$;
- $T_{10} = (áké, áéi)$;
- $T_{11} = (éké, ééi, éé)$;
- $T_{12} = (ünk, unk, nk, juk, jük, uk, ük, em, om, am, m, od, ed, ad, öd, d,$ $ja, je, a, e, o)$;
- $T_{13} = (ánk, ájuk, ám, ád, á)$;
- $T_{14} = (énk, éjük, ém, éd, é)$;

- $T_{15} = (jaim, jeim, aim, eim, im, jaid, jeid, aid, eid, id, jai, jei, ai, ei, i, jaink, jeink, eink, aink, ink, jaitok, jeitek, aitok, eitek, itek, jeik, jaik, aik, eik, ik$);
- $T_{16} = (\acute{a}im, \acute{a}id, \acute{a}i, \acute{a}ink, \acute{a}itok, \acute{a}ik)$;
- $T_{17} = (\acute{e}im, \acute{e}id, \acute{e}i, \acute{e}ink, \acute{e}itek, \acute{e}ik)$;
- $T_{18} = (\ddot{o}k, ok, ek, ak, k)$;
- $T_{19} = (\acute{a}k)$;
- $T_{20} = (\acute{e}k)$;

Далее приводится описание алгоритма аналитического выделения основы.

Шаг 1:

Найти аффикс из списка T_1. Если найденный аффикс находится в области R_1 и ему предшествует удвоенная согласная из соответствующего списка, то удалить найденный аффикс и удалить одну согласную.

Шаг 2:

Найти самый длинный аффикс из списка T_2 и удалить его, если он находится в области R_1. Если полученная после удаления аффикса псевдооснова оканчивается на $\acute{a}(\acute{e})$, то заменить последнюю букву на $a(e)$.

Шаг 3:

Найти самый длинный аффикс из списка $T_3' = T_3 \cup T_4$.

Если найденный аффикс принадлежит списку T_3 и находится в области R_1, то заменить его на a.

Если найденный аффикс принадлежит списку T_4 и находится в области R_1, то заменить его на e.

Шаг 4:

Найти самый длинный аффикс из списка $T_5' = T_5 \cup T_6 \cup T_7$.

Если найденный аффикс принадлежит списку T_5 и находится в области R_1, то удалить его.

Если найденный аффикс принадлежит списку T_6 и находится в области R_1, то заменить его на a.

Если найденный аффикс принадлежит списку T_7 и находится в области R_1, то заменить его на e.

Шаг 5:

Найти аффикс из списка T_8. Если найденный аффикс находится в области R_1 и ему предшествует удвоенная согласная из соответствующего списка, то удалить найденный аффикс и удалить одну согласную.

Шаг 6:

Найти самый длинный аффикс из списка $T_9' = T_9 \cup T_{10} \cup T_{11}$.

Если найденный аффикс принадлежит списку T_9 и находится в области R_1, то удалить его.

Если найденный аффикс принадлежит списку T_{10} и находится в области R_1, то заменить его на a.

Если найденный аффикс принадлежит списку T_{11} и находится в области R_1, то заменить его на e.

Шаг 7:

Найти самый длинный аффикс из списка $T_{12}' = T_{12} \cup T_{13} \cup T_{14}$.

Если найденный аффикс принадлежит списку T_{12} и находится в области R_1, то удалить его.

Если найденный аффикс принадлежит списку T_{13} и находится в области R_1, то заменить его на a.

Если найденный аффикс принадлежит списку T_{14} и находится в области R_1, то заменить его на e.

Шаг 8:

Найти самый длинный аффикс из списка $T'_{15} = T_{15} \cup T_{16} \cup T_{17}$.

Если найденный аффикс принадлежит списку T_{15} и находится в области R_1, то удалить его.

Если найденный аффикс принадлежит списку T_{16} и находится в области R_1, то заменить его на a.

Если найденный аффикс принадлежит списку T_{17} и находится в области R_1, то заменить его на e.

Шаг 9:

Найти самый длинный аффикс из списка $T'_{18} = T_{18} \cup T_{19} \cup T_{20}$.

Если найденный аффикс принадлежит списку T_{18} и находится в области R_1, то удалить его.

Если найденный аффикс принадлежит списку T_{19} и находится в области R_1, то заменить его на a.

Если найденный аффикс принадлежит списку T_{20} и находится в области R_1, то заменить его на e.

Полученный результат является искомой псевдоосновой анализируемой словоформы.

5.4. Казахский язык

Казахский язык относится к агглютинативным синтетическим языкам с элементами аналитизма. Аналитические формы характерны преимущественно для глаголов.

Для казахского языка в терминах обобщенного алгоритма функция проверки условия $f \in F_C$ будет соответствовать проверке того, что выделяемая псевдооснова не пуста, а также данная функция накладывает ограничение на количество возможных преобразований словоформы для получения ее псевдоосновы.

В соответствии со схемой обобщенного алгоритма анализа будет сформирован единый список аффиксов T_1:

- $T_1 =$ (дар, дер, тар, тер, лар, лер, м, ым, ім, мыз, ымыз, імыз, міз, ыміз, іміз, *ң*, ы*ң*, і*ң*, *ң*ыз, ы*ң*ыз, і*ң*ыз, *ң*із, ы*ң*із, і*ң*із, сы, сі, ы, і, дан, ден, тан, тен, нан, нен, да, де, та, те, нда, нде, *ғ*а, ге, *қ*а, ке, на, не, а, е, ды*ң*, ді*ң*, ты*ң*, ті*ң*, ны*ң*, ні*ң*, ды, ді, ты, ті, ны, ні, н, бен, пен, мен, дікі, тікі, нікі, мын, мін, мыз, міз, сын, сін, сыз, сіз, сындар, сіндер, сыздар, сіздер, ып, іп, п, а, е, й, *қ*, к, *ң*, *ң*дар, *ң*дер, *ң*ыз, *ң*із, *ң*ыздар, *ң*іздер, са, се, *ғ*ы, *ғ*і, *қ*ы, *қ*і, йын, айын, ейын, йін, айін, ейін, йы*қ*, айы*қ*, ейы*қ*, йік, айік, ейік, ныздар, ніздер, ыныздар, ыніздер, іныздар, ініздер, ндар, ндер, ысын, ысін, ісын, ісін, ар, ер, р, атын, етін, йтын, йтін, быз, біз, сы*ң*, сі*ң*, сы*ң*дар, сі*ң*дер, сыз, сіз, пын, пін, пыз, піз, ба*қ*, бек, па*қ*, пек, ма*қ*, мек, *ғ*ан, ген, *қ*ан, кен);

Далее приводится описание алгоритма аналитического выделения основы.

Шаг 1:

Найти самый длинный аффикс из списка T_1 и удалить его. Повторить шаг 1, пока происходит модификация псевдоосновы, но не более трех раз.

Полученный результат является искомой псевдоосновой анализируемой словоформы.

5.5. Турецкий язык

По своему морфологическому строю турецкий язык относится к агглютинативным языкам.

В турецком языке аффиксы имен существительных можно разделить на две группы:

- падежные аффиксы;
- аффиксы принадлежности.

При этом аффиксы глаголов можно разделить на три больших класса:

- аффиксы времени;
- аффиксы лица;
- аффиксы наклонения.

Рассматриваемая морфологическая модель анализирует имена существительные и глаголы, с учетом словоизменения.

В турецком языке аффиксы добавляются к основе с помощью конечного числа правил. Это позволяет представить морфологическую модель в виде конечного автомата. Состояние, помеченное символом «А», является стартовым. Терминальные состояния отмечены двойными окружностями. На дугах конечного автомата будем указывать номер суффикса (приведены далее в списках), который позволяет перейти из текущего состояния в следующее. При проходе по автомату, как только мы достигаем терминального состояния, то

проверяем, есть ли переход из этого состояния, который можно осуществить при выполнении некоторого условия. Если переход совершить невозможно, то анализ словоформы завершается, иначе — происходит переход в следующее состояние.

Можно выделить девятнадцать словоизменительных аффиксов имени существительного. Аффикс мы будем рассматривать с учетом алломорфов. Например, аффикс $-DAn$ имеет четыре алломорфа: dan, den, tan, ten. Часть аффикса, заключенная в круглые скобки, указывает на опциональность ее появления. Словоизменительные аффиксы имени существительного приведены в таблице 5.1. Личные и временные аффиксы, образующие формы глаголов в турецком языке, приведены в таблице 5.2.

Таблица 5.1. Словоизменительные аффиксы имени существительного в турецком языке.

Номер	**Аффикс**	**Тип**	**Варианты написания с учетом алломорфов**
1	lAr	Аффикс множественного числа	lar, ler
2	(U)m	Аффиксы принадлежности	m, ım, im, um, üm
3	(U)mUz		mız, miz, muz, müz, ımız, imiz, umuz, ümüz
4	(U)n		n, ın, in, un, ün
5	(U)nUz		nız, niz, nuz, nüz, ınız, iniz, unuz, ünüz
6	(s)U		ı, i, u, ü, sı, si, su, sü
7	lArI		ları, leri, lari, leri
8	y(U)	Падежные аффиксы (с учетом изафета)	y, yı, yi, yu, yü
9	n(U)		n, nı, ni, nu, nü
10	(n)Un		ın, in, un, ün, nın, nin, nun, nün
11	(y)A		a, e, ya, ye
12	nA		na, ne

13	DA		da, ta, de, te
14	nDA		nda, nta, nde, nte
15	DAn		dan, tan, den, ten
16	nDAn		ndan, ntan, nden, nten
17	(y)lA		la, le, yla, yle
18	ki	Особая форма указания на предмет	ki
19	(n)cA	Орудный (творительный) падеж	ca, ce, nca, nce

На рисунке 5.1 представлен конечный автомат для выделения основы имен существительных. Состояния автомата представляют собой словоформу после отсечения аффикса, записанного на входящем ребре. Номера на ребрах соответствуют аффиксам, приведенным в таблице 1. Начальное состояние «A». Терминальные состояния, попадая в которые, следует прекратить анализ: «A», «B», «E», «G», «H», «K», «L», «M». Переход осуществляется при наличии соответствующего аффикса и выполнении заданных условий (например, условия гармонии гласных).

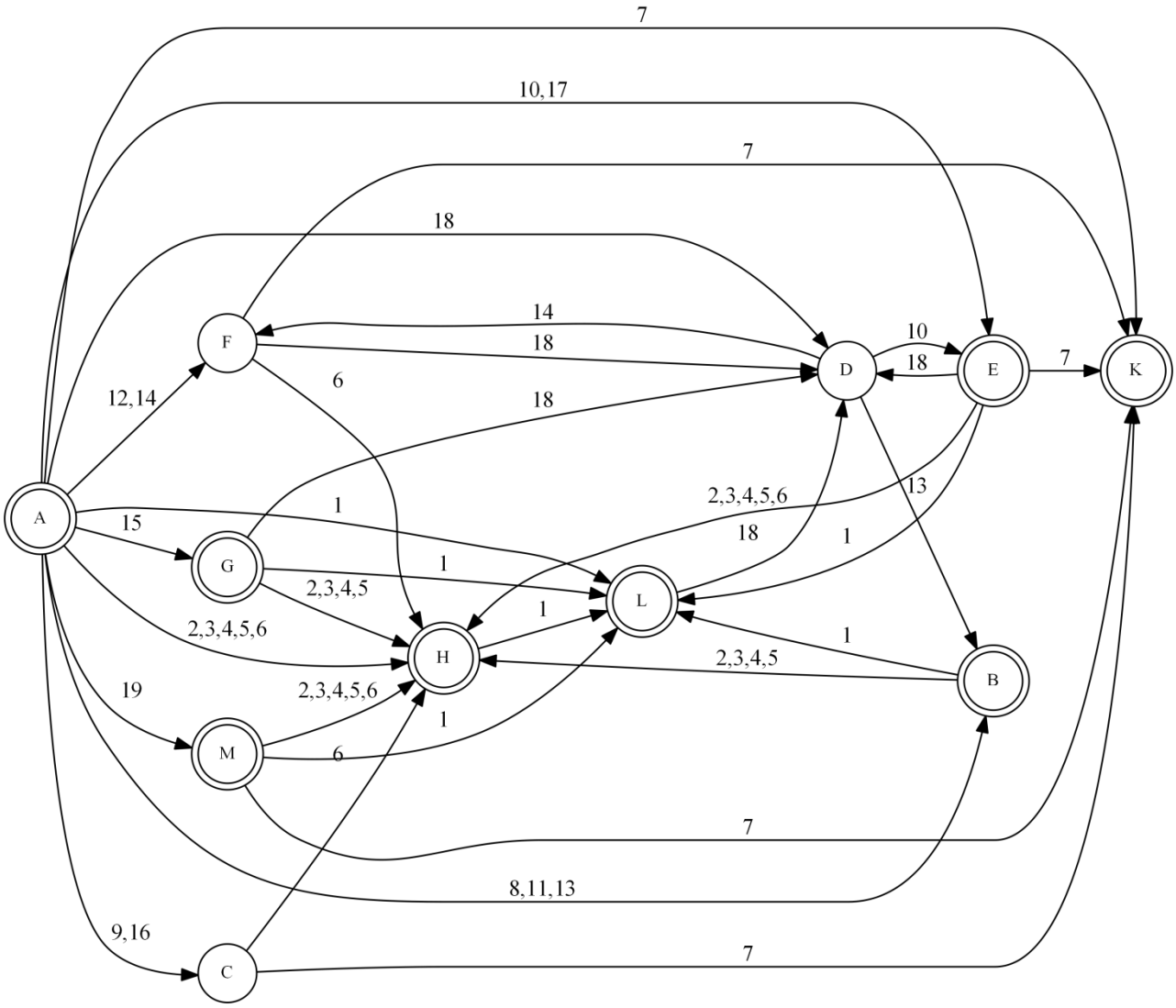

Рисунок 5.1. Конечный автомат для отсечения словоизменительных аффиксов имени существительного в турецком языке.

Таблица 5.2. Словоизменительные аффиксы глаголов в турецком языке.

Номер	**Аффикс**	**Тип**	**Варианты написания с учетом алломорфов**
1	(y)Um	Личные аффиксы	ım, im, um, üm, yım, yim, yum, yüm
2	sUn		sın, sin, sun, sün
3	(y)Uz		ız, iz, uz, üz, yız, yiz, yuz, yüz
4	sUnUz		sınız, siniz, sunuz, sünüz
5	lAr		lar, ler

6	mUş	Аффиксы времени	mış, miş, muş, müş
7	(y)AcAk		acak, ecek, yacak, yecek
8	(U)r		r, ır, ir, ur, ür
9	Ar		ar, er
10	(U)yor		yor, ıyor, iyor, uyor, üyor
11	mAktA		makta, mekte
12	mAlI	Аффикс наклонения	malı, melı, malI, melI
13	m	Личные аффиксы	m
14	n		n
15	k		k
16	nUz		nız, niz, nuz, nüz
17	DU	Аффикс времени	dı, di, du, dü, tı, ti, tu, tü
18	sA	Аффикс наклонения	sa, se
19	lIm		lım, lIm
20	(y)A		a, e, ya, ya
21	(y)UnUz		ınız, iniz, unuz, ünüz, yınız, yiniz, yunuz, yünüz
22	(y)Un		ın, in, un, ün, yın, yin, yun, yün
23	sUnlAr		sınlar, sinlar, sunlar, sünlar, sınler, sinler, sunler, sünler
24	DUr	Глагол «быть»	dır, dir, dur, dür, tır, tir, tur, tür
25	(y)DU	Аффиксы наклонения	dı, di, du, dü, tı, ti, tu, tü, ydı, ydi, ydu, ydü, ytı, yti, ytu, ytü
26	(y)sA		sa, se, ysa, yse
27	(y)mUş		mış, miş, muş, müş, ymış, ymiş, ymuş, ymüş
28	cAsInA	Аффиксы деепричастия	casına, cesıne, casIna, cesIne
29	(y)ken		ken, yken

На рисунке 5.2 представлен конечный автомат для выделения основы глаголов. По аналогии с автоматом, приведенным на рисунке 1, состояния — это словоформа после отсечения аффикса, записанного на входящем ребре. Номера на ребрах соответствуют аффиксам, приведенным в таблице 2. Начальное состояние «A». Терминальные состояния, попадая в которые,

следует прекратить анализ: «A», «C», «G», «L», «O», «Q». Переход осуществляется при наличии соответствующего аффикса и выполнении заданных условий (например, условия гармонии гласных).

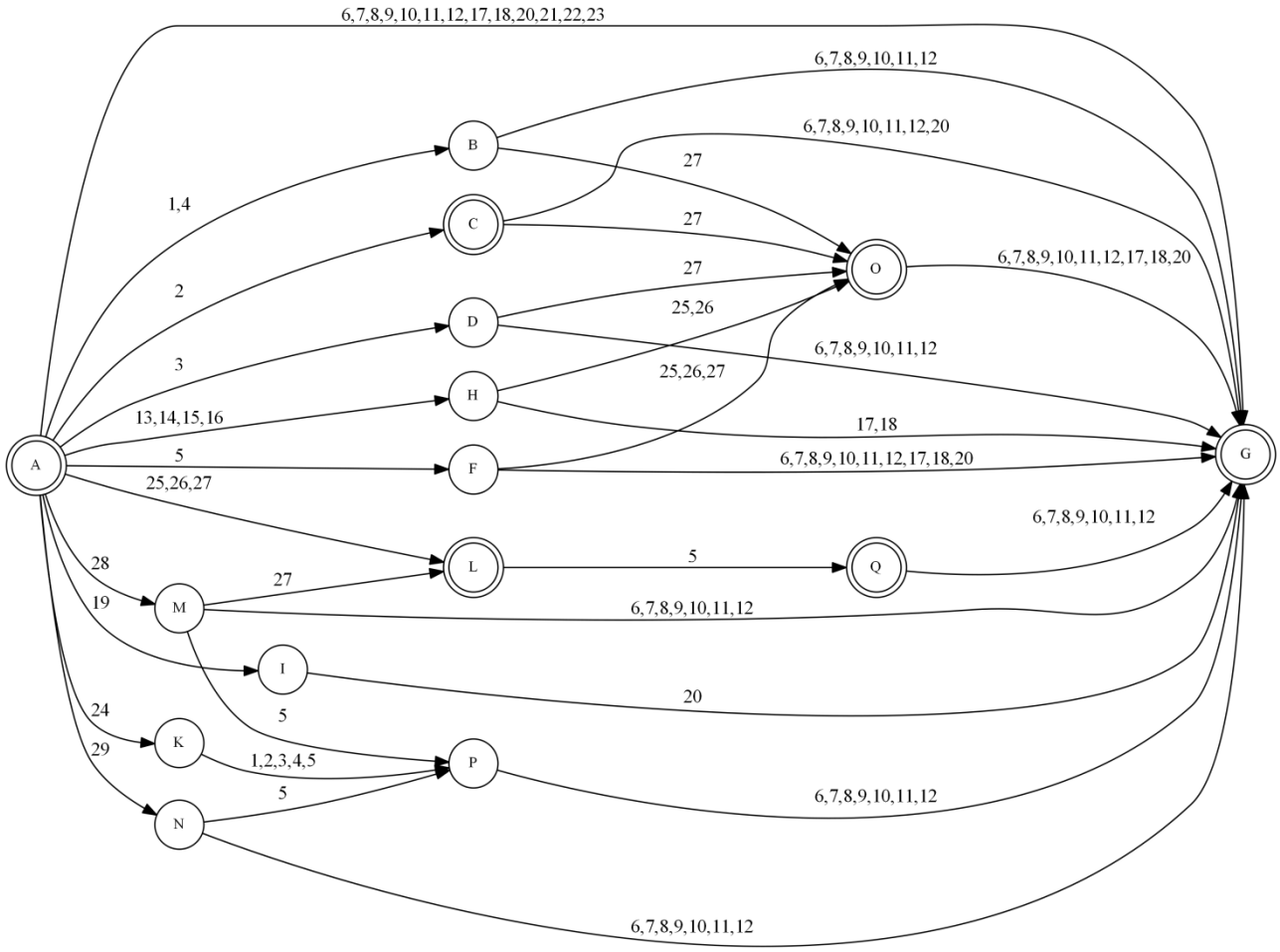

Рисунок 5.2. Конечный автомат для отсечения словоизменительных (лицо, время, наклонение) аффиксов глагола в турецком языке.

Боры, используемые в рамках обобщенного алгоритма автоматического анализа словоформ, формируются с учетом всех алломорфов суффиксов.

Для турецкого языка в терминах обобщенного алгоритма функция проверки условия $f \in F_C$ будет соответствовать проверке того, что выделяемая псевдооснова не пуста, а также данная функция накладывает ограничение на

количество возможных преобразований словоформы для получения ее псевдоосновы.

В соответствии со схемой обобщенного алгоритма анализа будет сформирован единый список аффиксов T_1:

- $T_1 = (lar, ler, m, ım, im, um, $ üm$, mız, miz, muz, m$üz$, ımız, imiz,$ $umuz,$ ümüz$, n, ın, in, un,$ ün$, nız, niz, nuz, n$üz$, ınız, iniz, unuz,$ ünüz$,$ $ı, i, u,$ ü$, sı, si, su, s$ü$, ları, leri, lari, leri, y, yı, yi, yu, y$ü$, nı, ni, nu, n$ü$, nın,$ nin, nun, nün$, a, e, ya, ye, na, ne, da, ta, de, te, nda, nta, nde, nte, dan,$ $tan, den, ten, ndan, ntan, nden, nten, la, le, yla, yle, ki, ca, ce, nca, nce,$ $yım, yim, yum, y$üm$, sın, sin, sun, s$ün$, ız, iz, uz,$ üz$, yız, yiz, yuz, y$üz$,$ $sınız, siniz, sunuz, s$ünüz$, mış, miş, muş, m$üş$, acak, ecek, yacak, yecek,$ $r, ır, ir, ur,$ ür$, ar, er, yor, ıyor, iyor, uyor,$ ü$yor, makta, mekte, malı,$ $melı, malI, melI, k, dı, di, du, d$ü$, tı, ti, tu, t$ü$, sa, se, lım, lIm, yınız, yiniz,$ $yunuz, y$ünüz$, yın, yin, yun, y$ün$, sınlar, sinlar, sunlar, s$ünlar$, sınler,$ $sinler, sunler, s$ünler$, dır, dir, dur, d$ür$, tır, tir, tur, t$ür$, ydı, ydi, ydu,$ ydü$, ytı, yti, ytu, yt$ü$, ysa, yse, ymış, ymiş, ymuş, ym$üş$, casına, cesıne,$ $casIna, cesIne, ken, yken,$);

Далее приводится описание алгоритма аналитического выделения основы.

Шаг 1:

Найти самый длинный аффикс из списка T_1 и удалить его. Повторить шаг 1, пока происходит модификация псевдоосновы, но не более трех раз, осуществляя проверку условий согласованности гласных и порядка следования аффиксов (на основе описанных автоматов), определенных в начале работы алгоритма.

Полученный результат является искомой псевдоосновой анализируемой словоформы.

5.6. Финский язык

Морфологический строй финского языка представляет собой преимущественно агглютинативный тип, хотя ему присущи и многие элементы флективности. Аналитические структуры представлены довольно широко, хотя вследствие высокой степени синтетизма широко употребительными являются словоформы с многочисленными морфемами.

Буквы в финском языке включают следующие формы с акцентом: ä, ö.

Следующие буквы гласные: a, e, i, o, u, y, ä, ö.

Определим R_1 как часть слова за первой негласной (согласной или заглавной гласной), следующей за гласной, или конец слова, если такой негласной нет. R_2 — часть слова за первой негласной, следующей за гласной, находящейся в R_1, или конец слова, если такой негласной нет.

Определим список «длинных гласных» $LV = (aa, ee, ii, oo, uu,$ ää, öö$)$.

Определим список «ограниченных гласных» $V = (a, e, i, o, u,$ ä, ö$)$.

В финском языке морфемную структуру имени существительного можно представить следующей схемой:

Рисунок 5.3. Морфемная структура имени существительного.

Для финского языка можно выделить 33 словоизменительных аффикса, разделенных в соответствии с грамматическим предназначением (см. таблицу 5.3).

На рисунке 5.4 приведен конечный автомат для отсечения словоизменительных аффиксов имени существительного. Заглавная буква O обозначает одну из гласных $o, ö$, аналогично A обозначает $a, ä$. Состояния — это словоформа после отсечения аффикса, записанного на входящем ребре. Номера на ребрах соответствуют аффиксам, приведенным в таблице 5.3. Начальное состояние «A». Терминальные состояния, попадая в которые следует прекратить анализ: «A», «B», «C», «D», «E», «F», «G», «H», «I», «J», «K», «L».

Таблица 5.3. Словоизменительные аффиксы имени существительного в финском языке.

Номер	**Аффикс**	**Тип**
1	kO	Энклитические частицы
2	kin	
3	kAAn	
4	kA	
5	pA	
6	hAn	
7	ni	Притяжательные суффиксы
8	si	
9	mme	
10	nne	
11	nsA	
12	An	
13	en	
14	hVn	Падежные суффиксы
15	siin	
16	den	

17	tten	
18	seen	
19	A	
20	ttA	
21	tA	
22	ssA	
23	stA	
24	llA	
25	ltA	
26	lle	
27	nA	
28	ksi	
29	ine	
30	n	
31	i	Суффиксы множественного числа
32	j	
33	t	

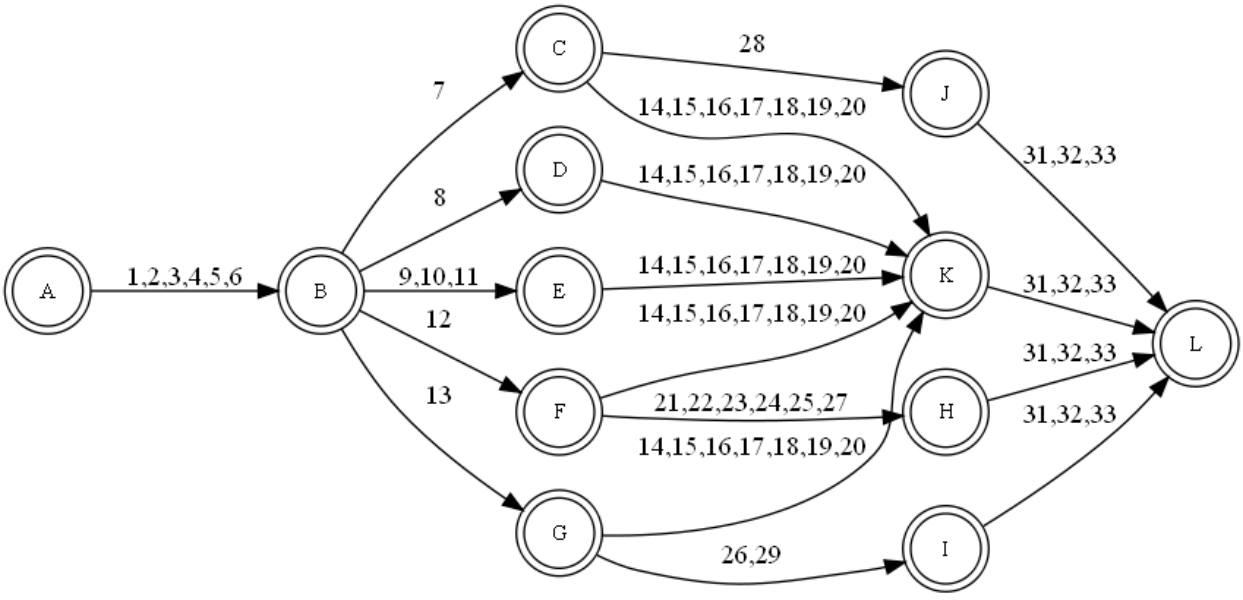

Рисунок 5.4. Конечный автомат для отсечения словоизменительных аффиксов имени существительного.

Для финского языка в терминах обобщенного алгоритма определяются следующие функции $f \in F_C$:

- Буква (или последовательность букв) располагается в области R_1;
- Буква (или последовательность букв) располагается в области R_2;
- Буква (или последовательность букв) находится после пары букв cv, где c — согласная буква, v — гласная буква;
- Буква (или последовательность букв) находится после сочетания букв из списка LV.

В соответствии со схемой обобщенного алгоритма анализа будут сформированы списки аффиксов $T_1, \dots, T_{17}$:

- $T_1 = (kin, kaan, kään, ko, kö, han, hän, pa, pä)$;
- $T_2 = (sti)$;
- $T_3 = (si)$;
- $T_4 = (ni)$;
- $T_5 = (nsa, nsä, mme, nne)$;
- $T_6 = (an)$;
- $T_7 = (än)$;
- $T_8 = (en)$;
- $T_9 = (han, hen, hin, hon, hän, hön)$;
- $T_{10} = (siin, den, tten)$;
- $T_{11} = (seen)$;
- $T_{12} = (ä, a)$;
- $T_{13} = (tta, ttä)$;
- $T_{14} = (ta, ssa, tä, ssä, sta, stä, lla, llä, lta, ltä, lle, na, nä, ksi, ine)$;
- $T_{15} = (n)$;
- $T_{16} = (mpi, mpa, mpä, mmi, mma, mmä)$;
- $T_{17} = (impi, impa, impä, immi, imma, immä, eja, ejä)$;

Далее приводится описание алгоритма аналитического выделения основы.

Шаг 1:

Найти самый длинный аффикс из списка $T_1' = T_1 \cup T_2$.

Если найденный аффикс принадлежит списку T_1 и находится в области R_1, а также найденному аффиксу предшествует n, t или гласная, то удалить его.

Если найденный аффикс принадлежит списку T_2 и находится в области R_2, то удалить его.

Шаг 2:

Найти самый длинный аффикс из списка $T_3' = T_3 \cup T_4 \cup T_5 \cup T_6 \cup T_7 \cup T_8$.

Если найденный аффикс принадлежит списку T_3 и находится в области R_1, а также найденному аффиксу не предшествует k, то удалить его.

Если найденный аффикс принадлежит списку T_4 и находится в области R_1, а также найденному аффиксу предшествует kse, то удалить аффикс, а kse заменить на ksi.

Если найденный аффикс принадлежит списку T_5 и находится в области R_1, то удалить его.

Если найденный аффикс принадлежит списку T_6 и находится в области R_1, то удалить его, если найденному аффиксу предшествует один из аффиксов: $ta, ssa, sta, lla, lta, na$.

Если найденный аффикс принадлежит списку T_7 и находится в области R_1, то удалить его, если найденному аффиксу предшествует один из аффиксов: $tä, ssä, stä, llä, ltä, nä$.

Если найденный аффикс принадлежит списку T_8 и находится в области R_1, то удалить его, если найденному аффиксу предшествует один из аффиксов: lle, ine.

Шаг 3:

Найти самый длинный аффикс из списка $T_9' = T_9 \cup T_{10} \cup T_{11} \cup T_{12} \cup T_{13} \cup T_{14} \cup T_{15}$.

Если найденный аффикс принадлежит списку T_9 и находится в области R_1, а также найденному аффиксу предшествует гласная из середины найденного аффикса, то удалить найденный аффикс.

Если найденный аффикс принадлежит списку T_{10} и находится в области R_1, а также найденному аффиксу предшествует сочетание букв Vi, где V — это гласная из списка «ограниченных гласных» V, то удалить найденный аффикс.

Если найденный аффикс принадлежит списку T_{11} и находится в области R_1, а также найденному аффиксу предшествует сочетание букв из списка LV, то удалить найденный аффикс.

Если найденный аффикс принадлежит списку T_{12} и находится в области R_1, а также найденному аффиксу предшествует сочетание букв cv, где c — согласная буква, v — гласная буква, то удалить найденный аффикс.

Если найденный аффикс принадлежит списку T_{13} и находится в области R_1, а также найденному аффиксу предшествует e, то удалить найденный аффикс.

Если найденный аффикс принадлежит списку T_{14} и находится в области R_1, то удалить его.

Если найденный аффикс принадлежит списку T_{15} и находится в области R_1, то удалить его. Помимо этого, если найденному аффиксу предшествует сочетание букв из списка LV или ie, то удалить последнюю гласную.

Шаг 4:

Найти самый длинный аффикс из списка $T_{16}' = T_{16} \cup T_{17}$.

Если найденный аффикс принадлежит списку T_{16} и находится в области R_2, а также найденному аффиксу не предшествует po, то удалить его.

Если найденный аффикс принадлежит списку T_{17} и находится в области R_2, то удалить его.

Шаг 5:

Если на шаге 2 псевдооснова была изменена, то удалить последнюю i или j, если данная буква находится в области R_1. В противном случае удалить последнюю t, если она находится в области R_1, если ей предшествует гласная. Если t была удалена, то удалить аффикс mma или $imma$, если он находится в области R_2, кроме тех случаев, когда аффиксу mma предшествует аффикс po.

Шаг 6:

Все условия далее проверяются последовательно.

Если область R_1 оканчивается сочетанием букв из списка LV, то удалить последнюю букву.

Если область R_1 оканчивается сочетанием букв вида cX, где c — это согласная, а X — одна из букв $a, ä, e, i$, то удалить последнюю букву.

Если область R_1 оканчивается на oj или uj, то удалить последнюю букву.

Если область R_1 оканчивается на jo, то удалить последнюю букву.

Если слово оканчивается на удвоенную согласную, за которой располагается 0 и более гласных, то удалить одну из согласных.

Полученный результат является искомой псевдоосновой анализируемой словоформы.

5.7. Эстонский язык

Эстонский язык в основном агглютинативный, однако в нем в достаточной степени проявляются и флективные черты. Аналитические морфологические формы представлены незначительно.

Буквы в эстонском языке включают следующие формы с акцентом: ä, ö, ü, õ, š, ž.

Следующие буквы гласные: a, e, i, o, u, y, ä, ö, ü, õ.

Определим R_1 как часть слова за первой негласной (согласной или заглавной гласной), следующей за гласной, или конец слова, если такой негласной нет. R_2 — часть слова за первой негласной, следующей за гласной, находящейся в R_1, или конец слова, если такой негласной нет.

Определим список «длинных гласных» $LV = (aa, ee, ii, oo, uu,$ ää, öö, õõ, üü).

Для эстонского языка в терминах обобщенного алгоритма определяются следующие функции $f \in F_C$:

- Буква (или последовательность букв) располагается в области R_1;
- Буква (или последовательность букв) располагается в области R_2.

В соответствии со схемой обобщенного алгоритма анализа будут сформированы списки аффиксов $T_1, \dots, T_4$:

- $T_1 = (sti, ti, ldi, li, kesi, lt, mini, malt)$;
- $T_2 = (si, ni, nsa, nsä, mme, nne, an, än, en)$;
- $T_3 = (d, de, te, id, sid, e, i, u, sse, s, st, l, lt, le, na, ks, ga, ni, ta)$;
- $T_4 = (m, im)$;
- $T_5 = (i, j)$;

Далее приводится описание алгоритма аналитического выделения основы.

Шаг 1:

Найти самый длинный аффикс из списка T_1.

Если найденный аффикс находится в области R_1 и ему предшествует n, t или гласная, то удалить его.

Шаг 2:

Найти самый длинный аффикс из списка T_2, и если он находится в области R_1, то удалить его.

Шаг 3:

Найти самый длинный аффикс из списка T_3, и если он находится в области R_1, то удалить его.

Шаг 4:

Найти самый длинный аффикс из списка T_4, и если он находится в области R_2, то удалить его.

Шаг 5:

Если на шаге 3 псевдооснова была модифицирована, то найти аффикс из списка T_5 и если он находится в области R_1, то удалить его.

Иначе, если псевдооснова оканчивается на t и ей предшествует гласная, находящаяся в области R_1, то удалить последнюю букву.

Шаг 6:

Все условия далее проверяются последовательно.

Если область R_1 оканчивается сочетанием букв из списка LV, то удалить последнюю букву.

Если область R_1 оканчивается сочетанием букв вида cX, где c — это согласная, а X — одна из букв $a, ä, e, i$, то удалить последнюю букву.

Если область R_1 оканчивается на oj или uj, то удалить последнюю букву.

Если область R_1 оканчивается на jo, то удалить последнюю букву.

Если слово оканчивается на удвоенную согласную, за которой располагается 0 и более гласных, то удалить одну из согласных.

Полученный результат является искомой псевдоосновой анализируемой словоформы.

БИБЛИОГРАФИЧЕСКИЙ СПИСОК

1. Анно Е.Н. Система морфологического анализа с синтезом словоформ // Семиотика и информатика. М., 1978, вып. 10. С. 168-187.
2. Аношкина Ж.Г. Морфологический процессор русского языка // Альманах «Говор». Сыктывкар, 1995. С. 17-23.
3. Ашманов И.С. Архитектура и технология промышленной реализации прикладных лингвистических систем (проверка правописания и электронные словари): автореферат дис. ... кандидата технических наук. Переславль-Залесский, 1995. 22 с.
4. Баранов А.Н. Введение в прикладную лингвистику. М.: Издательство ЛКИ, 2007. 360 с.
5. Белоногов Г.Г. Определение грамматических признаков «новых» слов с помощью словаря // Инженерная лингвистика. Л., Уч. Записки ЛГПИ им. А.И.Герцена. Т. 458. Ч. II., 1971. С. 225-229.
6. Белоногов Г.Г. Об использовании принципа аналогии при автоматической обработке текстовой информации // Проблемы кибернетики. М., 1974, вып. 28.
7. Белоногов Г. Г., Зеленков Ю. Г. Алгоритм морфологического анализа русских слов // Вопросы информационной теории и практики. № 53. Автоматическая словарная служба. Автоматическое индексирование документов. М., 1985. С. 62-93.
8. Белоногов Г. Г., Зеленков Ю. Г. Алгоритм автоматического обнаружения орфографических ошибок в текстах. М.: ВИНИТИ, 1986. 15 с.
9. Бенвенист Э. Классификация языков // Новое в лингвистике. Вып. 3. М., 1963. С. 36-59.
10. Бидер И.Г., Большаков И.А., Еськова Н.А. Формальная модель русской морфологии // ИРЯ АН СССР, Проблемная группа по экспериментальной и прикладной лингвистике, вып. 111-112, М., 1978. I — 60 с., II — 59 с.

11. Болховитянов А.В., Гусев С.В., Чеповский А.М. Морфологические модели компьютерной лингвистики: учеб. пособие / Моск. гос. ун-т печати, — М.: МГУП 2010. — 96 с.
12. Болховитянов А.В., Чеповский А.М. Методы автоматического анализа словоформ // Информационные технологии. М., 2011. № 4 (176). С. 24-29.
13. Валгина Н.С. Активные процессы в современном русском языке. М.: Логос, 2003. 304 с.
14. Варга Д. Проблемы осуществления морфологического анализа при машинном переводе // Научно-техническая информация. М.: ВИНИТИ, 1964. № 4. с. 47-50.
15. Гельбух А.Ф. Эффективно реализуемая модель морфологии флективного естественного языка: автореферат дис. ... кандидата технических наук. АНРФ., ВИНИТИ. М.,1994. 24 с.
16. Герд А.С. Русская морфология и машинный фонд русского языка // Вопросы языкознания. М., 1986. № 6. С. 90-96.
17. Гумбольдт В. фон Избранные труды по языкознанию: Пер. с нем.. – М.: ОАО ИГ «Прогресс», 2001. 400 с.
18. Деза Е.И., Деза М.М. Энциклопедический словарь расстояний. М.: Наука, 2008. 444с.
19. Ермаков А.Е., Плешко В.В., Митюнин В.А. Выделение объектов в тексте на основе формальных описаний. // Информационные технологии. - 2003. – N 12. – С. 1-6.
20. Зализняк А.А. Грамматический словарь русского языка. Словоизменение. М.: Русские словари, 2003. 800 с.
21. Интернет-пространство: речевой портрет пользователя: Коллективная монография / под ред. Т.И. Поповой. Авторы: Т.И. Попова, И.М. Вознесенская, Д.В. Колесова, В.М. Савотина. СПб.: Эйдос, 2012. 224 с.
22. Кнут Д.Е. Искусство программирования на ЭВМ. Т. 3. Сортировка и поиск, 2-е изд. М.: Вильямс, 2007. 832 с.

23. Кобзарева Т.Ю., Лесскис Г.А. Система автоматического морфологического членения текста // Научно-техническая информация. 1979. № 2. С. 23-27.
24. Козьмина Е.Л. Обратные словари. Принципы их создания и использования (на материале обратного словаря французского языка): автореферат дис. ... кандидата филологических наук. М.: МГУ им. М.В. Ломоносова, 1988. 20 с.
25. Кормен Т., Лейзерсон Ч., Ривест Р., Штайн К. Алгоритмы: построение и анализ, 2-е издание. М.: Вильямс, 2005. 1296 с.
26. Кулагина О.С. Исследования по машинному переводу. М.: Наука, 1979. 320 с.
27. Леонтьева Н.Н. Информационная модель системы автоматического перевода // НТИ. Сер. 2. М.,1985. № 10. С. 22-29.
28. Лингвистический энциклопедичекий словарь /Гл. ред. В.Н.Ярцева. – М.: Большая Российская энциклопедия, 2002. 709 с.
29. Мальковский М.Г., Волкова И.А. Анализатор системы TULIPS-2. Морфологический уровень // Вестник Московского Университета, сер. 15. 1981. № 1. С. 70-76.
30. Мальковский М.Г., Старостин А.С., Система морфосинтаксического анализа Treeton и мультиагентный синтаксический анализатор Treevial: принцип работы, система правил и штрафов // Екатеринбург: Изд-во Уральского университета, 2007. — С. 135–143.
31. Марчук Ю.Н. Компьютерная лингвистика. М.: АСТ: Восток-Запад, 2007. 317 с.
32. Мельчук И.А. Курс общей морфологии. Том I. М. – Вена: «Языки русской культуры», Венский славистический альманах, Издательская группа «Прогресс», 1997. 416 с.
33. Мельчук И.А. Язык: от смысла к тексту. М.: Языки славянской культуры, 2012. 176 с.
34. Мечковская Н.Б. История языка и история коммуникации: от клинописи до Интернета: курс лекций по общему языкознанию. М.: Флинта : Наука, 2009. 584 с.

35. Мечковская Н.Б. Общее языкознание. Структурная и социальная типология языков: курс учеб. Пособие для студентов филологических и лингвистических специальностей. М.: Флинта : Наука, 2011. 312 с.
36. Ножов И.М. Прикладной морфологический анализ без словаря. // КИИ-2000. Труды конференции – М.: Физматлит, 2000. Т.1. С. 424-429
37. Осипов Г.С. Лекции по искусственному интеллекту. М.: КРАСАНД, 2009. 272 с.
38. Пиотровский Р.Г., Бектаев К.Б., Пиотровская А.А. Математическая лингвистика. М.: Высшая школа, 1977. 383 с.
39. Пиотровский Р.Г. Инженерная лингвистика и теория языка. Л.: Наука ЛО, 1979. 112 с.
40. Пиотровский Р.Г. Информационные измерения языка. Л.: Наука ЛО, 1968. 116 с.
41. Плунгян В.А. Общая морфология: Введение в проблематику. М.: Эдиториал УРСС, 2003. 384 с.
42. Плунгян В.А. Введение в грамматическую семантику: Грамматические значения и грамматические системы языков мира. М.: РГГУ, 2011. 672 с.
43. Реформатский А.А., Агглютинация и фузия как две тенденции грамматического строения слова // Реформатский А.А. Лингвистика и поэтика. М., 1987. С. 52-76.
44. Реформатский А.А. Введение в языковедение. /5-е издание. – М.: Аспект Пресс, 2008. — 536 с.
45. Русская грамматика: научные труды // Российская академия наук. Институт русского языка им. В.В. Виноградова // Авилова Н.С., Бондарко А.В., Брызгунова Е.А., Дмитренко С.Н., Кручинина И.Н., Лопатин В.В., Ляпон М.В., Плотникова В.А., Суханова М.С., Улуханов И.С., Шведова Н.Ю. // Репринтное издание. М., 2005. 784 с.
46. Рыбаков Ф.И., Руднев Е.А., Петухов В.А. Автоматическое индексирование на естественном языке. М.: Энергия, 1980. 160 с.

47. Сегалович И., Маслов М. Русский морфологический анализ и синтез с генерацией моделей словоизменения для незнакомых слов // Диалог 98: тр. Междунар. сем. по компьютерной лингвистике и ее приложениям. Казань: ООО «Хэтер». 1998. Т. 2. С. 547-552.
48. Сепир Э. Избранные труды по языкознанию и культурологии. Пер. с англ. — М.: Издательская группа «Прогресс», 1993. 656 с.
49. Старостин С.А. Рабочая среда для лингвиста // Гуманитарные науки и новые информационные технологии. М., 1994. Вып. 2. С. 7-22.
50. Тресорукова И.В. Греческий язык. Справочник по грамматике. М.: Живой язык, 2009. 224 с.
51. Успенский Б.А. Структурная типология языков. М.: Наука, 1965. 286 с.
52. Фитиалов С.Я. О построении формальной морфологии в связи с машинным переводом. Тезисы. конф. по обработке информации, машинному переводу и автоматическому чтению текста. ВИНИТИ АН СССР. М., 1961. 21 с.
53. Хойер Г. Антропологическая лингвистика // Зарубежная лингвистика. Т. 2. – М.: Прогресс, 2002, С. 44-67.
54. Хопкрофт Дж. Э., Мотвани Р., Ульман Дж. Д. Введение в теорию автоматов, языков и вычислений, 2-е изд. М.: Вильямс. 2008. 528 с.
55. Черненьков Д.М. Об одном статистическом методе пополнения морфологического словаря. // Компьютерная лингвистика и интеллектуальные технологии: По материалам ежегодной Международной конференции «Диалог». Вып. 9 (16). М.: Изд-во РГГУ, 2010. С. 559-564.
56. Шереметьева С.О., Ниренбург С. Эмпирическое моделирование вычислительной морфологии. // Научно-техническая информация. Сер. 2. Информационные процессы и системы, 1996. № 7. С. 28-33.
57. Штиндлова И. Обратные словари // Автоматизация в лингвистике. Л.: Наука, 1966. С. 84-91.
58. Щипицина Л.Ю. Компьютерно-опосредованная коммуникация: Лингвистический аспект анализа. – М.: КРАСАНД, 2010. 296 с.
59. Языки мира: Уральские языки. М.: Наука, 1993. 398 с.

60. Языки мира: Тюркские языки. М.: Издательство «Индрик», 1997. 544 с.
61. Языки мира: Кавказские языки. М.: Academia, 1998. 480с.
62. Языки мира: Германские языки. Кельтские языки. М.: Academia, 1999. 472 с.
63. Языки мира: Романские языки. М.: Academia, 2001. 720 с.
64. Языки мира: Славянские языки / РАН. Институт языкознания; Ред. колл.: А.М. Молдован, С.С. Скорвид, А.А. Кибрик и др. М.: Academia, 2005. 656 с.
65. Языки мира: Балтийские языки / РАН. Ин-т языкознания; Ред. колл.: В.Н. Топоров, М.В. Завьялов, А.А. Кибрик и др. М.: Academia, 2006. 224 с.
66. Российский семинар по Оценке Методов Информационного Поиска (РОМИП). URL: http://romip.ru/ (дата обращения: 20.01.2013).
67. Русский морфологический словарь. URL: http://www.aot.ru/docs/rusmorph.html (дата обращения: 20.01.2013).
68. Aho A.V., Corasick M.J. Efficient string matching: an aid to bibliographic search // CACM. 1975. Volume 18 Issue 6. P. 333-340.
69. Bacchin M., Ferro N., Melucci M., The Effectiveness of a Graph-Based Algorithm for Stemming // Proceedings of the 5th International Conference on Asian Digital Libraries: Digital Libraries: People, Knowledge, and Technology. December 11-14, 2002. P. 117-128.
70. Bacchin M., Ferro N., Melucci M. A probabilistic model for stemmer generation // Inf. Process. Manage, 41(1), 2005. P. 121-137.
71. Bouras C., Tsogkas V. Improving Text Summarization Using Noun Retrieval Techniques // Proceedings of the 12th international conference on Knowledge-Based Intelligent Information and Engineering Systems, Part II. 2008. P. 593-600.
72. Caumanns J. A Fast and Simple Stemming Algorithm for German Words. Technical report, Center für Digitale Systeme, Freie Universität Berlin, 1999. 10 p.
73. Dalianis H. and Jongejan B. Hand-crafted versus Machine-learned Inflectional Rules: The Euroling-SiteSeeker Stemmer and CST's Lemmatiser // In Proceedings

of the 5th International Conference on Language Resources and Evaluation. 2006. P. 663-666.

74. Dawson J.L. Suffix Removal for Word Conflation // Association for Literary and Linguistic Computing Bulletin. 1974. Volume 2 Issue 3. P. 33-46.
75. Dolamic L., Savoy J. Stemming approaches for east european languages // Lecture Notes in Computer Science. 2008. Volume 5152. P. 37-44.
76. Dolamic L., Savoy J. Indexing and stemming approaches for the Czech language // Journal Information Processing and Management: an International Journal. 2009. Volume 45 Issue 6. P. 714-720.
77. Dolamic L., Savoy J. Indexing and searching strategies for the Russian language // Journal of the American Society for Information Science and Technology. 2009. Volume 60 Issue 12. P. 2540-2547.
78. Eiman Tamah Al-Shammari. Toward An Error-Free Stemming // Proceedings of ADIS European Conference Data Mining 2008. P. 160-163.
79. Eryiğit G., Adalı E. An Affix Stripping Morphological Analyzer for Turkish // Proceedings of the IASTED International Conference ARTIFICIAL INTELLIGENCE AND APPLICATIONS, Innsbruck, Austria, 2004. P. 299-304.
80. Frakes W.B. Term Conflation for Information Retrieval // Proceedings of the 7th annual international ACM SIGIR conference on Research and development in information retrieval (SIGIR'84). 1984. P. 383-389.
81. Frakes W.B., Fox C.J. Strength and Similarity of Affix Removal Stemming Algorithms // ACM SIGIR Forum. 2003. Volume 37 Issue 1. P. 26-30.
82. Figuerola C.G. , Gómez R., Zazo Rodríguez A.F., Luis J., Berrocal A. Stemming in Spanish: A first approach to its impact on information retrieval // Results of the CLEF 2001 Cross-Language System Evaluation Campaign. Working Notes for the CLEF 2001 Workshop. 3 September, Darmstadt, Germany. 2001. P. 197-202.
83. Funchun P., Nawaaz A., Xin Li, Yumao Lu. Context sensitive stemming for web search // Proceedings of the 30th annual international ACM SIGIR conference on Research and development in information retrieval. 2007. P. 639-646.

84. Galambos L. Semi-automatic Stemmer Evaluation // Intelligent Information Processing and Web Mining, Proceedings of the International IIS: IIPWM 04 Conference. Zakopane, Poland, May 17-20, 2004. P. 2009-218.
85. Gaustad T., Bouma G. Accurate Stemming of Dutch for Text Classification // Computational Linguistics in the Netherlands — CLIN. 2001. P. 104-117.
86. Goldsmith J. Unsupervised Learning of the Morphology of Natural language // Journal Computational Linguistics. 2001. Volume 27 Issue 2. P. 153-198.
87. Goldsmith J., Higgs D., Soglasnova S. Automatic language-specific stemming in information retrieval // Cross language information retrieval and evaluation: Proceedings of the CLEF 2000 workshop. C. Peters, Ed.: Springer Verlag. 2001. P. 273-283.
88. Harman D. How Effective is Suffixing? // Journal of the American Society for Information Science. 1991 . Volume 42 Issue 1. P. 7-15.
89. Hull D.A. Stemming algorithms: A Case Study for Detailed Evaluation // Journal of the American Society for Information Science. 1996. Volume 47 Issue 1. P. 70-84.
90. Hull D.A., Grefenstette G. A Detailed Analysis of English Stemming Algorithms. Rank Xerox Research Centre Technical Report. 1995. 27 p.
91. Jurafsky D., Martin J.H. Speech and Language Processing: An Introduction to Natural Language Processing, Computational Linguistics, and Speech Recognition. Second edition. Prentice Hall, New Jersey. 2009. 1024 p.
92. Kraaij W., Pohlmann R. Viewing Stemming as Recall Enhancement // Proceedings of the 19th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval. 1996. P. 40-48.
93. Keselj V., Sipka D. A Suffix Subsumption-based Approach to Building Stemmers and Lemmatizers for Highly Inflectional Languages with Sparse Resources // NFOTHECA, Journal of Informatics and Librarianship. 2008. Vol. IX, no. 1-2. P. 23a-33a, 21-31.
94. Korenius T., Laurikkala J., Järvelin K., Juhola M. Stemming and lemmatization in the clustering of finnish text documents // ACM CIKM. 2004. P. 625-633.

95. Kreslins K. A stemming algorithm for Latvian: a doctoral thesis. Loughborough University, 1996. 204 p.
96. Krovetz R. Viewing morphology as an inference process // Proceedings of the 16th annual international ACM SIGIR conference on Research and development in information retrieval. Pittsburgh, June 27-July 1, 1993. P. 191-202.
97. Larkey L.S., Ballesteros L., Connell M.E. Improving stemming for Arabic information retrieval: light stemming and co-occurrence analysis // Proceedings of the 25th annual international ACM SIGIR conference on Research and development in information retrieval. 2002. P. 275-282.
98. Lovins J.B. Development of a Stemming Algorithm // Mechanical Translation and Computational Linguistics 11. 1968. P. 22-31.
99. Majumber P., Mitra M., Parui S.K., Kole G., Mitra P., Datta K. YASS: Yet another suffix stripper // ACM Transactions on Information Systems. 2007. Volume 25 Issue 4. P. 18-38.
100. Majumber P., Mitra M., Pal D. Bulgarian, Hungarian and Czech Stemming Using YASS // 8th Workshop of the Cross-Language Evaluation Forum (CLEF 2007). Budapest, Hungary, September 19-21, 2007. P. 49-56.
101. Mayfield J., McNamee P. Single N-gram stemming // Proceeding of the 26th annual international ACM SIGIR conference on Research and development in information retrieval. 2003. P. 415-416.
102. Melucci M., Orio N. A novel method for stemmer generation based on hidden Markov models // Proceeding of the 12th international conference on Information and knowledge management. 2003. P. 131-138.
103. Ntais G. Development of a Stemmer for the Greek Language. Master Thesis at Stockholm University / Royal Institute of Technology. 2006. 41 p.
104. Oard D.W., Levow G.A., Cabezas C.I. CLEF experiments at Maryland: Statistical stemming and back off traslation // Workshop of Cross-Language Evaluation Forum on Cross-Language Information Retrieval and Evaluation (CLEF), Springer, London. 2001. P. 176-187.

105. Orasan C., Pekar V., Hasler L. A comparison of summarisation methods based on term specificity estimation // Proceedings of the Fourth International Conference on Language Resources and Evaluation (LREC-04). Lisbon, Portugal. 2004. P. 1037-1041.
106. Paice C.D. Another stemmer // SIGIR Forum. 1990. Volume 24 Issue 3. P. 56-61.
107. Paice C.D. Method for Evaluation of Stemming Algorithms based on Error Counting // Journal of the American Society for Information Science. 1996. Volume 47 Issue 8. P. 632-649.
108. Paik J.H., Mitra M., Parui S.K., Järvelin K. GRAS: An effective and efficient stemming algorithm for information retrieval // ACM Transactions on Information Systems (TOIS). 2011. Volume 29 Issue 4. 19 p.
109. Peters C., Braschler M., Clough P. Multilingual Information Retrieval: From Research To Practice. Springer-Verlag, 2012. 217 p.
110. Popovič M., Willett P. The Effectiveness of Stemming for Natural-Language Access to Slovene Textual Data // Journal of the American Society for Information Science. 1992. Volume 43 Issue 5. P. 384-390.
111. Porter M.F. An algorithm for suffix stripping // Program: electronic library and information systems. 1980. Volume 14 Issue 3. P. 130-137.
112. Porter M.F. Snowball: A language for stemming algorithms. URL: http://www.snowball.tartarus.org/texts/introduction.html (дата обращения: 11.12.12).
113. Rosell M. Improving clustering of Swedish newspaper articles using stemming and compound splitting // 14th Nordic Conference on Computational Linguistics (NoDaLiDa 2003). 2003. P. 30-31.
114. Savoy J. Stemming of French Words Based on Grammatical Categories // Journal of the American Society for Information Science. 1993. Volume 44 Issue 1. P. 1-9.

115. Savoy J. Light Stemming Approaches for the French, Portuguese, German and Hungarian Languages // Proceedings of the 2006 ACM symposium on Applied computing (SAC 2006). 2006. P. 1031-1035.
116. Savoy J. Searching Strategies for the Bulgarian Language // Information Retrieval. 2007. Volume 10 Issue 6. P. 509-529.
117. Schinke R., M. Greengrass, A. Robertson, P. Willett. A stemming algorithm for Latin text databases // Journal of Documentation. 1996. Volume 52. P. 172-187.
118. Tordai A., de Rijke M. Four Stemmers and a Funeral: Stemming in Hungarian at CLEF 2005 // Lecture Notes in Computer Science. 2006. Volume 4022. P. 179-186.
119. Wahbeh A., Al-Kabi M., Al-Radaideh Q.A., Al-Shawakfa E.M., Alsmadi I. The Effect of Stemming on Arabic Text Classification: An Empirical Study // IJIRR. 2011. P. 54-70.
120. Willett P. The Porter stemming algorithm: then and now // Program: Electronic Library and Information Systems. 2006. Volume 40 Issue 3. P. 219-223.
121. Xu J., Croft B.W. Corpus-based stemming using co-occurrence of word variants // ACM Transaction on Information Systems. 1998. Volume 16 Issue 1. P. 61-81.
122. Inxight Federal Systems URL: http://inxightfedsys.com/products/sdks/lx/default.asp (дата обращения: 21.02.2013)
123. Text REtrieval Conference (TREC). URL: http://trec.nist.gov/ (дата обращения: 20.01.2013).
124. Cross Language Evaluation Forum. URL: http://www.clef-campaign.org/ (дата обращения: 11.12.2012).