Учреждение Российской академии наук
Санкт-Петербургский институт информатики и автоматизации РАН

На правах рукописи

Крижановский Андрей Анатольевич

Математическое и программное обеспечение построения списков семантически близких слов на основе рейтинга вики-текстов

Специальность: 05.13.11: «Математическое и программное обеспечение вычислительных машин, комплексов и компьютерных сетей»

Диссертация на соискание учёной степени
кандидата технических наук

Научный руководитель
д.т.н. проф. А.В. Смирнов

Санкт-Петербург

2008

Оглавление

Введение

Некоторые определения. Для более ясного понимания материала и во избежание недоразумений целесообразно привести следующие определения.

Тезаурус – это словарь, в котором слова, относящиеся к каким-либо областям знания, расположены по тематическому принципу и показаны семантические отношения (родо-видовые, синонимические и др.) между лексическими единицами.[1]

Глоссарий – это собрание глосс (толкований) непонятных слов или выражений с толкованием (толковый глоссарий) или переводом на другой язык (переводной глоссарий).[2]

Информационные поисковые системы (ИПС) – системы поиска релевантных[3] документов (текст, изображение, аудио, видео файлы и др.) в сети Интернет или на локальном компьютере, где задача формулируется пользователем в виде набора ключевых слов с возможностью их объединения логическими правилами (И, ИЛИ и др.).

Актуальность темы диссертации. Увеличение числа и изменение качества[4] электронных документов на локальных компьютерах и в сети Интернет требуют новых алгоритмов для более точного и быстрого поиска.

Выделяют два вида сходства [179]: сходство между объектами (рассматривается в данной работе) и сходство между отношениями (relational similarity, см. [178], [177], [180])[5]. Является ли сущность свойством объекта или отношением – определяется контекстом [105]. Поиск похожих объектов

---

1 Определение, функции и примеры тезаурусов см. на стр. 45.

2 «В отличие от понятия тезауруса глоссарий – это понятийно-терминологические определения слов, используемых в тезаурусе конкретной предметной области» [2].

3 «Релевантность (relevance, relevancy) – соответствие документа запросу» [52].

4 Появился новый формат электронных документов – вики, см. стр. 51. Особенности корпуса вики-текстов, позволяющие говорить о качественном изменении вики по сравнению с html страницами, перечислены на стр. 24.

5 Другая задача, получившая название *semantic associations*, заключается в поиске и ранжировании сложных отношений в данных RDF. Две сущности в RDF графе *семантически связаны* (semantic associations), если существует связывающий их путь (или несколько путей). В работе [71] представлен поиск в семантической сети [17] интересных пользователю отношений между двумя сущностями и *ранжирование отношений*, основанное на статистических и семантических (например, учёт типа ссылок) критериях. См. прототип системы: http://lsdis.cs.uga.edu/projects/semdis/index.php?page=3

(similarity search) включает такие (на первый взгляд разные, но общие по способам решения) задачи, как поиск похожих текстовых документов, поиск семантически близких слов, поиск похожих вершин графа. Анализ работ в области вычислительной лингвистики показал большое разнообразие алгоритмов, предлагающих решение этих задач (алгоритм HITS [125], алгоритм PageRank [85] (и его модификация Green [145]), алгоритм распределения рангов ArcRank [174], ESA [103], алгоритм извлечения синонимов из толкового словаря [174], метод извлечения контекстно связанных слов [122], [146] и др.). Поиск похожих документов также может являться подэтапом алгоритма поиска документов по запросу [22].

Объектом исследования является синонимия и семантическая близость слов. Два текста связаны гиперссылкой, если один документ упоминает (то есть ссылается на) другой текст. Тематическая направленность каждого текста определена экспертом, который присваивает одну или несколько категорий тексту[1].

Под семантически близкими словами подразумеваются слова с близким значением, встречающиеся в одном контексте. Более строго и формально семантически близкие слова определяются ниже через понятия авторитетных и хаб-страниц[2]. Представляемая в работе программная система поиска семантически близких слов относится к семантическим[3], поскольку

---

1 Связь, осуществляемая гиперссылкой, не имеет семантики, то есть не описывает смысла этой связи (см. [44], а также http://ru.wikipedia.org/wiki/Семантическая_сеть). Однако категории представляют *однородную* (один тип отношений – родо-видовые) *бинарную* (связаны два объекта) *семантическую сеть*. Иной подход предлагается в работе [54], где семантические элементы считаются семантически связанными, если они связаны отношением «ссылается». Семантическими элементами названы дидактические единицы контента, например «лекция», «определение», «теорема», «термин».

2 См. определения авторитетных и хаб-страниц в главе 1 в подразделе «Алгоритм HITS» на стр. 27, см. также подраздел «Поиск синонимов с помощью HITS алгоритма» на стр. 74. Заметим, что слов хаб встречается в отечественной научной литературе, например, «термин-хаб» в работе [11].

3 «Семантическими принято считать системы, в которых в процессе анализа содержания текста делаются попытки учесть не только лингвосемантические, но и логико-семантические отношения между языковыми объектами. Кроме того, контекст, определяющий лингвосемантические отношения и в обычных системах синтаксического анализа не выходящий за пределы предложения, в семантических системах распространяется на уровни дискурса и текста. Наконец, предполагается, что система семантического анализа должна учитывать как сведения о данной предметной области, так и её связи с внешним миром в целом» [30].

лингвосемантические отношения рассматриваются на уровне текста, а также учитываются сведения о данной предметной области.

Современные алгоритмы поиска синонимов (например, алгоритм извлечения синонимов из толкового словаря [174], алгоритм SimRank [119], алгоритм Similarity Flooding [132]) изначально предназначены для вычисления меры сходства между вершинами графов. Поэтому алгоритмы не учитывают такую дополнительную информацию, как тематическая направленность и метаданные текста [143], [93], [113]. Данная работа призвана восполнить этот пробел.

Требованием к выбору алгоритма (для поиска семантически близких слов) является возможность использования (в рамках алгоритма) тех дополнительных возможностей, которые предоставляет рассматриваемый корпус документов. Это (1) наличие категорий (классифицирующих документы по их тематической принадлежности)[1], (2) наличие метаинформации в виде ключевых слов (в простейшем случае – это заголовок документа). Таким требованиям удовлетворяют алгоритмы HITS [125] и PageRank [85]. Для поиска семантически близких слов был выбран алгоритм HITS, а не PageRank по следующим причинам:

1) формулы вычисления в PageRank требуют экспериментального выбора коэффициента (*damping factor*)[2], а в HITS нет никакого коэффициента за счёт использования двух типов документов (авторитетные и хаб).

2) значения весов (рассчитанные с помощью PageRank) не могут быть использованы напрямую для поиска похожих страниц. То есть нужен дополнительный алгоритм, который будет искать похожие страницы на основе весов PageRank.

Поиск документов на основе алгоритма HITS тесно связан с вычислением сходства вершин в графе. Автором предложены два способа

1 Это позволяет не решать отдельную сложную задачу классификации – соотнесения документа заданным категориям. См., например, работу [137], в которой описано автоматическое отображение веб-страниц в *Yahoo! онтологию* с помощью классификатора Байеса, или статью [106] о поиске похожих документов в библиографическом корпусе на основе алгоритма *поиска ближайшего соседа* (k-NN). Забегая вперёд, укажем, что задача классификации в вики-текстах решена за счёт наличия категорий, указанных авторами текстов.

2 О сложности выбора амортизирующего коэффициента можно судить по работе [73].

вычисления меры сходства вершин графа на основе формализации понятия «похожие вершины» графа. Первый вариант использует понятия авторитетных и хаб-страниц и позволяет формализовать задачу поиска похожих страниц в HITS алгоритме. Во втором варианте получена формула сходства двух вершин *a* и *b*, основанная на поиске общих вершин среди соседей вершин *a* и *b*.

В данной работе представлены алгоритмы (адаптированный HITS алгоритм и оригинальный алгоритм вычисления меры сходства вершин графа) и реализация адаптированного HITS алгоритма в виде программной системы поиска семантически близких слов. Также спроектирована архитектура программной системы оценивания и разработаны способы численной оценки набора синонимов. Способы численной оценки набора синонимов необходимы для проведения экспериментальной части работы.

При выборе программных инструментальных средств разработки и проектирования архитектуры программы автор придерживался следующих требований: открытость исходного кода (open source), кроссплатформенность (возможность работы на разных платформах: Linux, Windows и др.), модульность архитектуры (возможность использовать предыдущие наработки и интегрировать решения разных подзадач). Важными требованиями были: использование достаточно широко распространённых и хорошо себя зарекомендовавших программных систем для обработки текста на естественном языке и представление результатов работы в виде текста и графики (визуализация). Использование общепринятого стандарта и модульность архитектуры позволяют решить задачу большой сложности (например, машинный перевод), разбив её на ряд подзадач. В качестве программной среды для обработки текстов на естественном языке была выбрана модульная система GATE [92], [98].

Сложность организации поиска семантически близких слов и, в частности, синонимов определяется рядом причин. Во-первых, автору не известно общепринятой количественной меры для определения степени синонимичности значений слов. Можно утверждать, что одна пара слов более синонимична чем другая, но не ясен способ, позволяющий однозначно

указывать – во сколько раз.[1] Во-вторых, понятие синонимии определено не для слов, а для значений слов, то есть синонимия неразрывно связана с контекстом. В-третьих, язык – это вечноизменяемая субстанция, открытая система. Слова могут устаревать или получать новые значения. Особенно активное словообразование и присвоение новых значений словам наблюдается в науке, в её молодых, активно развивающихся направлениях. Решение задачи поиска синонимов в частности (а также современных задач автоматизированной обработки текстов на естественном языке в целом) требуют предварительной морфологической обработки текста.

Отсутствуют (по крайней мере, неизвестны автору) доступные модули в системе GATE для морфологической обработки русского языка. Возможно, поэтому система GATE редко упоминается в системах обработки текстов на русском языке. Таким образом, существует насущная необходимость в наличии модуля морфологической обработки русского языка в системе GATE, позволяющая нормализовать слова (лемматизация[2]), получать морфологические признаки слова (например, род, падеж) и т. д. При этом существует общедоступная программа морфологической обработки русского Lemmatizer (разработанная в проекте Диалинг московскими учёными). Сложность в том, что GATE написан на языке программирования Java, а Lemmatizer написан на C++. Таким образом, решением данной задачи будет разработка архитектуры позволяющей интегрировать эти системы.

К задачам автоматической обработки текста (АОТ) относятся такие задачи, как: машинный перевод, поиск и хранение текста [52], кластеризация

---

1 «В начале 50-х годов XX века группа американских исследователей под руководством Ч. Осгуда опубликовала сенсационную книгу «Измерение значения». Для языковедов само сочетание этих слов было бессмыслицей: каждому ясно, что *значение слова, его смысл невозможно как-то там измерить* (курсив наш – А.К.). Но Ч. Осгуд действительно открыл для языкознания нечто новое. Он доказал, что в области семантики возможны измерения <...> Ч. Осгуд впервые выделил и измерил качественно-признаковый аспект значения слова» [48].

2 Лемматизация – приведение слова к неопределённой (словарной) форме, например, для глаголов – это получение инфинитива (*бежал – бежать*), для существительных – это 1-ое лицо, ед.ч. (*яблоки – яблоко*). В работе [67] используется термин *лексикографический контроль*. «Он заключается в приведении используемых ключевых слов к единой морфологической форме и к единому написанию, в учёте синонимии и многозначности ключевых слов» ([67], стр. 75).

текстов[1] [43], [70], определение тематически однородных частей текста и приписывание этим частям документа тематических тегов [72], [104], реферирование текстов, и многие др. Автоматический поиск синонимов и семантически близких слов является одной из задач АОТ.

Актуальность работы определяется возможными областями приложений результатов диссертации. Во-первых, это поиск похожих вершин графа в рамках задачи Ontology Matching [132], [164], [190]. Во-вторых, предложенное решение задачи автоматического поиска синонимов и семантически близких слов может использоваться в поисковых системах для расширения запроса (на основе вычисления сходства запроса и документа [86], сходства запросов между собой[2] [101], с помощью тезаурусов [10], [95], [163]), для автоматизированного построения онтологии по тексту[3], для расширения существующих и создания новых тезаурусов[4] [135]. В-третьих, разработанная программа поиска семантически близких слов, вероятно, будет востребована лингвистами-лексикографами при составлении словарей синонимов [7], [56], [161]. В работе [79] перечислены ещё два приложения, требующих решения задачи «*similarity search*»:

- «collaborative filtering» – определение пользователей, имеющих одинаковый вкус, предпочтения;
- поиск / исключение документов почти-копий (англ. «*near duplicate*»), которое требуется при индексировании документов.

---

1 «Кластер-анализ – это способ группировки многомерных объектов, основанный на представлении результатов отдельных наблюдений точками подходящего геометрического пространства с последующим выделением групп как “сгустков” этих точек.» [43]. Кластер в англ. это «сгусток», «гроздь (винограда)», «скопление (звёзд)» и т.п. Неформально, кластер – это связный подграф с большим числом внутренних и небольшим числом внешних рёбер [165].

2 В работе [79] указан вариант объединения двух задач: (1) уточнение поискового запроса и (2) определение сходства запросов между собой. Подход заключается в том, чтобы на основе сходства результатов (множеств найденных документов) находить похожие запросы. Тогда поисковая система сможет предложить пользователю альтернативные формулировки запроса.

3 В работе [116] представлена схема извлечения концептов и отношений из текста с помощью эксперта (система T-Rex – The Trainable Relation Extraction framework).

4 Достоинство тезаурусов, построенных с помощью Википедии, как отмечают в работе [135] – это стоимость, постоянное расширение, то есть адекватность современному лексикону, многоязычность (то есть привязка к концепту слов на разных языках).

Ещё одна актуальная область связана с задачей определения значения многозначного слова[1]. Основа алгоритма представленного в [153], [187] – анализ контекста слова. При этом начальные слова[2] в обучающем наборе (в алгоритме, предложенном в [187]) должны точно различать возможные значения. Выбор начальных слов (для заданного слова) можно выполнять с помощью предложенного в диссертации алгоритма поиска семантически близких слов. Другие актуальные направления новых информационных технологий, в которых могут использоваться результаты данной диссертационной работы – это направление запросно-ответных систем (question-answering system) и автоматическое создание проблемно-ориентированных тезаурусов[3].

Данная диссертационная работа выполнена в рамках указанного направления исследований.

Цель работы и задачи исследования. Целью работы является решение задачи автоматизированного построения упорядоченного списка семантически близких слов в проблемно-ориентированных корпусах с гиперссылками и категориями (на примере корпуса текстов открытой энциклопедии Википедия) с возможностью оценки результатов поиска. Для достижения поставленной цели необходимо:

1. Проанализировать методы поиска семантически близких слов, обосновать выбор текстовых ресурсов, алгоритма (с возможной адаптацией) и программных систем для автоматической обработки текстов на естественном языке (ЕЯ).
2. Разработать подход к поиску семантически близких слов (в корпусе текстовых документов с гиперссылками и категориями).
3. Разработать алгоритмы поиска семантически близких слов в корпусе текстовых документов с гиперссылками и категориями.

---

1 Задача определения значения многозначных слов (Word sense disambiguation или WSD) состоит в приписывании каждому экземпляру слова одного из известных (например из словаря) значений. Эта задача отличается от задачи вывода значения слова (sense induction).

2 Начальные слова (seed words) представляют контекст, то есть входят в словосочетания, содержащие исследуемое многозначное слово. Начальные слова подбираются так, чтобы словосочетание имело однозначный смысл.

3 В работе [123] предложен метод построения таксономии по набору документов (система TaxaMiner).

4. Спроектировать и реализовать программный комплекс поиска семантически близких слов; разработать способы численной оценки наборов синонимов.

Методы исследования. Для решения поставленных задач в работе используются методы кластерного анализа [43], [70], методы теории графов [19], [28], [29], [38], [45], [46], [49], элементы теории сложности алгоритмов [5], [23], [32], [42], стандарты открытых информационных сред. При разработке программного обеспечения использовалась технология объектно-ориентированного программирования (Java, C++) [13], язык структурированных запросов (SQL) управления данными в реляционных базах данных [26], программная среда для обработки текстов на естественном языке (GATE) [92], [98].

Научная новизна

1. Новизна предложенного подхода к поиску семантически близких слов в проблемно-ориентированном корпусе заключается в том, что кроме гиперссылок дополнительно учитывается метаинформация документов (ключевые слова, категории).
2. Новизна адаптированного HITS алгоритма состоит в том, что при поиске наиболее похожих документов в корпусе учитываются не только гиперссылки, но и категории, что позволяет применить механизм иерархической кластеризации, объединяющий семантически близкие слова в смысловые группы.
3. Новый способ построения корневого набора документов в адаптированном HITS алгоритме заключается в выборе документов, связанных гиперссылками с исходным документом (заданным пользователем), что позволяет отказаться от шага «предварительный веб-поиск документов».
4. Коэффициент Спирмена модифицирован для численного сравнения списков семантически близких слов; отличие заключается в возможности сравнивать списки разной длины.

5. Впервые предложен показатель степени синонимичности набора слов, заключающийся в сравнении этого набора с эталонным списком синонимов (например из тезауруса).
6. Впервые спроектирована распределённая архитектура программного комплекса, позволяющего выполнять поиск семантически близких слов и оценивать результаты поиска на основе удалённого доступа к тезаурусам.
7. Эксперименты подтвердили выполнение закона Ципфа для текстов Русской Википедии и Википедии на английском упрощённом языке на основе построенных индексных баз данных.

Обоснованность и достоверность научных положений, основных выводов и результатов диссертации обеспечивается за счёт тщательного анализа состояния результатов исследований в области вычислительной лингвистики, подтверждается корректностью предложенных моделей, алгоритмов и согласованностью результатов, полученных при компьютерной реализации, а также проведением экспериментов с поиском семантически близких слов в корпусе текстов английской и русской версии энциклопедии Википедия.

Практическая ценность работы заключается в том, что реализованная программная система позволяет выполнять поиск семантически близких слов в английской и русской версии энциклопедии Википедия. Причём нет принципиальных ограничений в применении программы к Википедия на других языках, к вики ресурсам вообще и корпусам текстов, удовлетворяющих указанным выше требованиям[1].

Наличие категоризации статей и большое количество самих статей в тестируемом источнике данных (Википедия) позволяют получить набор проблемно ориентированных текстов практически на любую тематику[2]. Таким образом, можно выполнять поиск семантически близких слов как по

1 Это обусловлено тем, что адаптированный HITS алгоритм оперирует категориями, ссылками между документами, ключевыми словами (в реализации – это заголовок документа). При этом заголовок документа рассматривается как неделимая сущность и не важно на каком языке он написан.

2 Число статей Английской Википедии превысило размер энциклопедии Британника.

всей энциклопедии, так и по некоторому подмножеству текстов определённой тематики[1].

Разработана архитектура программного модуля RuPOSTagger системы GATE для удалённого доступа к программе морфологического анализа русского языка (использован модуль морфологического анализа русского языка проекта Диалинг). Модуль RuPOSTagger может использоваться как внутри GATE (с другими модулями), так и быть интегрирован в отдельный (standalone) программный продукт. Спроектирована архитектура и реализована система индексирования вики-текстов.

Реализация результатов работы.

Исследования, отражённые в диссертации, были поддержаны грантами РФФИ (проект № 02-01-00284 «Методологические и математические основы построения компьютерных систем быстрой интеграции знаний из распределённых источников» 2002-2004 гг.; № 06-07-89242 "Методология и модели интеллектуального управления конфигурациями распределенных информационных систем с динамически изменяющимися структурами", 2006-2008 гг.; № 05-01-00151 "Методологические и математические основы построения контекстно-управляемых систем интеллектуальной поддержки принятия решений в открытой информационной среде", 2005-2007 гг.), грантами Президиума РАН (проект № 2.44 «Многоагентный подход к построению компьютерной среды для быстрой интеграции знаний из распределённых источников» 2001-2003 гг. и проект № 2.35 «Контекстно-управляемая методология построения распределённых систем интеллектуальной поддержки принятия решений в открытой информационной среде» 2003-2008 гг.), а также грантом ОИТВС РАН (проект № 1.9 «Разработка теоретических основ и многоагентной технологии управления контекстом в распределённой информационной среде» 2003-2005 гг.).

Разработан программный комплекс Synarcher на языке Java для поиска семантически близких слов в энциклопедии Википедия с динамической

1 Более подробно о фильтрации статей при поиске и чёрном списке категорий см. на стр. 86.

визуализацией результатов поиска[1]. Результаты поиска представлены в виде текста (список семантически близких слов), в виде таблицы (с возможностью упорядочения и редактирования) и в виде графического представления набора вершин и рёбер с возможностью показать/спрятать соседние вершины для текущей вершины. Настройка параметров поиска позволяет (i) указать размер пространства поиска, что определяет время поиска и результат, (ii) разрешить поиск статей определённой тематики (то есть сузить область поиска) за счёт выбора категорий статей.

Спроектирована и реализована распределённая клиент-серверная архитектура в программном комплексе *Russian POS Tagger*[2], позволяющая интегрировать среду GATE и модуль морфологической обработки русского языка Lemmatizer (фирма Диалинг). Комплекс RuPOSTagger предоставляет веб-сервис на основе XML-RPC протокола. Веб-сервис обеспечивает вызов функций модуля Lemmatizer из системы GATE или из отдельного Java приложения.

Часть результатов была использована при выполнении контракта «Интеллектуальный доступ к каталогам и документам» на создание системы поддержки клиентов, реализованной для немецкой промышленной компании Фесто, 2003–2004 гг. Разработан и реализован алгоритм кластеризация запросов (на естественном языке) и пользователей на основе использования онтологий в данном проекте [172].

Разработана архитектура программной системы поиска семантически близких слов в исследовательском проекте CRDF № RUM2-1554-ST-05 «Онтолого-управляемая интеграция информации из разнородных источников для принятия решений», 2005-2006 гг.

Апробация результатов работы. Основные положения и результаты диссертационной работы представлялись на международном семинаре «Автономные интеллектуальные системы: агенты и извлечение данных» (Санкт-Петербург 2005), международной конференции «Диалог» (Бекасово 2006), 11ой международной конференции «Речь и Компьютер» (Санкт-

1 Программа с открытым исходным кодом, доступна по адресу http://synarcher.sourceforge.net

2 Программа с открытым исходным кодом, доступна по адресу http://rupostagger.sourceforge.net

Петербург 2006), международной конференции «Корпусная лингвистика» (Санкт-Петербург 2006) и первой конференции в России «Вики-конференции 2007» (Санкт-Петербург 2007). Часть результатов работы представлена в публикациях [33], [36], [35], [57], [58], [168], [169], [170], [171], [172].

Публикации. Основные результаты по материалам диссертационной работы опубликованы в 8 печатных работах, в том числе в 2 журналах из списка ВАК («Труды Института системного анализа РАН», 2004, «Автоматизация в промышленности», 2008).

Структура и объём работы. Диссертационная работа состоит из введения, четырёх глав, заключения, списка литературы и пяти приложений. Работа изложена на 156 страницах и включает 35 рисунков, 14 таблиц, а также список литературы из 190 наименований; приложения на 14 страницах. Общий объём работы составляет 188 страниц.

Основные результаты. Предлагаемые в диссертации алгоритмы позволяют реализовать поиск синонимов и слов близких по значению в наборе текстов специальной структуры.[1] В ходе исследований, представленных в диссертации, были получены следующие результаты:

1. HITS алгоритм адаптирован к поиску наиболее похожих документов (в корпусе текстов с гиперссылками и категориями) на основе алгоритма иерархической кластеризации;
2. Разработано прикладное программное обеспечение для поиска семантически близких слов в проблемно ориентированном корпусе текстов с динамической визуализацией результатов поиска;
3. Предложена (1) архитектура распределённой программной системы оценивания результатов поиска на основе тезаурусов (WordNet, Moby) и (2) сами методы численной оценки (адаптация метода Спирмена для сравнения ранжирования в списках разной длины);

1 См. «Требования к корпусу проблемно-ориентированных текстов» на стр. 24.

4. Разработана и реализована архитектура подсистемы GATE для удалённого доступа к программе морфологического анализа русского языка (на основе XML-PRC протокола).

Таким образом, в результате исследований, проведённых автором, получено решение актуальной проблемы автоматизированного построения списков семантически близких слов.

В первой главе приводится анализ основных проблем автоматической обработки текста и поиска семантически близких слов. В качестве текстового ресурса выбрана энциклопедия Википедия, рассмотрен ряд алгоритмов и выбран алгоритм HITS[1], определён список задач (необходима адаптация алгоритма HITS к корпусу текстов с категориями, необходимо разработать способ оценки работы алгоритма, необходим способ визуализации результатов поиска). Была выбрана система обработки текста на естественном языке – модульная система GATE, позволяющая унифицировать программные компоненты. Указан недостаток системы, который необходимо исправить – это отсутствие доступного модуля в системе GATE для морфологической обработки русского языка.

Во второй главе представлена адаптация алгоритма HITS (с использованием алгоритма кластеризации) к поиску похожих документов[2] в корпусе с ссылками и категориями. Приведена оценка временной сложности адаптированного HITS алгоритма. Также предложен алгоритм алгоритм вычисления меры сходства вершин графа. Выполнена оценка временной сложности данного алгоритма и предложены две эвристики, позволяющие уменьшить временную сложность алгоритма. В конце главы предложены методы численной оценки наборов синонимов, полученных на выходе адаптированного HITS алгоритма (это адаптация метода *Spearman's footrule* и оценка на основе тезаурусов *WordNet* и *Moby*).

Третья главе посвящена архитектуре и моделям программ, реализующих

1 Список требований, на основе которых был выбран алгоритм HITS, приведён выше.

2 Понятие похожесть документов основано на концепциях авторитетных и хаб-страниц [125]. В общих чертах, два документа будут считаться похожими, если существует достаточное число документов, которые ссылаются на эти два документа в одном контексте. Более подробное определение авторитетных и хаб-страниц см. в гл.1 в подразделе «Алгоритм HITS», стр. 27. Было формализовано понятие «похожие вершины» графа, см. стр. 76, формулы (2.3)-(2.6).

разработанные алгоритмы. В главе описана архитектура программы Synarcher, реализующей адаптированный HITS алгоритм, детально описан модуль визуализации программы: интерфейс и функциональность. Описана архитектура программного модуля системы GATE для удалённого доступа к программе морфологического анализа русского языка *Lemmatizer* (предлагается использовать разработанные автором XML-RPC клиент и сервер). В главе представлена архитектура программной системы, позволяющей оценить построенные списки семантически близких слов. Оценка основана на данных тезаурусов (например WordNet, Moby). Разработана архитектура системы индексирования вики-текстов, включающая программные модули GATE и Lemmatizer. Реализован программный комплекс индексации текстов Википедии на трёх языках: русский, английский, немецкий.

В четвёртой главе описаны эксперименты поиска синонимов в Английской и Русской Википедии с помощью адаптированного HITS алгоритма. Представлен пример работы разработанного автором программного модуля *Russian POS Tagger* в составе системы GATE. Описаны эксперименты по построению индексных баз данных Русской Википедии и Википедии на английском упрощённом языке.

Положения, выносимые на защиту.

1. Подход к поиску семантически близких слов на основе метаинформации в проблемно-ориентированном корпусе, содержащем два типа текстовых документов (статья и категория) и два типа отношений: иерархические отношения (родо-видовые и часть – целое) и гиперссылки.
2. Адаптированный HITS алгоритм поиска семантически близких слов в корпусе текстовых документов с гиперссылками и категориями. Модификация алгоритма включает: (1) новый способ построения корневого набора (релевантных документов), позволяющий отказаться от предварительного поиска документов, а также (2) использование механизма иерархической кластеризации для объединения слов в смысловые группы.
3. Клиент-серверная архитектура программного комплекса, предназначенного для решения задачи поиска семантически близких слов с возможностью оценки (с помощью удалённого доступа к тезаурусам и на основе модификации коэффициента Спирмена) семантической близости построенных списков слов.
4. Программный комплекс поиска семантически близких слов в проблемно-ориентированном корпусе текстов с динамической визуализацией результатов поиска.
5. Архитектура системы индексирования вики-текстов и её программная реализация.

1. Анализ проблемы автоматической обработки текста и поиска семантически близких слов

Для автоматической обработки текста (АОТ) требуются такие ресурсы, как тексты (корпуса текстов), алгоритмы и их реализация в виде программных систем. Данные ресурсы будут рассмотрены в этой главе с точки зрения возможности решения с их помощью поставленных задач.

Проблема синонимии

Данная работа тесно связана с понятиями значение, смысл, семантическая близость слов. По Выготскому Л.С. [16] следует различать «смысл» слова и его «значение». «Смысл, не являясь в отличие от значения, неразрывно связанным с определённой знаковой формой, отличим от знака. Всегда существует *возможность выражения одного и того же смысла через различные наборы знаков* (курсив наш. – А.К.). Иначе говоря, смысл никогда не связан какой-либо жёсткой знаковой формой. Количество степеней семантической свободы знака обусловлено, в свою очередь, его положением в контексте. <...> Смысл слова неисчерпаем.[1] <...> Смысл никогда не является полным.» [16] (цит. по [48]). Таким образом, многовариантность знакового выражения одной и той же вещи (явления) определяет явление синонимии.

«В языке нет полных синонимов. Нет точных соответствий между схожими по значению словами в разных языках» [2]. Это определяется явлениями полисемии (многозначность) [50], омонимии (два или более слова с совершенно разными исконными значениями, одинаковые по форме), синонимии, энантиосемии (одна и та же форма слова может вмещать прямо противоположные значения – *просмотрели* означает «видели, увидели» и «не увидели» [39].

«Синонимом в полном смысле следует считать такое слово, которое определилось по отношению к своему эквиваленту (к другому слову с тождественным или предельно близким значением) и может быть

1 Обратим внимание на более жёсткую позицию в работе [51], где утверждается, что «... только слова имеют значения. Текст же имеет смысл, а не значение.»

противопоставлено ему по какой-либо линии: по тонкому оттенку в значении, по выражаемой экспрессии, по эмоциональной окраске, по стилистической принадлежности, по сочетаемости...» [56]. В [56] выделяют две функции синонимов — «уточнительная» (акцентирование того или иного оттенка понятия) и стилистическая.

Однако в работе [18] не делается различия между этими двумя функциями, а утверждается, что важнейшая стилистическая функция синонимов – это наиболее точное выражение мысли. Учитывая смысловые и стилистические отличия синонимов, их разделяют на несколько групп[1]:

1. Синонимы, различающиеся оттенками в значениях, называются *семантическими* (от гр. semantikos – обозначающий), другое название – «идеографические» (гр. idea – понятие, grapho – пишу), или понятийные (молодость – юность, красный – багровый – алый) [18]. Под *фразеологическими синонимами* понимают “фразеологизмы[2] с близким значением, обозначающие одно и то же понятие, как правило, соотносительные с одной и той же частью речи, обладающие частично совпадающей или (реже) одинаковой лексико-фразеологической сочетаемостью, но отличающиеся друг от друга оттенками значения, стилистической окраской, а иногда и тем и другим одновременно” [27].
2. Синонимы, которые имеют одинаковое значение, но отличаются стилистической окраской, которая не позволяет заменять их в одном контексте, например, *глаза – очи – зенки*, называются *стилистическими* [18], [1]. Фразеологические синонимы отличаются большей стилистической однородностью, чем

1 Не менее интересная типизация присуща и антонимам, в работе [65] (стр. 9) (вслед за Л.А. Новиковым) выделяют классы: контрарные, комплементарные и векторные антонимы (семантическая классификация антонимии). Там же определены точные / неточные, производные и отражённые антонимы.

2 «Фразеологизм – это воспроизводимый в речи оборот, построенный по образцу сочинительных и подчинительных словосочетаний, обладающий целостным (реже – частично целостным) значением и сочетающийся со словами свободного употребления.» [27]. Примеры фразеологических синонимов: *как свои пять пальцев* (разг.); *вдоль и поперёк* (разг.); *до последней запятой* (разг.), *до <последней> точки* (разг.), *до тонкости* (разг.), *до точности* (прост.).

лексические синонимы (состоящие из слов свободного употребления), так как фразеологизмам в основном присуща эмоциональна экспрессивная окрашенность [27].

3. Синонимы, которые отличаются и по значению, и своей стилистической окраской, называются *семантико-стилистическими* [18].
4. Синонимы, представляющие для нейтрального слова его экспрессивные, эмоционально окрашенные дериваты, называются *деривационными*, например, *старик – старикан*, *старик – старичок* [1].

Стоит отметить, что деление синонимов на стилистические и «идеографические» достаточно условное. Поскольку «... материал показывает, что невозможно провести границу между теми и другими, зачислив одни в стилистические, а другие только в идеографические. Основная, подавляющая масса синонимов служит и стилистическим и смысловым (оттеночным, уточнительным) целям, часто выполняя и ту и другую функции одновременно» [56].

Способы определения синонимии в современных системах

В проектах WordNet и EuroWordNet синонимия определяется через понятие взаимозаменяемости. «Два слова (выражения) считаются синонимами, если существует хотя бы один контекст С, в котором замена одного слова другим не приводит к изменению истинностного значения» [99] (цит. по [1]).

Поскольку, во-первых, взаимозаменяемость в контексте не всегда связана с общностью значений, во-вторых, некоторые синонимы не являются взаимозаменяемыми в контексте из-за особенностей синтаксической или же лексической сочетаемости, постольку авторы тезауруса RussNet используют *критерий взаимозаменяемости* только как дополнительный критерий. Основными критериями семантической близости в RussNet являются идентичность словарных определений или взаимная отсылка в синонимических определениях, что проверяется при дефиниционном анализе [1]. Таким образом, в RussNet отношение синонимии устанавливается между

лексико-семантическими вариантами слов, которые (i) принадлежат одной части речи, (ii) имеют сходные значения, (iii) могут быть взаимозаменяемы в контексте.

Следующие типы синонимов определены в тезаурусе РуТез [41]:

1. Лексические синонимы (полные синонимы; синонимы, отражающие различные языковые стили; синтаксические синонимы; словообразовательные синонимы);

2. Условные синонимы (сокращения; сложные и сложносокращённые слова; некоторые антонимы[1]; некоторые родовидовые синонимы[2]; существительные, обозначающие лиц мужского и женского пола[3]);

3. Другие типы (дериваты; образные наименования; фрагменты толкования; энциклопедические синонимы[4]; исторические синонимы; словосочетания с исключением внутреннего члена; словосочетания с различными реализациями одного из актантов главного слова термина[5]; термины, тесно связанные отношениям *причина-следствие* и др.; термины, несущие в себе дополнительную модальность по отношению к основному термину[6]; термины, совпадающие в одной своей части, а в другой – состоящие из ситуационно связанных терминов[7]; термины, в которых словосочетание с неоднозначным термином становится однозначным).

---

1 Пример антонимов: *доверие правительству – вотум недоверия правительству, правовое обеспечение – правовой вакуум*.

2 Пример родовидовых синонимов: *здравоохранение – укрепление здоровья, каракулево-смушковое сырьё – каракуль – каракульча – смушка*.

3 Например, *спортсмен – спортсменка, владелец – владелица*.

4 Энциклопедические синонимы – такие языковые выражения, тождественность которых вытекает из энциклопедических знаний. Например, *альтернативная гражданская служба – альтернативная военная служба – альтернативная служба, внутренние войска – войска МВД*.

5 Например, *встреча на высшем уровне – встреча в верхах*.

6 Пример дополнительной модальности: *артиллерийский обстрел – артиллерийская канонада – артиллерийская подготовка – артиллерийский удар*.

7 Например, *безопасность судоходства – безопасность кораблей – безопасность на море*.

Проблема текстовых ресурсов.

К задачам лингвистики относят, с одной стороны, идентификацию структурных единиц (например, морфемы, слова, фразы) и описание того, как одни структурные единицы формируют другие, более крупные (например, по каким правилам можно строить из слов фразы). С другой стороны, благодаря наличию текстов и аудио записей, изучают речь в том виде, как мы её слышим. В этом случае необходимо наличие корпуса – набора текстов с грамматической, синтаксической разметкой или без таковых. Среди множества проблем создания корпуса, можно выделить общую проблему отсутствия единого стандарта и сложности практического характера: опечатки, сохранение переносов в тексте [55]. Данная работа непосредственно связана с корпусной лингвистикой. Проблемы корпусной лингвистики раскрываются в работах [82], [133], [174]. В диссертации в качестве корпуса текстов предлагается использовать коллективную онлайн энциклопедию Википедия. Это позволяет решить в какой-то мере проблему стандарта (все статьи унифицированы, а именно: есть стандартные метаданные – заголовок статьи, категории, определяющие тематику статьи), но появляются новые сложности (например, проблема неравномерности количества и качества статей в зависимости от тематики)[1].

Одной из первых работ в области компьютерного семантического анализа можно считать построение «Русского семантического словаря» компьютером в 1982 (группа под руководством чл.-корр. АН СССР Ю. Караулова). Программа сравнивала описание значений слов в разных словарях. При наличии сходства в описании, программа относила слово к одной группе, то есть считала слова сходными по значению. Таким образом, программа является является автоматическим понятийным классификатором слов [48].

Требования к корпусу проблемно-ориентированных текстов

В данной работе рассматриваются корпуса проблемно-ориентированных текстов с гиперссылками и категориями. Эти тексты должны отвечать следующим условиям.

1 Об этой и других проблемах Википедии см. в подразделе «Корпус текстов вики-ресурса Википедия».

1. Каждому текстовому документу (далее «статья») соответствует одно или несколько ключевых слов, отражающих содержание статьи. Например, в случае энциклопедии – энциклопедической статье соответствует одно слово – название статьи.
2. Статьи связаны ссылками. Для каждой статьи определены: набор исходящих ссылок (на статьи, которые упоминаются в данной статье) и входящих ссылок (на статьи, которые сами ссылаются на данную статью).
3. Каждая статья соотнесена одной или нескольким категориям (тематика статьи). Категории образуют дерево таким образом, что для каждой категории есть родитель-категория (кроме корня) и один или несколько детей-категорий (кроме листьев).

Данная структура является не абстрактным измышлением. Она имеет конкретное воплощение в структурах типа вики (wiki), получивших широкое распространение в последнее время в сети Интернет, например, в виде электронной онлайн энциклопедии Википедии[1]; “вики используется российскими органами власти[2] и Департаментом образования Москвы[3] при создании административных интернет-сайтов.”[4]

Наличие единообразных метаданных (заголовок документа, категории), принадлежащих документам корпуса, позволяет отнести поисковую систему, выполняющую поиск на основе этих данных, к классу *гипертекстовых информационно-поисковых систем*[5]. Разработке такой системе посвящена данная работа.

---

1 См. http://wikipedia.org

2 ФЦП «Электронная Россия». Информационное сопровождение Программы. Организация коммуникации по вопросам административного обеспечения государства и использования ИКТ в практике администрирования. http://projects.economy.gov.ru/pms/DownloadFile.aspx/tt_eg2006v3_nov05.doc?workproductid=70a5c8fa-5e73-463f-9233-15db250b80ba.

3 Департамент образования г. Москвы: Методические рекомендации для школ, подключаемых к сети Интернет. http://web.archive.org/web/20070828105815/http://www.educom.ru/ru/projects/link-up/package/.

4 См. http://ru.wikipedia.org/wiki/Википедия:Пресс-релиз/В_десятке!.

5 В соответствии с классификацией, предложенной в работе [15]. «Гипертекстовые ИПС – характеризуются наличием не только содержания, но и некоторой унифицированной структуры сведений о документах. Такие сведения являются метаданными относительно исходных документов» [15].

1.1 Основные алгоритмы поиска похожих интернет страниц, поиска слов близких по значению, вычисления меры сходства вершин графа

Алгоритмы, выполняющие поиск похожих документов и близких по значению слов, можно условно[1] разделить на группы:[2]

1. поиск на основе *анализа ссылок* (вычисления на *графах*)
   i. ссылки заданы явно гиперссылками (HITS [125], PageRank [85], [102], ArcRank [174], Green [145], WLVM [134]);
   ii. ссылки нужно построить[3] (Similarity Flooding [132], алгоритм извлечения синонимов из толкового словаря [84], [83], [174]);
2. поиск на основе анализа текста:
   iii. статистические алгоритмы (ESA [103], сходство коротких текстов [159], извлечение контекстно связанных слов на основе частотности словосочетаний [146]);
   iv. автоматическое понимание текстов[4];
3. поиск на основе анализа и ссылок и текста [81], [129][5].

Для уточнения результатов поиска могут использоваться данные о семантически близких словах из тезаурусов Роже, WordNet, Moby, Викисловаря и др.

Входными данными могут быть [106]:

i. *запрос*, состоящий из ключевых слов, тогда будет выполняться поиск документов, похожих на запрос;

---

1 Практическая реализация может объединять возможности разных подходов.

2 См. также обзор и классификацию методов и приложений вычисления сходства коротких текстов в [147].

3 Для определения силы связи между словами по совместной встречаемости в документах либо в общем контексте — могут использоваться специальные алгоритмы [40].

4 На сегодняшний момент, автору не встретились работы, посвящённые поиску семантически близких слов с помощью систем автоматического понимания текстов (АПТ). О системах АПТ см. в [41].

5 В работе [129] предложена мера вычисления семантического сходства интернет страниц на основе учёта и ссылок, и текста. Сходство текста вычисляется с помощью TF (формула косинусного коэффициента). Сходство ссылок вычисляется с помощью формулы «частота ссылок – обратная частота документов» (то есть в формуле TF-IDF документы оставили, а слова заменили на ссылки).

ii. *идентификатор документа*, будут искаться документы, похожие на заданный.[6]

Алгоритмы анализа гиперссылок: HITS, PageRank, ArcRank, WLVM

Алгоритм HITS[2]

Алгоритм HITS (Hyperlink-Induced Topic Selection)[3] позволяет находить Интернет страницы, соответствующие запросу пользователя, на основе информации, заложенной в гиперссылки [125]. Демократическая природа Интернет позволяет использовать структуру ссылок как указатель значимости страниц (эта идея есть и в алгоритме PageRank [85], встроенном в поисковик Google). Страница p, ссылаясь на страницу q, полагает q авторитетной, стоящей ссылки. Для поиска существенно, что страница q соответствует тематике страницы p.

Поиск в Интернет (Web search) – это нахождение релевантных страниц, соответствующих запросу. Можно выделить два крайних типа запросов: конкретный (проблема недостатка страниц) и чрезмерно общий (проблема избытка страниц). При наличии общего запроса ставится задача *дистилляции широких поисковых тем* с помощью авторитетных источников по этим темам.

HITS алгоритм использует такие понятия, как: авторитетный документ и хаб-документа (или авторитетная и хаб-страница). *Авторитетный документ* – это документ, соответствующий запросу пользователя, имеющий больший удельный вес среди документов данной тематики, то есть большее число документов ссылаются на данный документ. *Хаб-документ* – это документ, содержащий много ссылок на авторитетные документы.[4]

6 Возможность поиска похожих документов реализована в современных поисковых системах [52], например, Яндекс («*похожи на страницу*»), Google («*Find pages similar to the page*»). Достоинство такого вида поиска для пользователя – нужно нажать одну кнопку, для системы – документ содержит больше информации, чем запрос пользователя.

2 Детальный анализ алгоритма, постановка задачи, дополнительные замечания, а также поиск синонимов с помощью HITS алгоритма представлены в гл. 2, стр. 69.

3 Ещё одно название HITS алгоритма – «Connectivity analysis algorithm for hyperlinked environment» – предложено в работе [81].

4 Оригинальное расширение HITS алгоритма предложено в работе [136]. Авторы построили и проанализировали граф Темы-Системы для поиска наиболее успешных тем, выявляющих слабые и

Алгоритм PageRank (отличия от алгоритма HITS)

Параметр *PageRank* страницы *p(i)* (её авторитетность) определяется так [102]:

$$p(i)=\frac{q}{N}+(1-q)\sum_{j:j\to i}\frac{p(j)}{k_{out}(j)},\quad i=1,2,\ldots,N \qquad (1.1)$$

где N – общее число страниц; $j \to i$ обозначает гиперссылку от страницы j к странице i; $K_{out}(j)$ – это число исходящих ссылок страницы j; $(1-q)$ – амортизирующий коэффициент (damping factor)[1]. Набор уравнений (1.1) решается итеративно.

Оба алгоритма: PageRank и HITS предлагают общую идею итеративного вычисления авторитетных страниц, где авторитетность определяется наличием (количеством) и характером (степень авторитетности источника) ссылок. Однако есть и разница. Отличие алгоритма PageRank от алгоритма HITS в том, что у каждой страницы только один параметр, который соответствует её популярности, это вес PageRank. В алгоритме HITS каждой странице сопоставлено два параметра, которые определяют авторитетность и наличие ссылок на авторитетные страницы. Это соответственно параметры *authority* и *hub*. Отметим, что PageRank не требует дополнительных вычислительных затрат во время обработки запроса, HITS более дорогой с вычислительной точки зрения алгоритм.

В работе [89] указывают на сходство результатов работы HITS и PageRank[2]. Большинство документов, полученных как авторитетные в HITS, были представлены в результатах PageRank, но упорядочены были по другому. Однако в другой работе [80] при поиске авторитетных страниц с помощью алгоритмов HITS и PageRank по всей Английской Википедии и по некоторым подмножествам страниц (например: People, Historical Events,

---

сильные поисковые системы, на основе данных TREC. Поскольку результаты работы поисковиков могут быть неудачными, постольку веса могут быть отрицательными, поэтому авторы вводят понятие *неавторитетности* (*unauthority*) и выделяют тип вершин, ссылающихся на неавторитетные вершины (*unhubness*). Тогда «хаб-документ – это документ, содержащий много ссылок на неавторитетные вершины, причём ссылки имеют большой отрицательный вес» [136].

1 О выборе амортизирующего коэффициента в алгоритме PageRank см. в работе [73].

2 В работе [89] авторы используют в Байесовой сети доверия веса, рассчитанные с помощью HITS, для поиска релевантных документов. С помощью HITS посчитали четыре веса для каждого документа: *hub* и *authority*, локальные (по запросу), глобальные (по всем документам).

Countries and Cities)[1] были получены в общем разные классы концептов. Также в работе [145] приводятся эксперименты по сравнению различных методов поиска похожих статей в ВП, в том числе сравнивают работу алгоритмов LocalPageRank[2] и PageRankOfLinks (аналог PageRank) в пользу последнего. Авторы [145] делают вывод, метод HITS ищет *хуже* похожие статьи в ВП, чем PageRank, поскольку считают метод LocalPageRank аналогичным методу HITS. Скорее всего, это не верный вывод, поскольку LocalPageRank и HITS – разные алгоритмы – они используют разное число весов для каждой страницы. Таким образом, требуются дополнительные исследования по сравнению алгоритмов с помощью экспериментов.

О популярности алгоритма PageRank говорит наличие нескольких его модификаций. В работе [145] предложены методы Green, SymGreen для поиска «вершин, семантически связанных с исходной». Эти методы являются модификациями алгоритма PageRank на основе Марковских цепей.

Алгоритм распределения рангов (ArcRank)

Алгоритм предназначен для поиска семантически близких слов, а не синонимов. ArcRank вычисляет рейтинг вершин аналогично алгоритму PageRank. Небольшая модификация в том, что вершинам источникам[3] и висячим вершинам[4] не присваиваются предельные значения.

В ArcRank дугам назначаются веса на основе весов вершин. Если $|a_s|$ – число исходящих дуг из вершины s, p_t – вес вершины t, тогда важность (relevance) дуги (s,t) определяется так [174]: $r_{s,t}=\frac{p_s/|a_s|}{p_t}$

При выборе слов, связанных с w, первыми выбираются те рёбра, инцидентные w, которые имеют больший вес.

---

1 Подмножество определяется благодарю наличию категорий у страниц. Например статья в русской Википедии: «Куинджи, Архип Иванович» содержит такие категории, как: «Художники России», «Передвижники» и др., см. http://ru.wikipedia.org/wiki/Куинджи,_Архип_Иванович

2 В работе [145] указан недостаток LocalPageRank, присущий и HITS: в сильно связанном графе (например ВП), соседние вершины (для данной) – это значительная часть всего графа.

3 Нет входящих дуг, то есть слова не упоминаются ни в одной словарной статье.

4 Нет исходящих дуг, то есть словам не даны определения, они не встречаются в заголовках словарных статей.

Алгоритм WLVM

Алгоритм векторной модели ссылок Википедии (*англ.* Wikipedia Link Vector Model или WLVM) вычисляет сходство двух статей ВП на основе содержащихся в них ссылок [134]. Алгоритм включает шаги:

1. по заданному термину получить все статьи ВП с похожими заголовками;
2. обработать ссылки (разрешить «редиректы»[1]; для ссылок на страницы «дизамбиги»[2] взять все ссылки, перечисленные на «дизамбигах»);
3. подсчитать вес ссылок (см. ниже);
4. построить вектор (исходящих) ссылок для каждой страницы;
5. из множества пар статей (для двух терминов) выбираются наиболее похожие, то есть с наименьшим углом между векторами ссылок.

Семантическое сходство двух страниц ВП определяется углом между векторами ссылок этих страниц. Сходство будет выше, если обе страницы ссылаются на страницу, на которую мало ссылаются другие страницы.

Вес ссылки с исходного документа на целевой определяется по правилам:

- 1 или 0, если есть или нет такая ссылка в исходном документе;
- обратно пропорционально общему числу ссылок на целевой документ.

А именно, вес ссылки w со страницы a на страницу b рассчитывается по формуле: $w(a \to b) = |a \to b| \cdot \log \sum_{x=1}^{t} \frac{t}{|x \to b|}$, где t — общее число страниц в ВП.

Для оценки алгоритма использовался тестовый набор 353-TC.[3] Была предпринята малоуспешная попытка *автоматически* выбирать верное значение для ссылок на многозначные статьи: коэффициент корреляции Спирмена с эталонным набором оказался равным 0.45, при разрешении

1 «Редирект» – это страница-перенаправление; «*разрешить* редирект» означает подменить ссылки x→y→z на x→z, спрямляя путь до целевой статьи.

2 «Дизамбиги» – статьи Википедии, содержащие перечисление значений. Создаются для многозначных терминов, см., например, статью «Лемма».

3 Тестовый набор подробно описан на стр. 127. Сравнение результатов работы алгоритма WLVM с другими см. в табл. 4.6 на стр. 130.

многозначных статей *вручную* — 0.72. Доступна реализация WLVM алгоритма.[1]

Алгоритмы построения и анализа ссылок: Similarity Flooding, алгоритм извлечения синонимов из толкового словаря и другие

Одна из задач, рассматриваемых в диссертации[2], относится к типу задач scheme / ontology alignment (другое название – scheme / ontology matching). Суть этой более общей задачи в том, что «на входе есть две схемы / онтологии, каждая из которых состоит из дискретных сущностей (например, таблицы, XML элементы, классы, свойства, правила, предикаты), на выходе должны быть получены отношения (например отношение эквивалентности, отнесение к некоторой категории), связывающие эти сущности» [164].

Этот тип задач замечателен тем, что включает и лексическую[3], и семантическую[4] компоненты. Обзор современных работ по scheme / ontology matching представлен в [164], критический обзор открытых (open source) программных систем изложен в [190].

Кратко опишем несколько работ, посвящённых данной теме. Затем более подробно осветим алгоритм Similarity Flooding и алгоритм извлечения синонимов из толкового словаря.

Итак, в работе [77] для обнаружения неявных (latent) связей между вершинами считают число общих соседей двух вершин. Сходство двух вершин (relevance measure) вычисляется как отношение числа общих вершин к числу всех соседних вершин ([77], стр. 3). Авторы поставили себе общую задачу – обнаружение отношений в семантическом графе[5]. Эксперименты

---

1 Программа Wikipedia Miner Toolkit, см. http://sourceforge.net/projects/wikipedia-miner.

2 В данной работе предлагается оригинальный алгоритм поиска похожих вершин графа, см. гл. 2.

3 Примером использования лексической компоненты в задачах scheme / ontology alignment может быть задача сравнения графов (graph matching), задача сравнение строк (например, сравнение имён) и др. [164].

4 Пример семантической компоненты – использование формальной онтологии верхнего уровня SUMO, DOLCE (там же, стр. 9).

5 Семантический граф – это граф, который содержит вершины разных типов и отношения между ними, также разных типов [77]. Семантическая сеть (semantic network) – это направленный граф с вершинами, соответствующими концептам, и рёбрам, представляющим семантические отношения между концептами

заключались в предсказании возможных атак и уязвимостей служб безопасности на основе видеофильмов и данных терактов.

Поиск похожих вершин в графах возможен с помощью поиска соответствий между вершинами (отображений). Во-первых, точное соответствие, одна вершина к одной (рассматривается в данной работе, стр. 86). Во-вторых, *неточное* – многие к одной или многие к многим, то есть кластер концептов отображается в один концепт. В работе [97] предложен алгоритм неточного сравнения графов онтологий[1] на основе максимизации ожидания[2]. Онтология представлена в виде направленного графа с метками.

Алгоритм Similarity Flooding

Кратко опишем один из алгоритмов, решающих описанную выше задачу scheme / ontology alignment, – алгоритм *Similarity Flooding*, ключевая идея которого в том, что «два элемента считаются сходными, если сходны их соседи. Сходство двух элементов распространяется на их соседей» [132]. На первом шаге в соответствии с входными схемами, которые нужно сравнить (например SQL таблицы или запросы), строятся два графа. На втором шаге задаются начальные значения сходства между вершинами двух графов, путём сравнивая имён вершин. На третьем шаге итеративно вычисляется сходство вершин до тех пор, пока суммарное изменение степени сходства по всём вершинам больше наперёд заданного ε. Значение сходства между вершинами учитывается для вычисления сходства соседних вершин. На четвёртом шаге выбираются наиболее похожие пары вершин.

Таким образом, алгоритм *Similarity Flooding* находит похожие вершины, принадлежащие разным графам, то есть вычисляет меру сходства вершин графа. Разработанный и рассматриваемый в диссертации алгоритм (стр. 86) предназначен для поиска вершин, похожих на заданную, в одном и том же графе. В этом отличие разработанного алгоритма от алгоритма *Similarity Flooding*.

(см. http://en.wikipedia.org/wiki/Semantic_network).

1 В таком графе концептам соответствуют вершины, отношениям – дуги.

2 См. http://en.wikipedia.org/wiki/Expectation-maximization_algorithm.

Алгоритм извлечения синонимов из толкового словаря

В работах [84], [83], [174] французские учёные, под руководством Винсента Блондела (Vincent Blondel), предлагают обобщение HITS алгоритма для поиска синонимов в толковом словаре.[1]

Предположим, что (1) синонимы имеют много общих слов в определениях (в статьях толкового словаря); (2) синонимы часто употребляются в определении одних и тех же вокабул[2]. Построим граф G, в котором вершины соответствуют вокабулам словаря. Строится дуга от вершины u к v, если слово, соответствующее v, встречается в определении вокабулы u.

Пусть ищем синонимы для слова w. Для этого строим G_w (подграф графа G), включающий (1) все вершины из G, ссылающиеся на w и (2) все вершины, на которые ссылается w. Считаем сходство слов относительно слова w, чтобы выбрать лучшие слова как синонимы.

Сходство между вершинами графа вычисляется так. Каждой вершине i графа G_w назначаем три веса x_i^1, x_i^2, x_i^3 с начальным значением единица. Веса обновляются итеративно по такому правилу:

$$\begin{cases} x_{i1} \leftarrow & \sum_{j:(i,j)\in E} x_{i2}, \\ x_{i2} \leftarrow & \sum_{j:(j,i)\in E} x_{i1} + \sum_{j:(i,j)\in E} x_{i3}, \\ x_{i3} \leftarrow & \sum_{j:(j,i)\in E} x_{i2}, \end{cases}$$

- Вес x_i^1 равен сумме весов x_j^2 всех вершин j, на которые указывает вершина i;
- Значение x_i^2 равно сумме весов x_j^3 (вершин, на которые указывает вершина i) и x_j^1 (вершин, указывающих на i).
- Значение x_i^3 равно сумме весов x_j^2 вершин, указывающих на i.

На каждом шаге веса одновременно обновляются и нормализуются. Синонимами считаются слова с максимальным значением x_i^2 (центральный

1 Обобщение, поскольку Блондел вводит понятие *структурный граф* и получает, что в алгоритме Клейнберга HITS [125] структурный граф состоит всего из двух вершин (hub → authority), при этом все вершины рассматриваемого графа G имеют по два веса (hub и authority), которые показывают насколько эти вершины графа G похожи на вершину hub и authority соответственно. Для поиска синонимов в словаре Блондел предлагает использовать структурный граф из трёх вершин (1→ 2 → 3), таким образом, вершины графа словаря будут иметь по три весовых значения.

2 Вокабула – заголовок словарной статьи (лексикографический термин).

вес). Вычисления проводились на английском словаре Вебстера. Построенный граф содержал 112 169 вершин, 1 398 424 рёбер.

Алгоритмы статистического анализа текста: ESA, поиск контекстно-связанных слов

Алгоритм ESA

В работе [103] авторы описали алгоритм ESA, позволяющий представить значение любого текста в терминах концептов ВП.[1] Оценка эффективности метода выполнена за счёт автоматического вычисления степени семантической связи между фрагментами текста произвольной длины на ЕЯ. Для ускорения работы построили инвертированный индекс: слову соответствует список концептов, в статьях которых оно появляется. Была выполнена предобработка концептов ВП:

- удалили концепты, которым соответствуют небольшие статьи (меньше 100 слов, меньше 5 исходящих и входящих ссылок);
- удалили стоп-слова и редкие слова;
- получили леммы слов (тексты на английском языке).

В алгоритме ESA на вход подаются два текста. По ним строятся два вектора из концептов ВП следующим образом. По фрагменту текста (1) строится вектор по TF-IDF схеме, (2) из инвертированного индекса выбираются концепты и объединяются во взвешенный вектор[2]. Произведение этих векторов и даёт вектор концептов ВП релевантных фрагменту текста. Для сравнения текстов сравнивают два вектора, например, с помощью косинусного коэффициента.

Эксперименты в работе [103] показали преимущество ESA в точности поиска семантически близких слов в ВП по сравнению с алгоритмами поиска

1 ESA – это аббревиатура от «Explicit Semantic Analysis» (явный семантический анализ). Название выбрано в противовес «скрытому семантическому анализу» (LSA), поскольку в ESA концепт – это название статьи. То есть концепты явные, формулируются человеком, легко объяснить их значение.

2 Причём связь слова и концепта не указывается, если концепты получили небольшой вес для данного слова. Для вычисления взвешенного вектора используется параметр k – запись в инвертированном индексе, указывающая на степень связи слова и концепта. Из статьи [103] не ясно, как вычисляется эта степень связи слова и концепта, возможно, как (1) относительное число повторов слова в статье, (2) позиция в тексте (чем ближе к началу статьи, тем больше вес).

LSA [100], WikiRelate! [173] и других, выполняющими поиск на основе данных WordNet, Роже и ВП. Достоинство метода также в том, что он позволяет определять значение многозначного слова.

Метод извлечения контекстно связанных слов

Метод извлечения контекстно связанных слов на основе частотности словосочетаний предлагается в [146] для поиска контекстно похожих слов (КПС) и для машинного перевода. Данными для поиска КПС служат (1) семантически близкие слова из тезауруса, (2) словосочетания из базы данных (БД) с указанием типа связи между словами. Для слова *w* формируется *cohort w*, то есть группа слов, связанных одинаковыми отношениями со словом *w*, из базы словосочетаний. КПС слова *w* – это пересечение множества похожих слов (из тезауруса) с *cohort w*. Работа [146] интересна формулами, предлагаемыми для вычисления сходства между группами слов.

Обычно вычисляется сходство между отдельными парами слов. В работе [146] вычисляется сходство между группами слов G_1 и G_2 на основе формул, предложенных в [122]. Вершины графа – слова, взвешенные рёбра указывают степень сходства между словами (таким образом, матрица инциденций – *sim* – матрица сходства, хранит сходство между отдельными элементами). Вычисляются:

AI – absolute interconnectivity (абсолютная связность), как суммарная сходство между всеми парами в группах: $AI(G_1, G_2) = \sum_{x \in G_1} \sum_{y \in G_2} \mathrm{sim}(x, y)$.

AC – absolute closeness (абсолютная близость или плотность) определяется как среднее сходство между парами элементов: $AC(G_1, G_2) = \frac{1}{|G_1| \cdot |G_2|} \cdot \mathrm{AI}(G_1, G_2)$

Разница между *AI* и *AC* в том, что в *AC* учитывается пары имеющие нулевое сходство (рис. 1).

В [122] предлагает нормализовать абсолютную связность и близость за счёт вычисления внутренней связности и близости отдельных групп. Внутренняя связность и близость определяются на основе вычисления разбиения каждой группы (поиск за *O(N)* минимального числа рёбер, удаление которых приведёт к разбиению графа на две части).

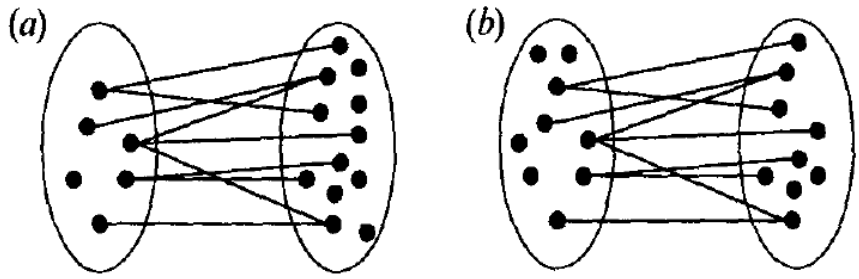


Рис. 1. Пример, иллюстрирующий разницу между мерами *AI* и *AC*. Значение *AI* в случаях (a) и (b) остаётся постоянным, но значение *AC* в случае (a) больше, поскольку в (b) больше вершин, не имеющих похожих вершин [146]

Можно упомянуть ещё ряд статистических алгоритмов для вычисления семантической близости слов: LSA [100], PMI-IR [180].

Метрики

Выделяют несколько способов определения похожих документов[1] [53]. Полагаем, что документы и запросы представляются с помощью индексных терминов или ключевых слов. Обозначим посредством символа |.| – размер множества ключевых слов, представляющих рассматриваемый документ. *Простой коэффициент соответствия* $|X \cap Y|$ показывает количество общих индексных терминов. При вычислении коэффициента не берутся в рассмотрение размеры множеств X и Y.

Таблица 1.1

Коэффициенты сходства для документов, для ключевых слов [53]

Формула	***Название***
$\frac{\lvert X \cap Y \rvert}{\lvert X \rvert + \lvert Y \rvert}$	Коэффициент Дайса (dice)
$\frac{\lvert X \cap Y \rvert}{\lvert X \cup Y \rvert}$	Коэффициент Джаккарда (jaccard)
$\frac{\lvert X \cap Y \rvert}{\lvert X \rvert^{1/2} \cdot \lvert Y \rvert^{1/2}}$	Косинусный коэффициент

1 Речь идёт о текстовых документах, а не о интернет страницах, то есть нет ссылок. Документам можно поставить в соответствие вершины графа. Если степени сходства документов сопоставить расстояние между вершинами, то более похожим документам будут соответствовать более близкие вершины.

$\frac{\lvert X \cap Y \rvert}{min(\lvert X \rvert, \lvert Y \rvert)}$	Коэффициент перекрытия

В таблице 1.2 показаны способы вычисления степени сходства, основанные на учёте:

- частотности слов в корпусе;
- расстояния в таксономии;
- одновременно и частотности слов, и расстояния в таксономии.

Таблица 1.2

Классификация метрик и алгоритмов вычисления степени сходства слов

Формула / ссылка на описание алгоритма	***Название***
1. Учёт частотности слов в корпусе	
$NGD(x,y)=\frac{max(\log f(x), \log f(y)) - \log f(x,y)}{\log M - min(\log f(x), \log f(y))}$	Нормализованное расстояние Google (NGD)[1]
$jaccard(x,y)=\frac{Hits(x\ AND\ y)}{Hits(x)+Hits(y)-Hits(x\ AND\ y)}$	jaccard[2] [173]
Описание алгоритма см. на стр. 34.	ESA [103]
2. Учёт расстояния в таксономии[3]	
Расстояние соответствует числу ребер кратчайшего пути между концептами	Метрика применялась для концептов Тезауруса Роже [117]
$lch(c_1, c_2)=-\log\frac{length(c_1, c_2)}{2D}$	Leacock & Chodorov 1997, [99] стр. 265-283
$wup(c_1, c_2)=\frac{lcs_{c_1, c_2}}{depth(c_1)+depth(c_2)}$	Wu & Palmer [186]
$res_{hypo}(c_1, c_2)=1-\frac{\log(hypo(lcs_{c_1, c_2})+1)}{\log(C)}$	Метрика *res* [151], адаптированная к таксономии категорий ВП [173]

1 NGD расшифровывается как *normalized Google distance*, http://en.wikipedia.org/wiki/Semantic_relatedness

2 Формула вычисления сходства слов с помощью коэффициента Джаккарда (см. предыдущую таблицу) предложена в работе [173], где *Hits(x)* – определяется как число страниц, которые возвращает Google для слова *x*.

3 В англоязычной литературе такие метрики называют: *path based measures*, *edge counting methods* [173].

3. Учёт частотности слов и расстояния в таксономии	
$res(c_{1,}c_2)=max_{C\in S(c_1,c_2)}[-\log(P(C))]$	Расстояние res [151], [152]
$lin(c_{1,}c_2)=\frac{2*\log(P(c_0))}{\log P(c_1)+\log P(c_2)}$	Расстояние lin[1] [128]
4. Учёт пересечения текста	
пересечение текста (глоссы WordNet)	Lesk, 1986 [74]
extended gloss overlap – пересечение глосс с учётом глосс соседних концептов WordNet	Banerjee & Pedersen, 2003 [75]
$relate_{gloss/text}(t_{1,}t_2)=\tanh\frac{overlap(t_{1,}t_2)}{length(t_1)+length(t_2)}$	[173]

Расстояние Google – это мера семантической связности, вычисленная на основе числа страниц, полученных с помощью поисковика Google для заданного набора ключевых слов. В таблице приведена формула вычисления нормализованного расстояния Google (*NGD*) для двух слов: *x* и *y*, где *M* – это общее число веб-страниц, проиндексированных Google; *f(x)* и *f(y)* – число страниц, содержащих ключевые слова *x* и *y*, соответственно; *f(x, y)* – число страниц, содержащих сразу и *x,* и *y*. Если *x* и *y* на всех страницах встречаются вместе, то полагают *NGD=0*, если только по отдельности, то *NGD*=∞.

Выделим класс метрик, вычисляющих сходство на основе данных таксономии (табл. 1.2). Данные метрики используются для вычисления сходства концептов WordNet [74], [99], [128], [151], [152], [186], GermaNet [139], ВП [173].

В книге [99] Leacock и Chodorov предложили вычислять близость концептов как расстояние между концептами в таксономии, нормализованное за счёт учёта глубины таксономии. В формуле $lch(c_{1,}c_2)=-\log\frac{length(c_{1,}c_2)}{2D}$, функция *length(c_1, c_2)* – это число вершин вдоль кратчайшего пути между вершинами c_1 и c_2; *D* – максимальная глубина таксономии. В работе [99] авторы рассмотрели только одно отношение *is-a* и только между существительными.

1 Данная метрика получила развитие в работе [146], см. метод извлечения контекстно связанных слов, стр. 35.

В работе [186] предложена формула, учитывающая как глубину концептов в иерархии, так и глубину ближайшего общего родителя *lcs* (least common subsumer): $wup(c_{1,}c_2)=\frac{lcs_{c_1,c_2}}{depth(c_1)+depth(c_2)}$.

Резник [151] предложил считать, что два слова тем более похожи, чем более информативен концепт, к которому соотнесены оба слова, то есть чем ниже в таксономии находится общий верхний концепт (синсет в WordNet).[1] При построении вероятностной функции *P(C)*, потребуем, чтобы вероятность концепта не уменьшалась при движении вверх по иерархии: $res(c_{1,}c_2)=max_{C\in S(c_1,c_2)}[-\log(P(C))]$. Тогда более абстрактные концепты будут менее информативны. Резник предложил оценивать вероятность через частоту синонимов концепта в корпусе таким образом: $P(C)=\frac{freq(C)}{N}$, $freq(C)=\sum_{n\in words(C)} count(n)$, где *words(C)* – это существительные[2], имеющие значение *C*; при этом *N* – общее число существительных в корпусе. Пусть, если для двух концептов ближайшим общим концептом является корневая категория, то сходство равно нулю.

В работе [173] метрика Резника *res* была адаптирована к ВП и информативность категории *P(C)* вычислялась как функция от числа гипонимов (категорий в ВП), а не статистически[3] (то есть не посчитали частотность термов в ВП):

$$res_{hypo}(c_{1,}c_2)=1-\frac{\log(hypo(lcs_{c_1,c_2})+1)}{\log(C)} ,$$

где *lcs* — ближайший общий родитель концептов c_1 и c_2, *hypo* — число гипонимов[4] этого родителя, а *C* — общее число концептов в иерархии.

---

1 Заметим, что в ВП у слова обычно несколько категорий, то есть может быть несколько ближайших общих категорий.

2 В экспериментах Резник оценивал сходство существительных, учитывал отношение WordNet *IS-A* (гипонимия).

3 Возможно, это одна из причин, почему мера res_{hypo} показала в экспериментах [173] относительно слабый результат.

4 Гипонимы категории *K* в Википедии – это все подкатегории *K*, а также все статьи, принадлежащие этим подкатегориям и категории *K*.

Lin [128] определил сходство объектов *A* и *B* как отношение количества информации, необходимой для описания сходства *A* и *B*, к количеству информации, полностью описывающей *A* и *B*. Для измерения сходства между словами Lin учитывает частотное распределение слов в корпусе текстов (аналогично мере Резника): $lin(c_{1,}c_2)=\frac{2\cdot\log(P(c_0))}{\log P(c_1)+\log P(c_2)}$, где c_0 – ближайший общий супер-класс в иерархии для обоих концептов c_1 и c_2. P – вероятность концепта, вычисляемая на основе частоты появления концепта в корпусе. Отличается от формулы *res* способом нормализации, корректным вычислением *lin (x, x)* (не зависит от положения концепта *x* в иерархии), учитывает наличие и общих, и различающихся свойств у объектов [152].

В работе [173] мера *lesk*, основанная на вычислении степени пересечения глосс концептов WordNet, была адаптирована к ВП (за глоссу авторы взяли первый абзац в статье ВП). Итак, сходство двух текстов t_1, t_2 вычисляется с двойной нормализацией (по длине текста и с помощью гиперболического тангенса) так:

$$relate_{gloss|text}(t_{1,}t_2)=\tanh\frac{overlap(t_{1,}t_2)}{length(t_1)+length(t_2)} \ ,$$

$overlap(t_{1,}t_2)=\sum_n m^2$, где пересекаются *n* фраз, *m* слов[1].

В работе [139] (стр. 4) приведённая в таблице 1.2 формула *lin* была адаптирована к поиску в структуре GermaNet. В данной работе приведены две TF-IDF схемы для вычисления сходства между запросом и текстом документа.

Глава о метриках была бы неполной без упоминания того, что кроме сходства, метрики позволяют вычислять степень различия объектов. Так например, в задачах кластеризации используются функции, определяющие *степень различия*[2] между документами. Если P – множество объектов,

1 Закон Ципфа утверждает, что чем длиннее фраза, тем реже она встречается в корпусе. На основании этого, было предложено наличие общих фраз длиной в *n* слов (в глоссах сравниваемых слов) оценивать как n^2 [75].

2 Любая функция оценки *степени различия* между документами *D* может быть преобразована в функцию, определяющую *степень соответствия S* следующим образом: $S=(1+D)^{-1}$.

предназначенных для кластеризации, то функция D определения степени различия документов удовлетворяет следующим условиям [53]:

1. $D(X,Y)\geq 0$ для $\forall X,Y\in P$
2. $D(X,X)=0$ для $\forall X\in P$
3. $D(X,Y)=D(Y,X)$ для $\forall X,Y\in P$
4. $D(X,Y)\leq D(X,Z)+D(Z,Y)$ для $\forall X,Y,Z\in P$

При анализе свойств Интернет сетей[1], при оценке свойств графов, созданных с помощью генератора случайных чисел, используют такие метрики [130]:

1. Распределение расстояний *d(x)* – число пар вершин на расстоянии *x*, делённое на общее число пар n^2 (включая пары типа *(a,a)*);

2. betweeness – мера центральности – взвешенная сумма числа кратчайших путей, включающих данную вершину (ребро);

3. вероятностное распределение вершин *P(k)* – число вершин степени *k* в графе;

4. правдоподобие (likelihood) – сумма произведений степеней смежных вершин;

5. кластеризация[2] *C(k)* – отношение среднего числа ссылок между соседями вершины степени *k* к максимально возможному числу таких ссылок C_k^2 .

1 Виды сетей, их топологические свойства и приложения см. в обзорных работах [70], [96], [142].

2 В работе [77] эта метрика называется *коэффициент кластеризации вершины* и вычисляется по формуле $C(i)=\frac{E_i}{k_i(k_i-1)/2}$, где k_i – степень вершины i, E_i – число ссылок между k_i соседями. Усреднение *C(i)* по всем вершинам даёт коэффициент кластеризации графа.

1.2 Системы и ресурсы для обработки текста

Автоматическая обработка текстов (АОТ) на естественном языке (ЕЯ) подразумевает наличие как программных систем[1], обрабатывающих тексты, так и корпусов, содержащих эти тексты. Общие проблемы создания программных систем рассматриваются в работе [12].

В данной работе под корпусом текстов понимают «набор текстов доступных для машинной обработки, на основе которых можно проводить какие-либо лингвистические исследования» [133].

Проблема отсутствия общепринятых стандартов для корпусов текстов приводит к тому, что для каждого отдельного корпуса создаётся своя система АОТ. Одно из решений этой проблемы, реализованное в виде системы GATE, предлагают английские учёные из университета Шеффилд.

GATE

Система GATE (General Architecture for Text Engineering) предлагает инфраструктуру для разработки и внедрения программных компонент с целью обработки текста на ЕЯ. Эта система (i) определяет архитектуру, то есть способ организации данных и программных компонент, обрабатывающих текст, (ii) предлагает реализацию архитектуры (набор классов, который может встраиваться в программные приложения независимо от GATE), (iii) помогает разрабатывать и использовать компоненты с помощью графического инструментария [92].

Система GATE написана на языке Java [8], [25], [68], [115], имеет модульную структуру, предоставляется на правах лицензии GNU library licence[2]. С научной точки зрения достоинство GATE заключается в возможности проводить численные измерения текста, которые можно повторить. В работе [109] критикуют систему GATE за плохую масштабируемость и за то, что она плохо справляется с большими

1 На сегодняшний день существует огромное количество программных систем для поиска и обработки текста, см. например, каталог программ Data Mining (http://www.togaware.com/datamining/catalogue.html и http://www.togaware.com/datamining/gdatamine/index.html).

2 Это ограниченная форма лицензии GNU, позволяющая, в случае необходимости, встраивать GATE в коммерческие продукты

коллекциями документов. Однако отмечают «большой потенциал GATE как инструмента для планирования и разработки приложений АОТ в области Information Extraction».

Модульность архитектуры позволяет (i) включать только необходимые компоненты, (ii) использовать имеющиеся наработки, (iii) начать работу с уже существующими компонентами, по мере необходимости, создавая новые.

В системе GATE можно выделить компоненты, не зависящие от языка текста (например Doc Reset) и зависящие (например, English POS Tagger). Система GATE позволяет обрабатывать документы на таких языках, как: английский, французский, немецкий, арабский, китайский и др. Это определяется наличием соответствующего модуля.

В системе GATE предложен следующий способ автоматической аннотации текстовых документов.[1] GATE позволяет связать подстроку текста документа (слово, фраза, предложение) с аннотацией. Аннотации описывают иерархическое разбиение текста, простой пример – это разбиение текста на слова (tokens). Более сложный пример (при полном синтаксическом анализе) – это декомпозиция предложения на именную, глагольную группы слов с выделением главного слова и т. п.

Отсутствуют (по крайней мере, неизвестны автору) *доступные* модули в системе GATE для морфологической обработки русского языка. Возможно, поэтому система GATE редко упоминается в системах обработки текстов на русском языке. Отрадное исключение представляют работы [64], [98], где представлены архитектура и реализация системы OntosMiner.

Для оценки качества функционирования систем Information Extraction (IE) используются такие метрики, как: точность (Precision), полнота (Recall) и качество (F-measure)[2]. В работе [64] предлагается новая система метрик, в

1 В проекте ALVIS предложен иной формат лингвистической аннотации (платформа Ogmios) для индексирования документов определённой тематики [109], [141]. Лингвистическая аннотация, добавляемая к текстовым единицам (словам, фразам, и т.д.) включает морфологические и синтаксическая теги, синтаксические отношения и семантические отношения (анафорические и специфичные для данной проблемной области).

2 Эти, а также ещё десяток других мер для оценки работы ранжирующих методик представлены в работе [156].

которой «аннотация представляется в формате, где явно специфицированы тип выделенного объекта (отношения) и его атрибуты, а также расположение аннотации в тексте относительно его начала (Offsets)». С одной стороны, указание типа объекта и положения подстроки в тексте (Offsets) сужает понятие объекта (именно объектами оперируют метрики точность, полнота и качество). С другой стороны, новые метрики подходят для оценки качества функционирования IE систем, построенных на основе GATE, поскольку тип объекта и положения подстроки в тексте включены в аннотации GATE.

Небольшой обзор систем, подобных GATE, а именно: KIM, TEXTRACT, Textpresso, Ogmios, представлен в работе [109].

Проект Диалинг

В данном подразделе дано краткое описание модулей автоматической обработки текста и морфологических словарей, разработанных рабочей группой Aot.ru [60]. Изначальный проект, посвящённый разработке русско-английского машинного перевода, назывался Диалинг. Разработанный процессор Диалинг включает графематический, морфологический и синтаксическим модули. Программная реализация процессора выполнена на языке C++. «Неоспоримым достоинством процессора Диалинг является его завершённость: программная реализация доведена до уровня промышленного использования, – система характеризуется приемлемой скоростью анализа и устойчивостью на открытом пространстве реальных текстов» (цит. по [47]).

Морфологический словарь, или лексикон, содержит все словоформы одного из языков: английский, немецкий или русский. Словарь предоставляется в двух вариантах: с возможностью редактирования и в бинарном варианте. Оболочка редактирования словаря позволяет выполнять: (i) поиск в словаре по лемме, словоформе, морфологической интерпретации, (ii) редактирование словаря. Словарь в бинарном формате предоставляет возможность выполнять: (1) морфологический анализ (получение по словоформе леммы, её свойств, уникального ID леммы, морфологических характеристик входной словоформы[1] и (2) морфологический синтез

1 Каждая словоформа представляется множеством морфологических омонимов [47].

(получение по уникальному ID леммы всей парадигмы слова со всеми словоформами и их морфологическими характеристиками). Бинарное представление словаря оптимизировано для проведения морфологического анализа. Основу этого представления составляет конечный автомат. Работает морфологическое предсказание слов, отсутствующих в словаре [60].

В прикладной части данной диссертационной работы для нормализации слов используется программа морфологического анализа (Lemmatizer)[1]. Например, для текста «смерч обрушился на южные селенья» нормализованным вариантом будет – «смерч обрушиться на южный селение» [31].

Тезаурусы WordNet, РуТез, Викисловарь

Тезаурус – это сложный компонент словарного типа, отражающий основные соотношения понятий в описываемой области знаний [41]. Тезаурусы включают всю терминологию, специфическую для предметной области (ПО), а также парадигматические отношения[2] между понятиями ПО. Тезаурус может выполнять разные функции в разных системах [41]:

- является *источником специальных знаний* в узкой или широкой ПО, способом описания и упорядочения терминологии ПО;
- является *инструментом поиска* в ИПС [17];
- является инструментом ручного индексирования документов в ИПС (так называемый *контролирующий словарь*);
- является *инструментом автоматического индексирования* текстов.

Одним из наиболее успешных проектов, связанных с тезаурусами, является WordNet[3] [99] – тезаурус английского языка, представляющий состав и структуру лексического языка в целом, а не отдельных тематических областей [1]. WordNet группирует наборы слов со схожим значением в

1 Программа Lemmatizer (http://www.aot.ru) распространяется на условиях LGPL лицензии.

2 «*Парадигматические отношения* обусловлены наличием логических связей между предметами и явлениями, обозначаемыми словами. Такие отношения носят внеязыковой характер и не зависят от ситуации, для описания которой используются слова» [66]. Примерами парадигматических отношений являются отношения синонимии, антонимии.

3 См. http://wordnet.princeton.edu

синсеты[1] (от англ. synonym set, synset). WordNet содержит синсеты, краткие общие определения к синсетам (глоссы), примеры употреблений и несколько типов семантических отношений между синсетами. Авторы преследовали двоякую цель: объединить возможности тезауруса и наглядность словаря, а также создать ресурс для автоматической обработки текстов на естественном языке. База данных и программа выпущены на условиях BSD лицензии. Возможен онлайн доступ к содержимому базы данных.

WordNet был разработан в 1985 г. Работа над ним ведётся сотрудниками Лаборатории когнитологии Принстонского Университета (США) под руководством профессора психологии Дж. Миллера. К 2005 г. WordNet содержал около 150 тыс. слов, организованных в более чем 115 тыс. синсетов, всего 203 тыс. пар слово-значение. Словарь состоит из 4 файлов, соответствующих таким частям речи, как: существительное, глагол, прилагательное и наречие.

Семантические отношения связывают большинство синсетов. Представлены такие семантические отношения, как: гипонимия (родовидовое), меронимия (часть-целое), лексический вывод (каузация, пре-суппозиция) и др.

Гипонимия позволяет организовывать синсеты в иерархические структуры (деревья). Гипонимия связывает слова, «между содержанием понятий которых существует отношение семантического включения, то есть значение гиперонима полностью включено в значение гипонима» [1]. Например, значение слова *бояться* включено в значение слов *опасаться*, *остерегаться*.

Разработаны способы вычисления семантического расстояния между концептами либо словами с помощью тезауруса WordNet, например: мера Leacock-Chodorow [99],[2] меры на основе частотности концептов в корпусе (мера Резника [151], [152], мера Jiang-Conrath [120], мера Lin [128]), мера

1 С точки зрения теории графов системе WordNet соответствует направленный граф, вершины которого представлены концептами (наборы синонимов, синсеты), дуги представлены семантическими отношениями.

2 См. описание меры Leacock-Chodorow и других в табл. 1.2, стр. 37.

Hirst-St.Onge, мера пересечения расширенных глосс[1]. В работе [87] проведены эксперименты по сравнению пяти мер, вычисляющих семантическое расстояние между терминами WordNet. Эксперименты показали, что лучшие результаты даёт мера JiangConrath. Также обзор нескольких мер и эксперименты с ними представлены в диссертации итальянского учёного Calderan M. [90].

Данные WordNet используются для решения таких задач, как определение значения слова (WSD[2]) [138][3], [153], [187], вычисление логичности и связности предложений в тексте [110], [175], построение баз знаний [17] и тезаурусов.

В работе [154] авторы задались целью показать, что комбинация эвристик позволяет построить полную таксономию современного словаря на любом языке. В результаты были разработаны: (1) метрика расстояния между двумя словами (в двуязычном словаре) на основе таксономии гипонимов / гипернимов WordNet, (2) эвристики (и методика их интеграции) для определения значения (WSD) родовых терминов[4] двух словарей, (3) построена таксономия для испанского и французского языков на основе машинных словарей DGILE (испанский) и LPPL (французский).

Работа [121] интересна критикой WordNet. Авторы предложили итеративный способ решения задачи WSD на основе корпуса и словаря. Слова считаются похожими, если встречаются в похожих предложениях. Предложения похожи, если содержат похожие слова. Авторы разработали

---

1 С другой стороны глоссы WordNet критикуют за отсутствие единого стиля в их написании, считают также, что некоторые из них не очень информативны [158].

2 В свою очередь WSD (*word sense disambiguation*) методики успешно применяются в машинном переводе, основанном на статистическом подходе [88].

3 В работе [138] категориям (в классификации новостных тем) ищется соответствующий синсет WordNet. При этом категория может состоять из нескольких слов, то есть нет точно соответствующего слова в WordNet. Проблема была решена поиском подстроки среди слов WordNet (функция «Find Keywords by Substring»). Эту идею (поиск подстроки) можно применить при интеграции данных ВП и WordNet при поиске соответствия между словом из WordNet и названием статьи ВП, состоящей из нескольких слов.

4 В работе [154] под «родовым термином» (*англ.* genus term) подразумевается гипероним. Отношение гипонимии важно, так как является «основой таксономии и главным механизмом наследования, помогая в установлении других семантических отношений и свойств, обеспечивая строгую структуру, не обременённую многословием» [154]. Заметим также, что «в словарном определении заголовок статьи и “родовой термин” должны принадлежать одной части речи» [154].

меры сходства сходства слов и предложений, обладающие особенностями: ассиметричность, транзитивность, сходимость. Благодаря транзитивности данный метод позволяет оценивать сходство редких фраз, отсутствующих в корпусе. Были использованы данные словарей Webster, Oxford и WordNet. В экспериментах WordNet показал слабые результаты. Возможные причины таковы [121]:

- архитектура WordNet не предназначена для хранения данных о контекстном сходстве;
- расстояние в дереве WordNet (длина пути между концептами) не всегда соответствует интуитивным представлениям сходства слов, так как разные концепты находятся на разном уровне абстракции, имеют разное число гиперонимов.

Система WordNet используется во многих современных проектах, что, в свою очередь, приводит к появлению научно-исследовательских проектов, направленных на улучшение самой базы WordNet. В работе испанских учёных [158] предлагается использовать данные энциклопедии Википедия для расширения сети концептов WordNet. Авторы предлагают способ автоматического установления соответствия между статьями энциклопедии и концептами онтологии (здесь – семантической сети WordNet).[1] Для решения задачи авторы строят упрощённую версию Английской Википедии[2] таким способом, что из всех статей оригинальной Википедии были выбраны только те, заголовкам которых был найден соответствующий концепт в WordNet.[3] Для вычисления метрики сходства между статьёй Википедии и концептом WordNet использовалась модель VSM (Vector Space Model).

Далее будут описаны отечественные лингвистические базы данных и тезаурусы: каталог семантических переходов, тезаурус РуТез, Русский Викисловарь, а также тезаурус GEMET.

«Каталог семантических переходов» – база данных регулярно воспроизводимых лексико-семантических изменений,

1 Такое автоматическое установление соответствия является подзадачей автоматического построения онтологий, как верно замечают авторы [158].

2 Не путать с Википедией на английском упрощённом языке (Simple Wikipedia).

3 Этим объясняется небольшое количество статей в упрощённой Википедии (1841 статья на 15.11.2004)

засвидетельствованных в различных языках мира [21]. В этой БД выделено шесть типов семантических переходов (смысловых связей между словами), интересных с точки зрения изучения этимологии слов и создания этимологических словарей:

- *полисемия*;
- *семантическая эволюция* – изменение значения слова на разных временных срезах одного и того же языка;
- *когнаты* – «лексемы с двумя значениями, находящимися в отношении семантической производности, в родственных языках восходят к одной лексеме праязыка, в которой предположительно отсутствует соответствующая полисемия» [21];
- *заимствование* – семантическая адаптация иноязычных слов, в ходе которой может измениться значение слова;
- *морфологическая деривация* – образование новых значений при добавлении аффиксов (например «любить» – «любой»);
- *грамматикализация* – процесс превращения лексических единиц с ходом эволюции языка в грамматические показатели (например, глагол «стать» в конструкции «стану работать» означает начало действия в будущем времени).

Особенностью другой системы – тезауруса РуТез [41] является автоматическое индексирование. Термины тезауруса делятся на дескрипторы и варианты (синонимы) дескрипторов. Дескрипторы представлены отдельными существительными и именными группами. Синонимами могут быть две упомянутые грамматические группы, а также отдельные прилагательные, глаголы и глагольные группы. Применяются следующие правила включения дескрипторов в тезаурус:

1. Наличие связи с другими дескрипторами;
2. Наличие (если это словосочетание) таких тезаурусных связей, которые не вытекают из структуры словосочетания. Например, словосочетание *аренда земли* является свободным словосочетанием, и сумма значений его составляющих равна значению всего словосочетания, при этом, *аренда земли* является одним из видов

землепользования, и эта неочевидная связь служит основанием для включения этого словосочетания в тезаурус.

В РуТез включаются термины, не упоминавшиеся в текстах, если они: (а) нужны для объединения разрозненных дескрипторов, (б) пополняют ряд нижестоящих дескрипторов для уже существующего дескриптора. Предусмотрено включение многозначных терминов, а именно: несколько значений одного термина представляются разными дескрипторами. Если только одно значение многозначного термина включено в тезаурус, то дескриптор снабжается пометой «М». В тезаурус включены фразеологизмы, в состав которых входят термины тезауруса: например, *как с гуся вода*, *водой не разольёшь* и др. Отношения в тезаурусе (ВЫШЕ-НИЖЕ[1], ЦЕЛОЕ-ЧАСТЬ[2], АССОЦИАЦИЯ) позволяют представить тезаурус в виде связной иерархической сети (разрешена только одна компонента связности).

Достоин упоминания тезаурус GEMET[3]. Интересными особенностями тезауруса является привязка концептов ко многим языкам (в том числе к русскому), предоставление данных с помощью веб-сервиса (RDF). Авторы GEMET планируют улучшить данные тезауруса за счёт включения его в Английский Викисловарь[4] и отдачи со стороны пользователей Викисловаря.

Викисловарь является с одной стороны вики-ресурсом, поэтому в его пополнении может участвовать каждый, с другой – это толковый, грамматический, фразеологический, этимологический и многоязычный словарь, в том числе и тезаурус.[5] Русский Викисловарь содержит следующие

---

1 Y=ВЫШЕ(X), если можно утверждать, что X – это вид Y, например «государственная собственность» = ВЫШЕ(«государственное предприятие») [41]. Связь ВЫШЕ-НИЖЕ соответствуют отношению гипонимии в Викисловаре: X – это гипоним, Y – это гипероним.

2 Связи ЦЕЛОЕ-ЧАСТЬ соответствуют меронимы и холонимы в Викисловаре.

3 Аббревиатура GEMET (швед. *gem – скрепка, игра*) расшифровывается как GEneral Multilingual Environmental Thesaurus, см. http://www.eionet.europa.eu/gemet/about?langcode=en и http://en.wikipedia.org/wiki/GEMET

4 English Wiktionary, см. http://en.wiktionary.org.

5 См. http://ru.wiktionary.org, http://ru.wikipedia.org/wiki/Викисловарь.

семантические отношения: синонимы[1], антонимы[2], гиперонимы, гипонимы, согипонимы, холонимы, меронимы, паронимы[3], омонимы[4].

Разработка словарей требует огромных вложений времени и сил. Поэтому тезаурусы WordNet, Роже покрывают небольшую часть лексикона, и содержат мало имён собственных, неологизмов, жаргонных слов, специальной терминологии [103]. Можно надеяться, что благодаря вики-технологии такая ситуация не грозит Викисловарю. На 10.11.2007 Викисловарь содержал 130 тыс. слов и словосочетаний более чем на 180 языках.

Вики-ресурсы

Вики – это веб-сайт для совместной работы, где каждый может принять участие в правке статей. Вики-сайт предоставляет пользователям возможность изменять и добавлять страницы сайта.[5] Наиболее известный вики-ресурс – Википедия.

Ward Cunningham, разработчик первого вики-сайта WikiWikiWeb, первоначально описал вики как «простейшую онлайн базу данных, которая, возможно, работает».[6] Также вики – это часть программного обеспечения на стороне сервера, позволяющая пользователям коллективно создавать и редактировать содержания интернет страниц с помощью любого интернет браузера. Язык вики поддерживает гиперссылки (для создания ссылок между вики-страницами) и является более наглядным чем HTML и безопасным (использование JavaScript и Cascading Style Sheets ограничено).[7]

Концепция «свободного редактирования» имеет свои достоинства и недостатки. Открытость в редактировании текстов привлекает технически не подкованных пользователей, что позволяет развивать уже существующие вики ресурсы. К проблемам стоит отнести борьбу с вандализмом. Эта

1 См. http://ru.wiktionary.org/wiki/самолёт.
2 См. http://ru.wiktionary.org/wiki/сжимать.
3 См. http://ru.wiktionary.org/wiki/канифоль.
4 См. http://ru.wiktionary.org/wiki/бор.
5 См. http://en.wikipedia.org/wiki/Wiki
6 См. http://wiki.org/wiki.cgi?WhatIsWiki
7 См. http://en.wikipedia.org/wiki/Wiki

проблема решается благодаря возможности отката.[1] Другой вопрос – надёжность и достоверность информации – может быть решён либо с помощью внешних признаков (число правок, число авторов статьи, число внутренних ссылок), либо с помощью специальных программ.[2]

Таким образом, появился новый формат электронных документов – вики.[3] Причём в Интернете насчитывается уже 25 тыс. вики-ресурсов, из них 203 состоят из более чем 10 тыс. вики-статей.[4]

Одним из наиболее успешных вики проектов считается Википедия[5]. Если на конец 2005 г. существовало более 200 Википедий на разных языках с более чем 2.8 млн статей [184], то к августу 2007 г. более 253 Википедий содержало 8.1 млн статей, а к июню 2008 г. уже более 10 млн.[6]

Цель Wikimedia Foundation (организация, ответственная за работу Википедии) – обеспечение свободного доступа ко всем знаниям, накопленным человечеством. Кроме Википедии Wikimedia Foundation поддерживает и другие проекты: открытый медиа архив (Wikicommons), открытое хранилище электронных книг (Wikibooks), база новостей (Wikinews), многоязыковой словарь и тезаурус (Wiktionary) и другие. Стоит отметить тесную интеграцию данных проектов, например, рисунки, видео или аудио файлы (неотъемлимая часть современных энциклопедий) должны быть предварительно загружены в Wikicommons, эти файлы получают уникальный идентификатор (URL), используя который можно иллюстрировать энциклопедическую статью, поставив ссылку на данный медиа ресурс. Вышеперечисленные проекты разрабатываются совместными усилиями пользователей с помощью программного обеспечения MediaWiki.

---

1 Откат возможен, поскольку в БД хранится история всех правок, указывается кто, что и когда правил. Откат позволяет вернуться к предыдущей версии набора страниц или всей базы данных. Выполнить откат может любой пользователь

2 Надёжность статьи в Википедии можно оценить визуально с помощью специальной программы численной оценки степени доверия к тексту [69].

3 Появился новый формат электронных документов – вики, см. стр. 51. Особенности корпуса вики-текстов, позволяющие говорить о качественном изменении по сравнению с html страницами, перечислены на стр. 24.

4 См. http://s23.org/wikistats/largest_html.php?th=10000&lines=500, данные от 17.08.2007.

5 Это конкретный вики-ресурс, используемый в работе. См. http://en.wikipedia.org

6 См. список википедий http://meta.wikimedia.org/wiki/List_of_Wikipedias, данные от 17.08.2007.

Данные этих проектов распространяются по открытой лицензии GNU Free Documentation License.[1]

Корпус текстов вики-ресурса Википедия

Корпус текстов, представленный в энциклопедии Википедия, представляет несомненный интерес для вычислительной лингвистики[2] и, в частности, для задачи поиска синонимов. Есть несколько причин, позволяющих успешно работать с этим корпусом:

- заранее определён способ хранения документов энциклопедии в базе данных MySQL [26] (заданы таблицы, поля таблиц, связи между полями);
- существует программа MediaWiki (набор php-файлов) для просмотра и редактирования содержимого Википедии;[3]
- задана классификация текстов благодаря наличию у каждой статьи категорий, определяющих тематическую направленность. Категории для статьи выбираются авторами статьи из набора уже существующих категорий. Можно добавить новую категорию, связав её с уже существующими. (табл. 1 на стр. 184 перечисляет типы отношений в Википедии, в частности, *ассоциативные* – отношения между категориями);
- в энциклопедии представлено большое[4] количество статей (на русском[5], английском[6] и других языках) на различную тематику (наука, искусство, политика и др.), корпус содержит тексты и на самую современную тематику (база обновляется, буквально, каждый день);

---

1 Более подробно об авторском праве, интеллектуальной собственности и патентах, лицензировании в целом и лицензии GPL в частности см. в [62].

2 Наличие интереса к вики ресурсам и Википедии в научной общественности характеризуется появлением значительного количества публикаций по данной тематике, см. *Wiki Research Bibliography* (1) http://meta.wikimedia.org/wiki/Wiki_Research_Bibliography, (2) http://bibliography.wikimedia.de, а также *Wikipedia in academic studies* http://en.wikipedia.org/wiki/Wikipedia:Wikipedia_in_academic_studies.

3 См. http://www.mediawiki.org.

4 «Википедия содержит огромное количество тщательно организованных человеческий знаний» [103].

5 Русская версия Википедии содержит 171 тыс. статей по данным на 11 мая 2007 г. (http://ru.wikipedia.org/wiki/Служебная:Statistics). На 23 октября 2006 г. тексты содержали 22,5 млн слов и 650 тыс. лексем (http://ru.wikipedia.org/wiki/ВП:ЧС).

6 Английская Википедии содержит 1,8 млн статей (11 мая 2007 г). В мае 2005 Английская Википедия включала 512 млн. слов [157]. См. http://en.wikipedia.org/wikistats/EN/TablesDatabaseWords.htm, а также http://en.wikipedia.org/wiki/Wikipedia:Statistics

• энциклопедия общедоступна[1].

Временным недостатком Википедии можно считать неравномерное распределение качества (т.е. степень проработки и глубину изложения, формально определяемые по таким признакам, как: размер страницы, число авторов, число модификаций у страницы) и количества статей по разным тематическим направлениям. В частности, указывается перевес статей технического характера (в отличии, например, от филологического), что, возможно, объясняется составом авторов энциклопедии [114].

В исследовании исторического раздела Английской Википедии [157] указано, что из 52 исторических деятелей, представленных в специализированной энциклопедии «*American National Biography Online*» (18 тыс. статей) в Википедии содержится только половина, в энциклопедии *Encarta* – одна пятая.

Более интересные (популярные) темы проработаны лучше. Общее наблюдение таково, что новые (недавно созданные) статьи – небольшие по размеру и плохо написаны, однако статьи на популярные темы, имеющие сотни и тысячи правок, приближаются по размеру и качеству к статьям профессиональных энциклопедий [157].

Сетевой анализ[2] Английской Википедии как графа (вершины – это статьи, рёбра – ссылки) представлен в работе [80], в которой авторы показали, что Википедия – это единый связный граф.[3] Следующие эвристики, предложенные в [80], предлагается использовать для развития системы поиска синонимов Synarcher [34], [126]:

• объединять в один узел статьи, имена которых отличаются незначительно (регистром или пунктуацией);

• удалять все статьи, начинающиеся со слов: «List of...» (так как это просто наборы ссылок), в Русской Википедии такие статьи начинаются так: «Список...».

---

1 Базу данных Википедии можно скачать по адресу http://download.wikimedia.org, одно из описаний по установке Википедии см. на сайте http://synarcher.sourceforge.net

2 Сетевой анализ (часть теории графов) – это количественная оценка связности и расстояний в графах.

3 Более поздние работы отечественных исследователей показывают, что Википедия содержит изолированные статьи, не связанные с другими статьями внутренними ссылками. См. http://ru.wikipedia.org/wiki/Википедия:Проект:Связность.

В [162] выделяют три вида компонент в сетевой структуре: IN, OUT и компонента сильной связности (КСС). «Если пользователь начинает просмотр интернет страницы IN-компоненты, то затем он попадает в узел КСС и, возможно, в OUT-компоненту. Попав в OUT-компоненту, пользователь уже не сможет вернуться в исходную вершину. Однако пока пользователь находится внутри КСС, все вершины – достижимы и могут быть просмотрены повторно» [162]. Открытой задачей является практическая оценка статистических свойств входящих и исходящих ссылок[1], КСС и других компонент Википедии.

Кроме своей прямой энциклопедической функции Википедия, благодаря открытому доступу к её данным, служит для определения значения многозначных слов [166], может помочь в автоматическом поиске информации в запросно-ответных систем (question-answering service) и др. [157], является основой для автоматического построения многоязыкового тезауруса.[2]

Другие системы

Несмотря на своё название, система OpenCyc[3] не является полностью открытой: данные доступны для редактирования пользователям, но код программы недоступен для расширения разработчикам [183].

С точки зрения АОТ интерес может представлять такой модуль OpenCyc, как Dictionary Assistant (DA).[4] С помощью DA пользователи: (i) добавляют лексическую информацию, используемую системой CYC для обработки и генерации текстов на ЕЯ, (ii) выполняют привязку слов к концептам CYC.[5] Также DA позволяет строить отношения между концептами

1 Под статистическими свойствами вершин имеются в виду: средняя степень (числа входящих ссылок), максимальная степень, стандартное отклонение, параметр разнородности, максимальное сходство [162].

2 См. выдержки из диссертации немецкого учёного: Daniel Kinzler. “Outline of a method for building a multilingual thesaurus from Wikipedia”, 2008. http://brightbyte.de/page/WikiWord/Excerpt

3 См. http://www.opencyc.org.

4 См. http://www.cyc.com/cycdoc/ref/dict-assist.html.

5 DA предназначен для работы с английским языком. Поэтому, чтобы выполнить привязку русских слов к концептам Cyc, нужно каким-то образом «научить» DA выполнять лемматизацию русских слов и предоставить возможность пользователям указывать лексические свойства слов.

с помощью предикатов, например для перевода фразы «Fred fancies Sally» используется предикат *likesAsFriend*.

Авторы MSR веб-сервера[1] предлагают, наконец, объединить реализации различных подходов по вычислению семантической близости слов с целью обеспечить доступность к результатам работы и возможность сравнить их работу на одинаковых данных. Для стандартизации доступа к приложениям предлагается CGI[2] интерфейс.

1.3 Системы и способы графического представления тезаурусов и результатов поиска

Здесь перечислены современные системы графического представления тезаурусов и визуальные поисковые системы[3]. Дана их оценка с подчёркиванием тех сильных сторон, которые уже используются или могут быть внедрены в разрабатываемую систему.

Система Visual Thesaurus[4] предоставляет визуальный интерфейс к лексикону WordNet. Данная «программная адаптация интерфейса в динамическую структуру визуальных понятий в корне меняет и значительно интенсифицирует процесс обучения, освоения и применения данного продукта» [3]. Эстетически приятно оформлен ещё один визуальный онлайн WordNet интерфейс,[5] в котором слова равномерно распределены по кругу, а множество отношений рисуется в виде дуг окружностей.

В работах [3], [39], представлен метод формирования графовой структуры данных по текстовой информации и тезаурус предметной области в виде визуальной интерактивной среды[6]. Достоинством визуального представления является «возможность воспринимать содержимое текста не последовательно, а одномоментно. Это позволяет воспринимать структуру

1 MSR расшифровывается как мера семантической близости, см. сайт MSR http://cwl-projects.cogsci.rpi.edu/msr

2 CGI (Common Gateway Interface) интерфейс обеспечивает связь внешней программы с веб-сервером.

3 Визуальные поисковые системы обладают возможностью представлять результаты поиска визуально либо обеспечивают средствами для наглядного (интерактивного) формулирования самой задачи поиска.

4 См. http://www.visualthesaurus.com

5 См. http://www.ug.it.usyd.edu.au/~smer3502/assignment3/form.html

6 См. http://vslovar.org.ru

связей предметной области в комплексе, притом именно в том, который соответствует связям, сформированным специалистом..., а не формировать его самостоятельно при прочтении груды технической документации» [39]. Плюс интерактивной среды «Визуальный словарь» (СПИИРАН) в том, что данное приложение представлено в виде HTML страницы и для работы не требует от пользователя установки дополнительных приложений.

В [185] представлен алгоритм Roark и Charniak (и реализация с визуализацией результатов) для получения упорядоченного списка слов, принадлежащих указанной категории. Алгоритм включает шаги:

1. Для данной категории выбрать примеры (так называемые «*seed words»*, то есть начальные слова, «*затравка»* для алгоритма);
2. Подсчитать число сочетаний *seed* слов с другими словами корпуса;
3. Выбрать новые *seed* слова с помощью «*figure of merit*»[1];
4. Перейти к шагу 2, n итераций;
5. Упорядочить слова по степени принадлежности к категории (на основе «figure of merit») и выдать упорядоченный список.

Слабое место алгоритма в выборе начальных слов экспертом. Существует опасность «инфекции» – одно неправильно предложенное слово (не относящееся к категории) повлечёт за собой другие. Система, реализующая алгоритм, предназначена для автоматического создания словарей из текста на естественном языке. К достоинствам визуализации результатов можно отнести:

- *Web-интерфейс*. Следовательно, минимальные требования к клиенту – достаточно наличия интернет браузера;
- *Выбор корпуса текстов для поиска,* что даёт возможность тематического поиска.

Главный недостаток – это статическая картинка в качестве результата. Значит, для постановки новой задачи необходимо возвращаться на предыдущую страницу, перезагружать страницу (современные технологии,

1 «Similarity measure» в [185] – это и есть «figure of merit» (число совместных совпадений слов в списках).

например, AJAX[1], Piccollo[2], Flash позволяют решить эту проблему, см. пример визуального поисковика Kartoo[3]).

Облако тегов (tag cloud) – это ещё один способ формирования результатов поиска для одномоментного восприятия пользователем. В поисковой системе Ontos Semantic Web [124] используется подобное облако тегов. Другой пример – система Newzingo[4], автоматически сканирующая новости Google и представляющая их в виде облака новостей (Рис. 2).

В облаке тегов больший размер кегля (font-size) указывает на большую важность (популярность) слова. Чем больше найдено новостей, содержащих некоторое слово, тем слово будет больше. В данном случае (Рис. 2) популярными словами среди новостей технической тематики являются такие слова, как: *apple, google*.

Достоинство метода в том, что одним взглядом можно охватить все существенные события, все новостные сообщения. У облака новостей есть недостатки. Один из них – нет информации о динамике новостей, неясно какая тема набирает обороты, а к какой интерес угасает. В Newzingo новостная динамика представлена в виде списка новых тегов (recent tags). Другой недостаток облака новостей проявляется из-за того, некоторые темы (в силу исторических и других причин) обсуждаются регулярно (например: *cars, ibm, intel, apple*). Теги, соответствующие этим новостям, занимают место и заслоняют в облаке потенциально более интересные на сегодня темы.

1 См. http://ru.wikipedia.org/wiki/AJAX

2 См. http://www.cs.umd.edu/hcil/piccolo/index.shtml

3 Визуальный поисковик, использует Flash технологию. http://kartoo.com

4 См. http://newzingo.com

Рис. 2. Указание на важность слова за счёт большего размера кегля (Newzingo)

Альтернативой механизму тегов (folksonomy[1]) является поиск на основе пересечения данных, соотнесённых разным категориям, что представлено на отечественном сайте «Перекрестный каталог».[2] Категории, с одной стороны, являются альтернативой тегам, поскольку ресурс (интернет-страница, вики-страница, сайт, фотография) может иметь несколько тегов, несколько категорий (в Википедии). Можно выполнить операцию пересечения множеств и найти ресурсы, соответствующие нескольким тегам или категориям. С другой стороны, преимущество категорий в том, что они образуют иерархию. Пользователь может, отталкиваясь от данной категории, пойти к более общей либо к более частной категории.[3]

Действительно визуальным можно назвать приложение Flickr Graph[4] – визуализация социальной сети, поскольку для представления вершин графа используются картинки, а именно: фотографии участников проекта flickr (рис. 3). Приложение вычисляет положение вершин графа на основе классического алгоритма притяжения-отталкивания[5]. Идею вершин-картинок можно было бы использовать для визуализации отношений между страницами Википедии. В этом случае вершины графа, соответствующие

1 См. замечание с описанием folksonomy на стр. 183.

2 Сайт «Перекрестный каталог» предлагает несколько осей (схем) категоризации, выбирая значения которых, пользователь сужает пространство поиска. См. http://4kg.ru

3 Авторы систем, использующих закладки, предлагают различные способы организации закладок, что сближает подход тегов с подходом категоризации страниц. Например, на del.icio.us предлагается механизм «bundle tags» для организации иерархии закладок.

4 См. http://www.marumushi.com/apps/flickrgraph

5 См. http://en.wikipedia.org/wiki/Force-based_algorithms

страницам Википедии, будут представлены с помощью thumb картинок[1]. За пользователем можно оставить право выбора графического или текстового представления вершин графа, например, как на сайте Функциональной Визуализации (http://www.visualcomplexity.com/vc).

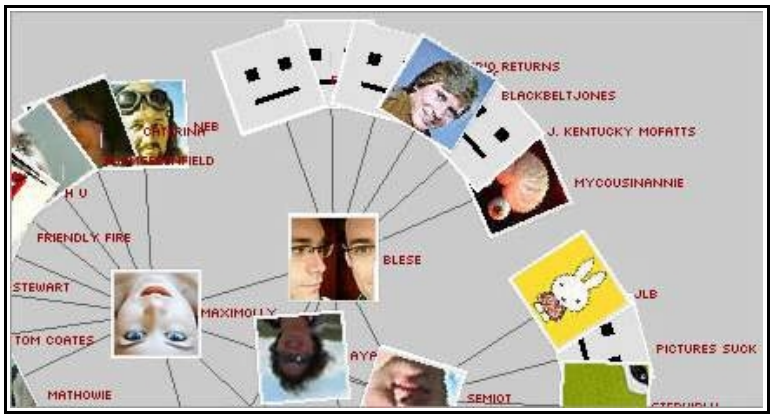


Рис. 3. Визуализация социальной сети в приложении Flickr Graph

Для анализа социальных сетей, получившихся в ходе работы в Википедии предназначено Java-приложение Sonivis [140].

Два следующих приложения не выполняют никакого поиска. Их задача – это анализ и графическое представление истории работы пользователя с вики сайтом.

В приложении Rhizome Navigation[2] анализируется история работы пользователя с вики страницами. На рис. 4 представлены вики страницы (закрашенные прямоугольники), которые посетил пользователь, и ссылки между страницами (линии). Чем дольше пользователь оставался на странице, тем больше будет по размеру соответствующий прямоугольник. Часто используемые ссылки отображаются более толстыми и короткими линиями.

---

1 «Многие статьи Википедии содержат один рисунок (ознакомительный), иллюстрирующий главную мысль статьи. В этом качестве для биографической статьи используется портрет, для статьи о бытовой технике — фотография предмета статьи, для статьи об общественном движении — его символ или флаг, и так далее. Рекомендуется именно этим изображением и начинать статью. Такое изображение должно обязательно иметь атрибут *thumb*» (http://ru.wikipedia.org/wiki/Википедия:Изображения).

2 См. http://www.metaportaldermedienpolemik.net/wiki/Blog/2006-04-18/RhNav3D

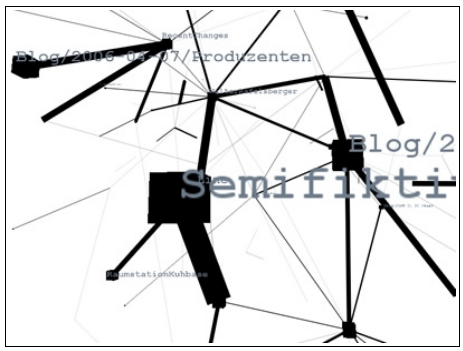


Рис. 4. Визуальное представление просмотренных пользователем вики страниц в приложении Rhizome Navigation

Приложение Pathway позволяет «не потеряться в бесчисленных перекрёстных ссылках Википедии»[1]. Приложение визуально представляет вики страницы, посещённые пользователем, в виде сети: вершина сети – это статья, ребро – это ссылка, по которой пользователь перешёл от одной страницы к другой (рис. 5). Эту сеть можно сохранять на диск и загружать.

Ещё два приложения: WikiViz и ClusterBall, написанные одним автором, Крисом Харрисоном, позволяют визуально представлять данные Википедии. В программе WikiViz[2] одновременно отображается значительная часть Википедии (десятки тысяч страниц и связей между ними), однако за счёт потери интерактивности. Первоначальный учёт ссылок между статьями и категориями приводил к тому, что все вершины сливались в единое целое, в один кластер, поэтому в WikiViz учитываются только ссылки, указанные в тексте страниц.

В приложении ClusterBall[3] визуализируется структура трёх уровней категорий Википедии. В центре графа отображается родительская вершина. Статьи, на которые ссылается родительская вершина, рисуются внутри шара. И, наконец, статьи, ссылающиеся на эти (внутренние) вершины отображаются во внешнем кольце. Полученные рисунки ничего не говорят о сути данных, то есть о том, какие статьи представлены, однако построенные кластеры позволяют косвенно судить о способе организации информации в Википедии, позволяют сравнить структуру Википедии с системами

1 Программа Pathway с открытым исходным кодом написана на языке Cocoa для компьютеров Макинтош. См. http://pathway.screenager.be

2 См. http://www.chrisharrison.net/projects/wikiviz/index.html

3 См. видеофильмы и рисунки на странице проекта http://www.chrisharrison.net/projects/clusterball

MVblogosphere[1], 6Bone IPv6[2], и Gnom[3], поскольку для них получены аналогичные рисунки.

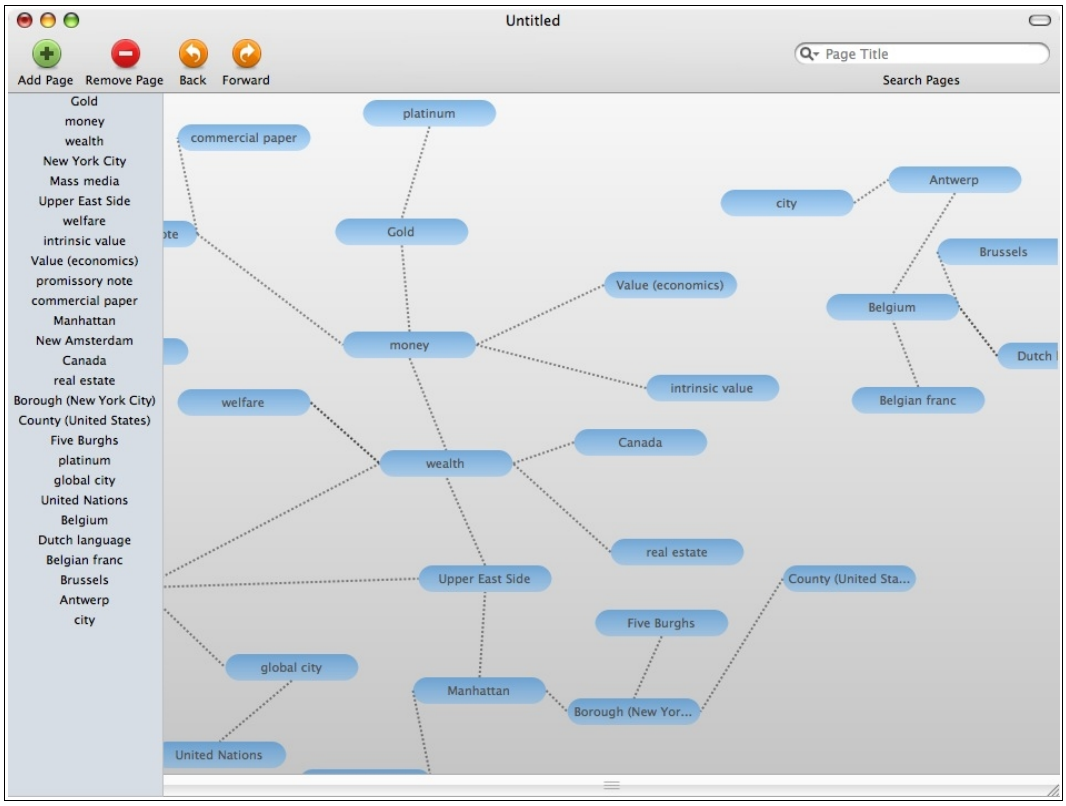


Рис. 5. История просмотра страниц Википедии в приложении Pathway

1.4 Постановка задачи исследования

На первом этапе исследований проведён анализ методов поиска синонимов и методов поиска похожих документов (интернет страниц, статей энциклопедии с гиперссылками и т.п.). Необходимо выполнить ряд подзадач для решения общей задачи автоматизации построения списков семантически близких слов. Был обоснован выбор HITS алгоритма для поиска. Далее необходимо, во-первых, адаптировать HITS алгоритм к поиску наиболее похожих документов в проблемно-ориентированном корпусе текстов с гиперссылками и категориями.

Во-вторых, нужно реализовать предложенный алгоритм поиска семантически близких слов (в том числе синонимов) в виде программы (с визуализацией результатов поиска с возможностями интерактивного

1 См. http://www.mvblogs.org/visuals/visual_08.php

2 См. http://www.visualcomplexity.com/vc/project_details.cfm?id=142&index=142&domain=

3 См. http://www.visualcomplexity.com/vc/project_details.cfm?id=55&index=55&domain=

поиска) для последующей экспериментальной проверки работоспособности и алгоритма и программы.

В-третьих, необходимо спроектировать архитектуру программной системы оценивания и разработать способы численной оценки набора синонимов. Способы численной оценки набора синонимов необходимы для проведения экспериментальной части работы. Проектирование архитектуры программной системы оценивания – это задел на будущее, для более всесторонней оценки работы алгоритма поиска.

Одно из приложений HITS алгоритма, используемое в данной работе – это вычисление меры сходства вершин графа. Поэтому, в-четвёртых, для полноценной оценки HITS алгоритма предлагается разработать альтернативный алгоритм вычисления меры сходства вершин графа. Реализация такого алгоритма и собственно сравнение являются частью будущих исследований и не будут представлены в данной работе.

Было указано выше на необходимость морфологической обработки текстов в задачах автоматической обработки текстов на естественном языке (такая обработка необходима в том числе и для автоматизированного построения списков семантически близких слов). Также была выбрана программная среда GATE для обработки текстов на естественном языке. Таким образом, пятая подзадача – эта разработка архитектуры и реализация программного модуля системы GATE для морфологического анализа текстов на русском языке.

Выводы по главе 1

Проведённый анализ проблемы автоматизированного построения списков семантически близких слов показал, что здесь можно выделить такие основные подзадачи, как: (1) выбор текстового ресурса для поиска слов; (2) выбор и адаптация одного из перечисленных выше алгоритмов для данного текстового ресурса; (3) решение задачи оценки алгоритма и выбор текстового ресурса для оценки результатов работы алгоритма; (4) метод визуализации результатов поиска.

Рассмотренные алгоритмы поиска похожих страниц, поиска семантически близких слов в значительной степени зависят от структуры документов, от корпуса текстов в пространстве которого выполняется поиск. Среди множества проблем создания корпуса, можно выделить общую проблему отсутствия единого стандарта. В диссертации в качестве корпуса текстов предлагается использовать коллективную онлайн энциклопедию Википедия. Это позволяет решить в какой-то мере проблему стандарта (все статьи унифицированы, а именно есть стандартные метаданные: заголовок статьи, категории, определяющие тематику статьи). Если подытожить причины использования Википедии как текстового ресурса, то получим – стандартизация, наличие программного обеспечения, классификация текстов, большое количество статей и общедоступность.

Недостатками алгоритмов (HITS и PageRank) является то, что они используют только структуру ссылок и не учитывают существующую классификацию текстов (например в корпусе текстов Википедия). Заметим, что необходимой частью алгоритмов является возможность численной оценки качества полученных результатов.

Анализ поисковых систем показал, что многие из них обеспечивают визуализацию результатов поиска. Автором выделены системы, представляющие результаты поиска в виде статической и динамические картинки. В данной работе предложена динамическая интерактивная визуализация результатов, обеспечивающая большую наглядность и удобство при поиске данных.

Таким образом, основными проблемами автоматизированного построения списков семантически близких слов можно считать (i) проблему построения *алгоритма поиска*, который, с одной стороны, *адаптирован* к существующему текстовому ресурсу, с другой стороны, результат его работы может быть однозначно *оценен*, и (ii) проблема *визуализации* результатов поиска. На решение этих проблем и направлена данная диссертационная работа.

2. Методологическое и математическое обеспечение для построения списков семантически близких слов в корпусе текстовых документов с гиперссылками и категориями

В этой главе описаны подход, основные алгоритмы, разработанные в рамках диссертационной работы: адаптированный HITS алгоритм с использованием алгоритма кластеризации (поиск семантически близких слов в корпусе текстовых документов с гиперссылками и категориями) и алгоритм вычисления меры сходства вершин графа (поиск похожих вершин графа). Представлены оценки временной сложности алгоритмов и эвристики. Предлагается несколько показателей численной оценки полученного набора семантически близких слов с помощью тезаурусов WordNet и Moby.

2.1 Подход к поиску семантически близких слов

Предлагаемый подход к поиску семантически близких слов (СБС) представлен на рис. 6. Входными данными для поиска семантически близких слов являются исходное слово, корпус документов[1] и список слов, уточнённый пользователем[2]. Последовательность взаимодействия частей системы указана на рис. 6 числами (1-7). Входными данными для поискового алгоритма являются слово, заданное пользователем (1-2), и корпус текстов (2). Алгоритм строит упорядоченный список СБС (3), а пользователь получает возможность работать с ним благодаря визуализации (4-5). В ходе работы пользователь уточняет список СБС и может запустить алгоритм повторно (6-7). Достоинствами данного подхода являются (1) визуализация

1 Из корпуса выбирается документ, заголовок которого совпадает с исходным словом, заданным пользователем. Таким образом, в данной схеме приняты следующие допущения: (1) разные документы в корпусе имеют разные заголовки, (2) заголовки состоят из одного слова. Заметим, что можно расширить эти ограничения, если (1) вместо заголовков выполнять поиск по ключевым слова документов, (2) выполнять поиск подстроки в заголовке (тогда заголовок может состоять из нескольких слов). Заметим также, если слово задано не в неопределённой форме, то его можно привести к ней с помощью модуля морфологической обработки русского языка Lemmatizer, описанного на стр. 106. Прочие ограничения для заголовков словарных статей указаны на стр. 74.

2 Список слов, найденный системой и уточнённый пользователем, то есть предполагается обратная связь для уточнения результатов поиска.

результатов поиска, (2) возможность уточнения запроса в ходе работы пользователя с программой.

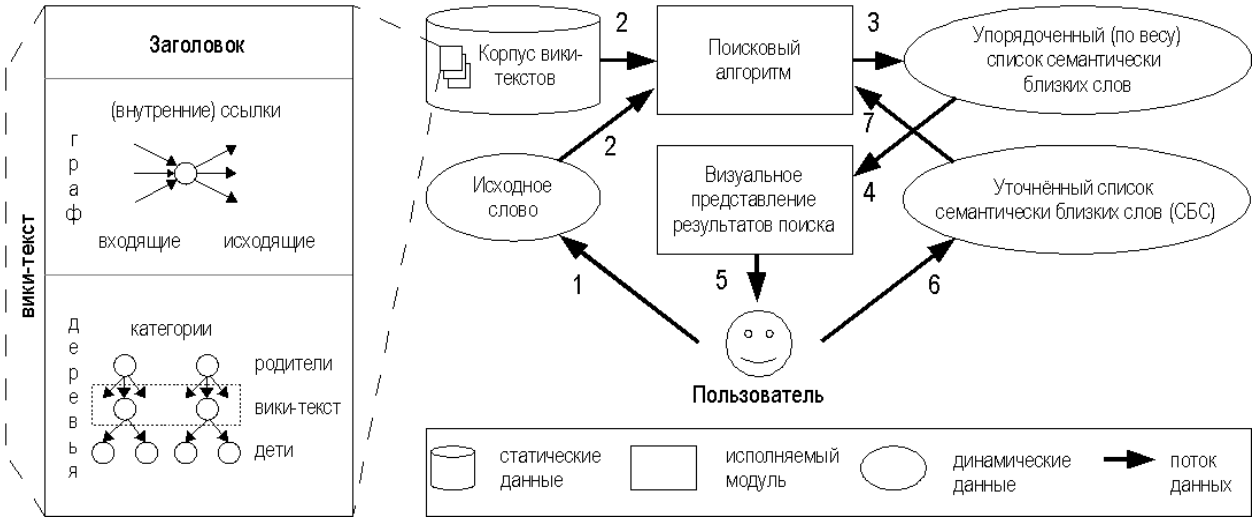


Рис. 6. Подход к поиску семантически близких слов

Перечисленные выше требования[1] к корпусу проблемно-ориентированных текстов задают два типа документов с гиперссылками (статья и категория) и три вида отношений, то есть ссылок между документами в корпусе (статья-статья, статья-категория и категория-категория). Отношения между категориями являются иерархическими (родо-видовые и часть – целое).[2]

Авторы алгоритма извлечения синонимов из толкового словаря [84], [83] предложили гипотезу, по которой два слова считаются «почти синонимами» (near synonyms), если слова:

1. употребляются в определении одного и того же слова;
2. имеют общие слова в определениях.

Таким образом, в работах [84], [83] полагают, что если статьи толкового словаря оказались семантически близкими (то есть выполняются два пункта, указанных выше), то и заголовки статей (слова, словосочетания) будут семантически близкими. Назовём это *гипотезой 1*.

Обобщение этой гипотезы на документы с гиперссылками предложил Kleinberg [125], используя понятия авторитетных и хаб-страниц[3]. Обобщение

1 См. Требования к корпусу проблемно-ориентированных текстов, гл. 1, стр. 24.

2 См. «Виды отношений в Википедии» в табл. 1 на стр. 184.

3 См. определения авторитетных и хаб-страниц далее в главе 1 в подразделе «Алгоритм HITS».

таково: пусть на страницу *p* ссылаются хаб страницы, которые могут иметь ссылки на авторитетные страницы.

На страницу *p* будут *похожими* (similar у Kleinberg'a, семантически близкие в нашей терминологии) те из авторитетных страниц, на которые ссылаются те же хаб-страницы, которые, в свою очередь, ссылаются на страницу *p*. Назовём это *гипотезой 2*.

Использование этих двух гипотез позволит нам (1) использовать гиперссылки для поиска похожих документов, (2) считать ключевые слова (например, заголовки документа) семантически близкими для похожих документов.

Таким образом исходными данными для поисковых алгоритмов будут направленный граф (вершины – это статьи, дуги – это ссылки) и дерево категорий (вершины – категории, дуги связывают категории-родителей и категории-детей). Направленный граф и дерево связаны, поскольку вершины направленного графа (статьи) связаны с одной или несколькими вершинами дерева категорий.

Общая идея адаптированного алгоритма[1] заключается в следующем. Документы корпуса связаны друг с другом ссылками[2] наподобие Интернет страниц. У каждого документа есть ключевые слова, его характеризующие и ему соответствующие; в простейшем случае – это заголовок документа. Если, таким образом, для документа найдены (с помощью HITS алгоритма) похожие документы, то можно утверждать, что найдены похожие заголовки, являющиеся семантическими близкими словами для заголовка исходного документа[3]. Остаётся улучшить HITS алгоритм за счёт особенностей рассматриваемого корпуса текстов[4].

---

1 Исходный HITS алгоритм представлен в главе 1 в подразделе «Алгоритм HITS» (стр. 27). Отметим работу [125], в которой предлагается искать похожие Интернет страницы на основе структуры ссылок.

2 Предлагаемый адаптированный HITS алгоритм применим только к корпусам текстам, удовлетворяющим ряду требований, см. «Требования к корпусу проблемно-ориентированных текстов» в главе 1 на стр. 24.

3 Здесь и далее под *статьёй*, *страницей*, *документом, вики-текстом* подразумевается текстовый документ корпуса. Разница между страницей, статьёй и категорией объясняется в подразделе «Замечания о категориях и ссылках Википедии» на стр. 186.

4 Под особенностями корпуса подразумевается (1) наличие категорий (категории классифицируют документы, то есть определяют их тематическую принадлежность; сами категории в идеале образуют иерархическую структуру) и (2) наличие двух списков страниц для каждой статьи (кто ссылается на

Причиной использования категорий для установления связи между страницами (относительно оригинального алгоритма HITS, который оперирует только ссылками) является следующее: «входящие ссылки» (in-links), то есть ссылки на данную страницу, «*содержат ссылки, связь которых с основной страницей может быть весьма слаба, в то время как в одну категорию обычно помещают страницы сходной тематики*»[1].

2.2 *HITS* алгоритм (формализация, анализ, поиск синонимов)

Формализация задачи

Дан направленный граф $G=(V, E)$, где V – вершины (Интернет страницы), E – дуги (ссылки). Для каждой страницы v известны два списка: $\Gamma^{+}(v)$ – это страницы, на которые ссылается данная статья, и $\Gamma^{-}(v)$ – это страницы, ссылающиеся на данную статью. Необходимо найти набор страниц, соответствующих запросу s (*релевантность*), на которые при этом ссылаются многие страницы (*авторитетность*).

Обработка всех страниц, содержащих текст запроса, является вычислительно дорогой процедурой и общее число таких страниц, вероятно, избыточно (если представить их пользователю, сформулировавшему запрос). Требуется идеальная *коллекция* (множество страниц) S_σ: такая, что она *небольшая* (подходит для вычислительно «тяжёлых» алгоритмов), содержит много *релевантных* страниц, большинство страниц являются *авторитетными* (в идеале — все).

Клейнберг [125] предлагает следующий алгоритм для поиска M страниц максимально похожих (подразумевается максимальное значение веса страницы – *authority*) на заданную страницу v:

1. Поиск с помощью общедоступного интернет поисковика (например, AltaVista) релевантных страниц – это корневой набор S_σ (root set).
2. Расширение корневого набора (с целью включения авторитетных страниц в набор) страницами, которые связаны

данную статью Г⁻(v), на кого ссылается данная статья Г+(v)).

1 Цит. по http://ru.wikipedia.org/wiki/Википедия:Категории.

ссылками с корневым набором – это базовый (base) набор страниц (рис. 7).

3. Итеративное вычисление весов (*authority* и *hub*) у вершин базового набора для поиска авторитетных и хаб-страниц. Завершение итераций, когда суммарное изменение весов меньше наперёд заданного ε.

4. Выбрать *M* страниц с максимальным значением веса *authority*) из базового набора.

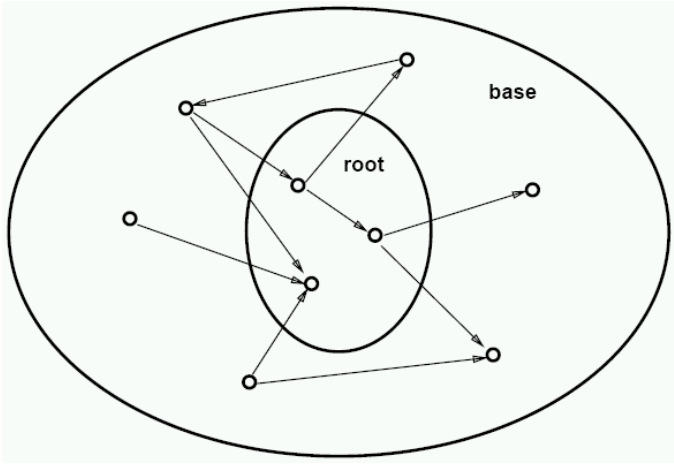


Рис. 7. Расширение корневого набора страниц (root) до базового (base) [125]

Каждому документу в HITS алгоритме сопоставляются веса *a* (*authority*) и *h* (*hub*), которые показывают, соответственно, насколько документ является авторитетным и насколько он является хорошим хаб-документом. Формулы алгоритма HITS для итеративного вычисления весов таковы:

$$h_j = \sum_{i:(j,i)\in E} a_i , \qquad (2.1)$$

$$a_j = \sum_{i:(i,j)\in E} h_i \qquad (2.2)$$

где h_j и a_j показывают соответственно насколько документ j (1) является хорошим указателем на релевантные документы (то есть j-ый документ рассматривается как хаб-документ) и (2) является авторитетным документом. Доказательство сходимости итеративного процесса вычисления весов h_j и a_j, использующее собственные значения матрицы смежности графа *G,* представлено в [125] и [81].

Детальное описание HITS алгоритма представлено в [125]. Описание изменений в алгоритме для адаптации к поиску в вики ресурсах представлено ниже[1].

Дополнительные замечания

Приведём *положения Клейнберга* [125] и свои замечания к ним:

1) *Авторитетные страницы должны быть авторитетными именно для данного запроса.*

Изменение авторитетности (веса страницы) в зависимости от запроса приводит к тому, что вычисления выполняются в режиме онлайн, то есть после формулирования запроса. Остаётся открытым вопрос о возможности предварительной предобработки, позволяющей ускорить поиск авторитетных и хаб-страниц.

2) *Нет такого (внутреннего) свойства страницы, по которому можно было бы определить её авторитетность.*

Иначе любая страница (усилиями авторов) стала бы авторитетной. Поскольку быть авторитетной страницей значит привлекать бóльшее число пользователей, и значит быть более доходной для коммерческих сайтов.

3) Проблема: страница может не содержать слова, которым соответствует её содержание. Например страница поисковика может не содержать слова "search".

Анализ структуры ссылок (HITS алгоритм) решает эту проблему, поскольку (например, для данного примера со словом "*search"*) найдутся страницы, которые содержат искомое слово и ссылаются на релевантную страницу (в данном случае на поисковик).

Глобальность HITS алгоритма в том, что ставится задача поиска лучших авторитетных страниц во всём Интернете. Отметим, что анализ ссылок (здесь) отличается от кластерного анализа, так как элементы/кластеры образуют единое целое, чего не скажешь о страницах Интернет в целом.

Проблемы HITS алгоритма [81], [89]:

1 См. подраздел 2.3 «Адаптированный HITS алгоритм, включающий алгоритм иерархической кластеризации» на стр. 76.

- *учитываются «только ссылки»*, что приводит к взаимному усилению веса страниц, ссылающихся друг на друга;[1]
- *существуют «автоматически создаваемые ссылки»*, то есть ссылки, созданные программой, не выражают мнение человека о значимости страницы;[2]
- *«не релевантные документы»*, граф содержит документы, не соответствующие теме запроса. Возникает проблема смещения темы (*topic drift*) из-за того, что большой вес имеют вершины-документы о популярных предметах (то есть имеющих много входящих ссылок), но таких, что уводят в сторону от исходной темы;[3]
- *связность*, если веб-граф представляет несколько несвязанных компонент, то HITS присвоит нулевые веса *hub* и *authority* всем страницам, кроме главной компоненты.

В [81] предложено расширение HITS алгоритма за счёт анализа текста документов, что в результате повысило точность поиска, особенно для редких запросов. Вес вершины вычисляется с помощью TF-IDF схемы[4] (сравниваются первые 1000 слов документа и слова запроса). Перемножаются значение веса *authority* и вес вершины (вычисленный по схеме TF-IDF), – теперь это будет новое значение *authority*, аналогично пересчитывается вес *hub*. При построении по запросу подграфа, используется

1 Этот случай относится к мультиграфу (например, Веб), содержащем множество дуг от одной вершины к другой. Такой проблемы нет в вики-текстах, поскольку вики в реализации *MediaWiki* не мультиграф: между двумя страницами либо есть *одна* направленная ссылка, либо нет (см. таблицу *pagelinks* в БД).

2 Таким образом, нарушается основной принцип, на который полагаются такие алгоритмы, как: HITS и PageRank. К счастью эта проблема не существенна при поиске в вики-ресурсах, поскольку вики (и ссылки в них) по определению (см. стр. 51), создаются людьми, а не программой.

3 На эту же проблему указывает Hearst M. [112] (цит. по [81]), а именно: HITS алгоритм не находит часть документов, соответствующую менее популярной интерпретации запроса. Hearst предлагает возможные решения: (1) кластеризация документов на подтемы и анализ каждого подграфа с помощью HITS отдельно; (2) модификация HITS алгоритма, присваивающая рёбрам внутри кластера больший вес по сравнению с весом рёбер между кластерами.

4 «IDF (обратная частота термина в документах, обратная документная частота) – показатель поисковой ценности слова (его различительной силы)» [52]. В 1972 г. Karen Sparck Jones предложил эвристику: «терм запроса, встречающийся в большом количестве документов, обладает слабой различительной силой (широкий термин), ему должен быть присвоен меньший вес, по сравнению с термином, редко встречающимся в документах коллекции (узкий термин)». Данная эвристика показала свою пользу на практике и в работе [155] представлено её теоретическое обоснование.

вес вершины, чтобы решить – оставить вершину в подграфе или удалить в том случае, когда значение веса ниже некоторой границы (приводится 3 способа вычисления границы).

Тематическая связность авторитетных страниц

Задача – извлечь тематически связанные авторитетные страницы из коллекции страниц S_σ (см. выше). Простейший подход: упорядочить страницы по степени захода (число ссылок на страницу). Проблема такой простой схемы ранжирования в том, что могут быть найдены страницы с большой степенью захода, но тематически несвязанные.

Возможны следующие решения задачи:

1) *Решение Клейнберга*. Авторитетные страницы (по данной тематике) содержат не только большое число входных ссылок, но и пересекаются между собой. Это обеспечивается хаб-страницами, содержащими ссылки сразу на несколько авторитетных страниц (одной тематики).

2) *Оригинальное решение* (на основе решения Клейнберга). В Википедии каждой странице эксперты приписывают несколько категорий. Категории образуют дерево, то есть у каждой категории есть категория-родитель и несколько детей. Такая тематическая определённость страниц позволяет:

a) найденный список синонимов (с помощью адаптированного HITS алгоритма) разбить на кластеры, каждый из которых соответствует одному из значений исходного слова либо

b) применить кластеризацию коллекции страниц для последующего применения HITS алгоритма к каждому кластеру отдельно.

Применение способов оценки результатов поиска в Интернет к HITS алгоритму

Поиск в Интернет (Web search) – это нахождение релевантных страниц, соответствующих запросу. Оценка качества поиска основана на оценке, сделанной человеком, так как релевантность – субъективное свойство. Не хватает объективной (численной) функции для определения качества поиска.

В классических информационно-поисковых системах (ИПС) результаты ранжируются (i) пропорционально частоте термов в документе, (ii) обратно пропорционально частоте терма по всем документам (относительно ключевых слов запроса) (TF-IDF схема).

При Веб поиске дополнительно учитывается множество параметров [76]: число ссылок, которые ссылаются на страницу; текст ссылки; положение искомого слова (наличие его в заголовке даёт больший вес странице); расстояние между искомыми термами на странице; популярность страницы (число посещений); содержание текста в метаданных (metatags) [143], [93], [113]; принадлежность страницы определённой тематике; новизна ресурса; степень точности соответствия запросу.

В HITS алгоритме вес страницы распределяется равномерно между страницами, на которые она ссылается (см. формулы (2.1) и (2.2) для вычисления весов hub и authority). Предлагается учитывать положение ссылки на странице таким образом: чем выше на странице упоминается ссылка, тем больший вес она получит.

Поиск синонимов с помощью HITS алгоритма

В работах [84], [83] предложена адаптация HITS алгоритма к поиску синонимов в словаре Webster[1]. Словарная статья состоит из заголовка и текста статьи. Словарные статьи ссылаются друг на друга (образуют граф) посредством упоминания слов, которые являются заголовками других статей. С помощью алгоритма HITS можно найти авторитетные[2] словарные статьи для исходной статьи. Авторы предполагают, что заголовки этих словарных статей будут содержать синонимы относительно заголовка исходной статьи. При построении графа словаря часть статей словаря была отфильтрована, поскольку:

- не учитываются составные слова, то есть из нескольких термов (например, *All Saints'*, *Surinam toad*);

1 Поиск выполнялся в словаре «*Online Plain Text English Dictionary*» (http://msowww.anu.edu.au/~ralph/OPTED), построенный на основе словаря «*Project Gutenberg Etext of Webster's Unabridged Dictionary*» (http://www.gutenberg.net), который, в свою очередь, использует словарь «*1913 US Webster's Unabridged Dictionary*» [84].

2 Определения авторитетных и хаб-страниц см. выше в подразделе «Алгоритм HITS».

- многозначные слова рассматриваются как однозначные;
- исключили статьи, состоящие из чисел и формул (мат., хим.);
- исключили часто встречаемые слова, которые встречаются чаще чем в 1000 определений;
- рассматривались слова только той же части речи, что искомое.

Далее вычислялись авторитетные и хаб-вершины. Проводилась оценка полученных синонимов для четырёх слов разных типов [83]:

1. Слово, имеющее несколько синонимов (*Dissappear*);
2. Терминологическое, узкоспециальное слово (*Parallelogram*), в действительности не имеющее синонимов, однако существуют слова для названий предметов близких по сути (quadrilateral, square, rectangle, rhomb).
3. Многозначное слово (*Sugar*);
4. Общеупотребительное слово, обозначающее понятие, не имеющее однозначного определения (*Science*).

Авторы [83] сравнили свой метод (адаптация HITS к поиску слов в словарных статьях) с методом расстояний и методом PageRank, вычисляя синонимы для вышеуказанных четырёх слов. Их метод показал показал более хорошие результаты, чем эти два метода.

Суть *метода расстояний* (distance method) заключается в том, что два слова считаются синонимами, если:

- употребляются в определении одного и того же слова;
- имеют общие слова в своих определениях.

В работе [83] предложили такую формализацию метода расстояний. Расстояние между двумя словами равно сумме количества слов в определении первого слова, отсутствующих в определении второго, и количества слов в определении второго слова, отсутствующих в определении первого.

2.3 Адаптированный *HITS* алгоритм, включающий алгоритм иерархической кластеризации

Постановка задачи. Дан направленный граф *G=(V, E)*, где *V* – вершины (текстовые документы), *E* – дуги (ссылки между документами). Для каждой вершины определены значения двух весовых коэффициентов *authority* и *hub*: $\{v \in V : a_v \in \mathbb{R}, h_v \in \mathbb{R}\}$. Для каждой страницы *v* известны два списка: $\Gamma^+(v)$ – это страницы, на которые ссылается данная статья, и $\Gamma^-(v)$ – это страницы, ссылающиеся на данную статью. Задана исходная страница *s*. Нужно найти похожие страницы.

Необходимо найти набор вершин *A*, *похожих* на вершину *s*. Похожесть определяется тем, что есть много хаб-вершин *H*, указывающих и на *s*, и на вершины из *A*. При этом вершины *A* являются авторитетными, то есть на них указывают многие вершины той же тематической направленности, что и исходная вершина *s*. Формализуем понятия похожести и авторитетности вершин, то есть формализуем постановку задачи.

Формализация понятия «похожие вершины» графа

Необходимо найти множество вершин *A*, которые являются (i) *авторитетными* вершинами для исходной вершины *s* (то есть значение (2.3) больше относительно других подмножеств вершин той же мощности), (ii) *похожими* на исходную вершину *s*, то есть существует множество хаб-вершин *H*, ссылающихся одновременно и на исходную вершину *s* и на вершины из *A* (2.4), (iii) где *H* – это хаб-вершины (то есть значение (2.5) относительно других подмножеств вершин той же мощности). Задача выбрать множество *A* авторитетных вершин и множество *H* хаб-вершин в значении (2.6), где *k* — весовой параметр.

$$A \subset V, |A| = N, \sum_{v \in A} a_v \to max \tag{2.3}$$

$$A \subset V, H \subset V, \forall a \in A \;\; \exists h \in H : \Gamma^+(h) \ni s, a \tag{2.4}$$

$$H \subset V, |H| = M, \sum_{v \in H} h_v \to max \tag{2.5}$$

$$A \subset V, H \subset V, k \in [0,1]: \quad k \cdot \sum_{v \in A} a_v + (1-k) \cdot \sum_{v \in H} h_v \to max \tag{2.6}$$

Адаптированный HITS алгоритм

Входные параметры алгоритма: $s \in V$; $t, d, N, C_{max} \in \mathbb{N}$; $\varepsilon \in \mathbb{R}$, где

- s – исходная вершина графа (исходный документ), требуется найти вершину похожую на исходную;
- t – размер корневого набора R_S (число страниц, включаемых в корневой набор);
- d – инкремент, параметр определяет размер базового набора B_s (d входящих ссылок для каждого документа в корневом наборе будет добавлено в базовый набор, более подробно см. [125]);
- N – число похожих вершин (документов, или семантически близких слов), которые необходимо найти;
- C_{max} – максимально допустимый вес кластера (число статей и категорий в кластере), где под кластером понимается набор статей, связанный ссылками с другими статьями и с категориями;
- ε – допустимая погрешность для останова итераций.

Шаги адаптированного HITS алгоритма представлены в виде блок-схемы на рис. 8. Шаги, предложенные автором, выделены пунктиром. Опишем более подробно шаги 2-6 адаптированного HITS алгоритма:

2. В корневой набор (root) включаются те страницы, на которые ссылается исходная страница; может быть включено не более t страниц (*вместо поиска с помощью поискового сервера – в HITS алгоритме*).

3. Для каждого страницы в корневом наборе: включаем все страницы, связанные исходящими ссылками и не более d страниц, связанных входящими ссылками. Так строим базовый набор страниц (Рис. 9).

4. Итеративно вычисляются веса с помощью формул Клейнберга (2.1) и (2.2) для поиска авторитетных и хаб-страниц среди страниц базового набора. Итеративные вычисления завершаются, когда суммарное изменение весов по всем вершинам a_v и h_v за одну итерацию меньше ε.

5. Алгоритм иерархической кластеризации применяется к базовому набору страниц для получения групп страниц, соответствующих разным тематическим направлениям (*шаг отсутствует в HITS алгоритме*). Алгоритм кластеризации учитывает веса страниц (полученные на предыдущем шаге) и ссылки между страницами (статьями и категориями).

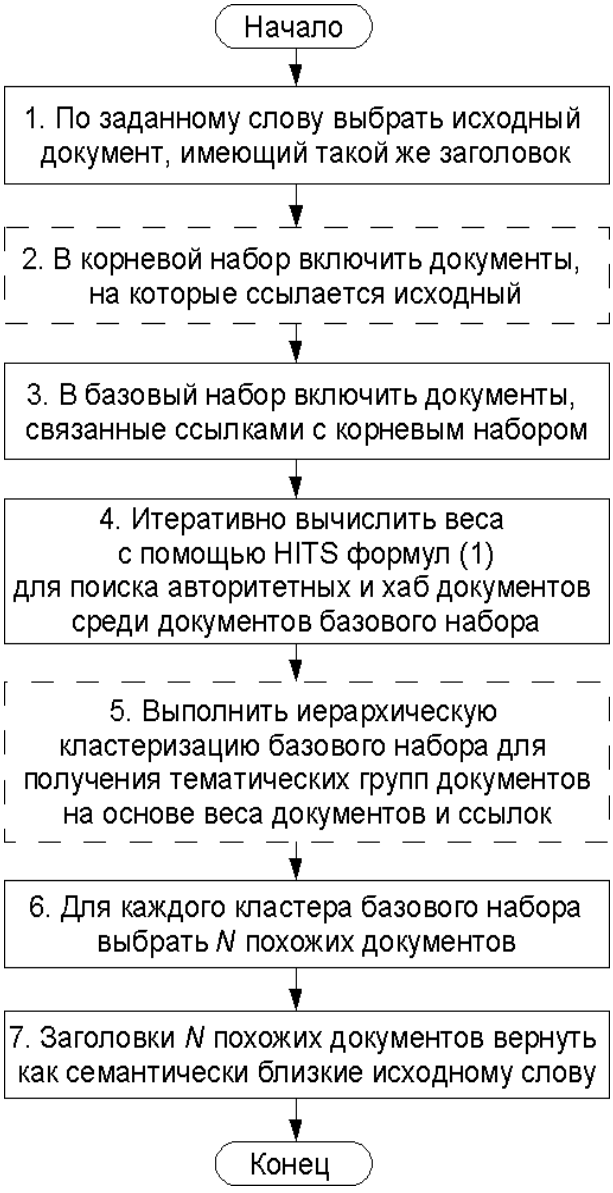


Рис. 8. Адаптированный HITS алгоритм

6. Для каждого кластера базового набора выбирается множество страниц A, содержащих N статей. Причём страницы

множества A являются (i) *авторитетными* для исходной страницы s (то есть суммарное значение весов authority страниц из A велико относительно других подмножеств страниц той же мощности), (ii) *похожими* на исходную вершину s в смысле наличия множества хаб-страниц H, ссылающихся одновременно и на исходную вершину s и на вершины из A[1].

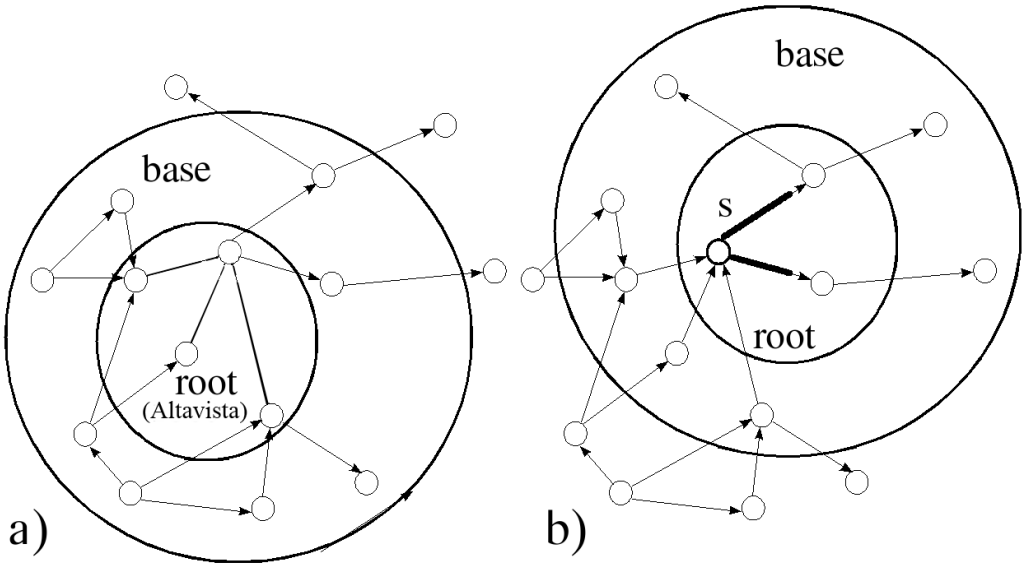


Рис. 9. Построение базового набора страниц (a) в HITS и (b) в адаптированном HITS алгоритме

Априори трудно сказать, чем лучше способ (a) или (b) при построении базового набора страниц (рис. 9). Основная задача при построении базового набора – включить как можно больше как можно более авторитетных страниц из пространства Интернет (в случае (a)), из пространства корпуса (в случае b). Оба этих подхода не могут гарантировать, что такое включение произойдёт[2].

Построение базового набора страниц

Адаптированный алгоритм построения базового набора страниц (шаги 1 и 2 адаптированного HITS алгоритма):

1 См. формулы (2.3)-(2.6) выше.

2 Одна из задач будущих исследований – настроить локальный поисковик (например, Google Desktop или Beagle) на корпус текстов, с которым работают в адаптированном HITS алгоритме. Тогда, можно будет сравнить два способа построения базового набора: (1) построение корневого набора с помощью поисковика и (2) построение корневого набора от исходной вершины (текущий вариант).

Subgraph(s,t,d)

$t, d \in \mathbb{N}$, $s \in V$.

s: исходная страница.

t: число страниц в корневом наборе R_s.

d: инкремент – число страниц, добавляемых в базовый набор B_s.

Обозначим

R_s – корневой набор из t страниц для страницы s;

B_s – базовый набор страниц.

R_s := первые t ссылок страницы s.

For each p $\in$ Rs

$\Gamma^+(p)$ – это набор всех страниц, на которые указывает p.

$\Gamma^-(p)$ – это набор всех страниц, которые ссылаются на p.

B_s += $\Gamma^+(p)$.

If $|\Gamma^-(p)| <= d$ then

Bs += $\Gamma^-(p)$.

Else

Bs += (d любых страниц из $\Gamma^-(p)$).

End

Return B_s

Число добавляемых страниц из $\Gamma^-(p)$ ограничено параметром *d*, так как на некоторые страницы могут указывать сотни и тысячи страниц.

Вычисление весов authority и hub

После построения базового набора страниц применяется итеративный метод вычисления весов, также как и в HITS алгоритме. Каждой странице присваивается две величины: authority и hub, которые вычисляются методом Iterate(ε) (шаг 3 алгоритма).

Iterate(ε, N):

$\varepsilon \in \mathbb{R}$, $N \in \mathbb{N}$, где

ε – допустимая погрешность для останова итераций.

N – число похожих страниц (авторитетных), которые необходимо найти

while (E > ε) {

Для каждой страницы из базового набора B_s :

обновить значения: authority, hub (см. формулы (2.1) и (2.2))

Нормализовать authority и hub так, что $\sum_{v \in V} a_v = 1$ и $\sum_{v \in V} h_v = 1$

Вычислить суммарное изменение веса по всем страницам:

$$E = \sum_{v \in V} \left(\left| h_v^{new} - h_v^{old} \right| + \left| a_v^{new} - a_v^{old} \right| \right)$$

}

Return N страниц с максимальным значением веса authority.

Временная сложность[1] вычисления весов (в зависимости от N – числа страниц в базовом наборе) составляет $O(I \cdot N^2)$, где I – число итераций. Поскольку для каждой страницы нужно обойти всех её соседей (в худшем случае их N).

Кластеризация на основе категорий статей

Предлагается следующий алгоритм иерархической кластеризации, позволяет объединить документы в смысловые группы. Определим структуры данных, используемые в алгоритме, представим блок-схему (Рис. 10) и псевдокод алгоритма.

Структуры данных и параметры алгоритма:

- $G=(V, E)$ – направленный граф, где V – вершины (статьи и категории, то есть два типа страниц, два типа вершин), E – дуги (три типа дуг: между статьями, между категориями, между статьями и категориями).
- E_{sorted} – массив рёбер, упорядоченный по весу ($E_{sorted}[0]$ – ребро с минимальным весом).
- Clusters – список кластеров, которые нужно построить.
- C_{max} – максимально допустимый вес кластера (число статей и категорий в кластере).

Для каждого ребра e определены следующие поля:

- e_{c1} и e_{c2} – это указатели на два соединяемых кластера;

1 Асимптотический анализ и, в частности, O-оценивание см. в [20], стр. 49-50.

- e_{weight} – вес ребра (пересчитывается), равный суммарному весу объединяемых кластеров c_1 и c_2.

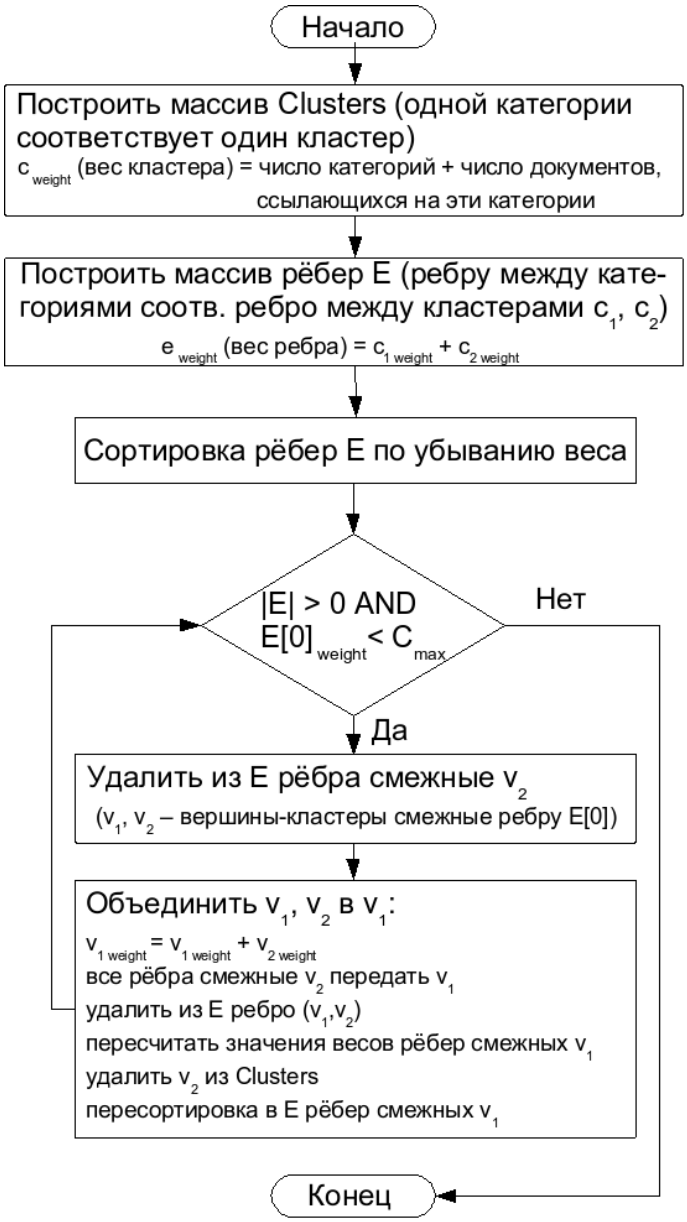


Рис. 10. Иерархическая кластеризация

Для кластера-вершины *c* определены такие поля:

- $|c_{edges}|$ – число объединённых рёбер кластера (рёбер между вершинами-категориями кластера);
- $|c_{articles}|$ – число статей, которые ссылаются на категории в кластере (знак мощности множества |.| используется в силу его наглядности, однако, поскольку $|c_{articles}|$ – это переменная, то ей можно присваивать значение (см. напр. строку 5b в алгоритме);

- c_{weight} – вес кластера;
- Г(c) – массив рёбер, связывающих кластер с другими кластерами.

При вычислении веса кластера учитываются: число статей в кластере, веса объединяемых кластеров (см. формулу (2.7) ниже).

Процесс кластеризации состоит из двух шагов: предобработка (инициализация массива кластеров, присвоение начальных значений полям вершин и рёбер) и сам алгоритм кластеризации.

Предобработка

(две косые черты '//' отделяют комментарий от псевдокода)

1. Построить кластеры (массив Clusters) по категориям: изначально каждый кластер соответствует отдельной вершине (категории). Приписать каждому кластеру (за счёт содержащихся в кластере категорий):

   a) $|c_{articles}|$ = число статей, которые ссылаются на категории в кластере,

   b) $c_{weight} = 1 + |c_{articles}|$ // изначально вес кластера – это число категорий в кластере (изначально одна) и число статей, которые ссылаются на эту одну категорию;

   c) $c_{category\_id}[0] = category_{id}$ // присваиваем кластеру уникальный идентификатор id первой (и единственной пока) категории, добавленной в кластер[1].

2. Для каждого ребра между категориями создать ребро между кластерами. Каждому ребру *e*, соединяющему два кластера c1 и c2 определить вес так:

$$e_{weight} = c1_{weight} + c2_{weight}$$

1 У каждой страницы (категории или статьи) Википедии есть уникальный идентификатор.

Алгоритм

1. E_{sorted} = sort(e_{weight}); // сортировка рёбер по весу
2. while(|E_{sorted}| > 0 && ($E_{sorted}[0] < C_{max}$)) BEGIN
3. e = E_{sorted} [0]; // v_1, v_2 – вершины смежные ребру e
4. $E_{sorted} = E_{sorted} \setminus \Gamma(v_2)$; // удалить из упорядоченного массива рёбер рёбра смежные v_2
5. merge(e); // объединить вершины-кластеры v_1 и v_2 в кластер v_1, то есть добавить вершину v_2 в кластер v_1, изменив свойства v_1 так:
   a) $v_{1\ weight}$ += $v_{2\ weight}$; // увеличить размер кластера (число категорий и статей)
   b) |$v_{1\ articles}$| += |$v_{2\ articles}$|; // увеличить число статей
   c) |$v_{1\ edges}$| += |$v_{2\ edges}$|; // увеличить число рёбер
   d) $v_{1\ category\_id}$[] += addUnique($v_{2\ category\_id}$[]); // добавить категории без повторов (на основе уникальности идентификаторов)
6. passEdges(); // все рёбра смежные вершине v2 передать вершине v1 (рёбра без повторений, это не мультиграф).
7. $E_{sorted} = E_{sorted} \setminus$ edge (v_1, v_2); // удалить ребро (v_1, v_2);
8. updateEdgesOfMergedCluster; // обновить указатели на вершины, удалить ребра (вершины и рёбра, смежные удаляемой вершине)
9. updateEdgeWeight(v_1); // пересчитать значения весов для всех рёбер смежных v1
10. remove(Clusters, v_2); // удалить кластер v_2 из массива кластеров
11. E_{sorted} = sort(e_{weight}) // пересортировка рёбер, сложность O(N), т. к. нужно обновить порядок только тех рёбер, которые смежны вершине v_1
12. END
13. Return Clusters

Результат работы алгоритма – это кластеры категорий (Clusters). По кластеру категорий получаем кластер статей, поскольку для каждого кластера *c* определено значение $c_{articles}$ (список статей, ссылающиеся на категории в кластере).

Заметим, что одна статья может ссылаться на несколько категорий. Поэтому одна статья может принадлежать нескольким кластерам. Но категория принадлежит ровно одному кластеру.

Варианты объединения результатов AHITS алгоритма и алгоритма кластеризации

Могут быть реализованы следующие варианты:

1) Использовать информацию о каждой вершине (hub, authority) (в алгоритме кластеризации) для вычисления начального веса кластера на этапе предобработки, в шаге 1б:

$$|c_{weight}| = 1 + k \sum_{v:\, v \in C}^{n^v_{article}} (h_v + a_v) \qquad (2.7)$$

Здесь h_v и a_v вычислены с помощью адаптированного HITS алгоритма, k – весовой коэффициент (параметр), наличие единицы объясняется тем, что на этапе предобработки кластер содержит только категорию. Суммирование проводится по всем статьям, которые ссылаются на категории кластер.

2) Выбрать (после кластеризации) из каждого кластера подмножество *hub, authority* статей с максимальным весом и представить пользователю как *наборы синонимов и близких по значению слов (для исходного слова), сгруппированные по значению*. Открытым остаётся вопрос – как определить число значений (здесь – кластеров) у слова? На данный момент число кластеров определяется максимально разрешённым весом кластера (параметр алгоритма кластеризации C_{max}).

Временная сложность алгоритма

Обозначим C – число категорий, N – число статей (число страниц в базовом наборе). Первый шаг предобработки требует $O(C \cdot N)$ операций, второй – $O(C^2)$. Тогда предобработка в целом требует $O(C \cdot N + C^2)$ операций.

В алгоритме первый шаг (сортировка рёбер) требует не более $O(C^2)$ операций. Циклов (шаг 2) должно быть не больше числа рёбер, то есть $O(C^2)$, так как на каждом витке цикла удаляется одно ребро. Наиболее вычислительно затратным будет пятый шаг, так как для объединения вершин необходимо $O(C+N)$ операций, поскольку нужно обойти категории и статьи, соседние для объединяемых кластеров. Тогда предобработка и алгоритм кластеризации в целом требуют

$$O(C \cdot N + C^2) + O(C^2 \cdot (C + N)) = O(C^3 + N \cdot C^2)$$

Временная сложность [20] вычисления весов (hub, authority) в HITS алгоритме составляет $O(I_k \cdot N^2)$, где I_k – число итераций в адаптированном HITS алгоритме. Тогда общая сложность адаптированного HITS алгоритма будет:

$$O(C^3 + N \cdot C^2 + I_k \cdot N^2) \quad (2.8)$$

где: I_k – число итераций на шаге 3 в адаптированном HITS алгоритме;

C – число категорий;

N – число статей (число страниц в базовом наборе).[1]

Эвристика: фильтрация на основе категорий статей

Предлагается следующий способ сужения поиска похожих документов в корпусе документов с категориями.

Дано: $N \in \mathbb{N}$ (глубина поиска) и $Category_{Blacklist}$ – чёрный список категорий, про которые заранее известно, что документы с такими категориями не подходят, то есть не являются похожими на исходный документ.

Для использования этих данных нужно выполнить фильтрацию страниц при построении корневого и базового набора страниц (первые два шага адаптированного HITS алгоритма), состоящую в следующем. Если категории страницы, или их надкатегории (не более N уровней относительно исходной страницы[2]) принадлежат списку $Category_{Blacklist}$, то страница пропускается и не добавляется в корневой / базовый набор, иначе – добавляется.

Данная эвристика была реализована в программе Synarcher, см. подраздел «Модуль визуализации: интерфейс и функции» на стр. 99.

2.4 Вычисление меры сходства вершин графа. Оценка временной сложности. Эвристики

Дано: направленный граф $G=(V, E)$, где V – вершины (страницы), E – дуги (ссылки), $w \in V$. Для каждой страницы v известны два списка: $\Gamma^+(v)$ – это страницы, на которые ссылается данная статья, и $\Gamma^-(v)$ – это страницы, ссылающиеся на данную статью. Для каждой вершины определены значения двух весовых коэффициентов *authority* и *hub*: $\{v \in V : a_v \in \mathbb{R}, h_v \in \mathbb{R}\}$.

1 Из числа статей и категорий в Английской (659 358, 113 483) и Немецкой (305 099, 27 981) Википедиях (на 01.2006 г. по [94]) следует, что три слагаемых оценки отличаются друг от друга менее чем в 10^4.

2 Полагаем, что категория *c1* страницы – это первый уровень, надкатегория *c2* категории *c1* – это второй уровень, надкатегория *c3* категории *c2* – это третий уровень и т.д.

Необходимо найти набор вершин *A*, похожих на вершину *w*. Определение *похожести* вершин графа приведено ниже и оно отличается от приведённого в подразделе «Формализация понятия «похожие вершины» графа» на стр. 76 тем, что (i) в результате работы алгоритма строится массив сходства вершин, (ii) не используются понятия авторитетных и хаб-страниц.

Задачу *поиска похожих статей*, которую решает адаптированный HITS алгоритм можно переформулировать в терминах теории графов так.

Задача поиска похожих вершин графа.

Дано: направленный граф *G=(V, E)*, где *V* – вершины, *E* – дуги, $w \in V$.

Найти: *n* вершин $z_i \in V, i = 1 \dots n$ графа *G* максимально похожие на *w*, упорядоченные по степени сходства. Степень сходства $y(a,b)$ между двумя вершинами *a* и *b* определим рекуррентно:

$$y(a,b) = \frac{k \cdot |\Gamma^+(a) \cap \Gamma^+(b)| + (1-k) \cdot \sum\limits_{x \in \Gamma^+(a),\, y \in \Gamma^+(b)} y(x,y)}{\sum\limits_{(v,w) \in (V \times V)} y(v,w)}, 0 \le k \le 1 \qquad (2.9)$$

Первое слагаемое[1] в числителе (2.9) – это число общих вершин среди соседей вершин *a* и *b*, второе слагаемое – это суммарное сходство соседей вершин *a* и *b*. В знаменателе – суммарное сходство по всем парам графа *G*. Весовой коэффициент *k* позволяет выбрать компонент, который оказывает большее влияние на суммарное сходство: число общих соседей или степень сходства между соседними вершинами.

Таким образом, $y(a,b) = 0$, если (1) у вершин *a* и *b* нет общих соседей и (2) среди вершин соседних *a* и *b* степень сходства равна нулю:

$$\sum_{x \in \Gamma^+(a),\, y \in \Gamma^+(b)} y(x,y) = 0 \qquad (2.10)$$

где $\Gamma^+(a), \Gamma^+(b)$ – это множество соседних вершин для вершин *a* и *b* соответственно. Такая постановка задачи определяет следующий алгоритм поиска похожих вершин графа.

---

1 Таким образом, получили рекуррентную формулу вычисления меры сходства между вершинами одного графа алгоритма SimRank [119] (стр.5, формула (5)). Разница заключается в наличии первого слагаемого в формуле (2.9), определяющего число общих соседей. Очевидно, что об общих соседях можно говорить в случае поиска похожих вершин для одного графа, а не в общем случае сравнения вершин разных графов.

Алгоритм поиска похожих вершин графа

CalcSimilarVertices(G, ε, k)

G=(V,E) – граф с *N* вершинами

$\varepsilon, k \in \mathbb{R}$, где

ε – погрешность вычислений

k – весовой коэффициент

Y[][] – сходство между вершинами графа

1. // инициализация Y[1]
2. for(i=0; i<N;i++) {
3. for(j=0; j<N;j++) {
4. Yold[i][j] = Y[i][j] = $|\Gamma^+(v_i) \cap \Gamma^+(v_j)|$ / N^2
5. }
6. }
7. // итеративное вычисление Y
8. error = 1 + ε
9. while (error > ε) {
10. norma = 0
11. for(i=0; i<N;i++) {
12. for(j=0; j<N;j++) {
13. $Y[i][j] = k \cdot |\Gamma^+(v_i) \cap \Gamma^+(v_j)| + (1-k) \cdot \sum_{x \in \Gamma^+(v_i), y \in \Gamma^+(v_j)} y(x, y)$
14. norma += Y[i][j]
15. }
16. }
17. for(i=0; i<N;i++) {
18. for(j=0; j<N;j++) {
19. $Y[i][j] = \frac{Y[i][j]}{norma}$
20. Ydelta[i][j] = |Y[i][j] – Yold[i][j]|
21. Yold[i][j] = Y[i][j]
22. }
23. }
24. $error = \sum_{i,j=1}^{N} Ydelta[i][j]$
25. }

1 Другой вариант инициализации – начальное значение весов вершин может быть вычислено, например, с помощью HITS алгоритма (вес authority берётся за начальный).

В шагах 13, 14, 19 реализуется формула (2.9) вычисления сходства между *i*-ой и *j*-ой вершинами графа. Шаги 20, 21 и 24 вычисляют суммарное изменение (error) на данном шаге итерации (цикл while).

В результате работы алгоритма получаем симметричный массив сходства вершин Y[][]. Теперь для поиска упорядоченного списка максимально похожих вершин, например на *j*-ую, достаточно отсортировать одномерный массив Y[j][].

Остаются открытыми следующие вопросы:

- доказательство сходимости итеративного алгоритма;
- скорость сходимости в реальных экспериментах.

Оценка временной сложности

Шаги вложенных циклов 9, 11, 12 и 9, 17, 18 дают сложность $O(I \cdot N^2)$, где I – число итераций, N – число вершин в графе.

Шаг 13 требует поиска пересечений всех соседей двух вершин. Поиск пересечения двух множеств размерности N требует $O(N)$ операций (например с помощью меток). Тогда временная сложность алгоритма $O(I \cdot N^3)$. Постараемся уменьшить временную сложность [20] алгоритма с помощью эвристик.

Эвристики[1]

1) В первой эвристике предлагается рассматривать не все пары вершин (как кандидаты на «похожие» вершины), а только те, у которых совпадают не менее L соседей.

При этом циклы на шагах 11, 12 и 17, 18 потребуют не $O(N^2)$ операций, а $O(M^2)$, где M – число вершин, которые имеют не менее L общих вершин.

Число M можно определить (сравнив попарно вершины на наличие не менее L соседей) до выполнения алгоритма за время $O(N^3)$, где N – число вершин графа. Вычислив M, можно принять решение: будет ли эвристика

1 В алгоритме SimRank [119] предлагается следующая эвристика: отсекать вершины (pruning), находящиеся на значительном расстоянии от вершины v, и поэтому, вероятно, имеющей мало общих вершин с отсекаемыми. Таким образом, в SimRank рассматриваются (как потенциально похожие) вершины, удалённые не более чем на радиус r друг от друга (где радиус – это один из параметров алгоритма).

давать значительный выигрыш в скорости при данном L либо нужно увеличить L и повторить вычисления.

Таким образом, вычисление степени сходства между M вершинами с помощью функции *CalcSimilarVertices* потребует времени $O(I \cdot M^3)$. После этого один раз выполняется тело цикла *while* (шаги 10-23) для вычисления степени сходства между всеми N вершинами за $O(N^2)$ операций. Таким образом, суммарная временная сложность составит $O(I \cdot M^3 + N^3)$.

2) Предлагается эвристика (для алгоритма поиска похожих вершин графа), использующая результаты работы HITS алгоритма.

После применения HITS алгоритма к графу $G=(V, E)$ с N вершинами получено M особых вершин (а именно: авторитетные и хаб-страницы), причём M – это параметр алгоритма.

Временная сложность вычисления весов (hub, authority) в HITS алгоритме составляет $O(I_k \cdot N_k^2)$, где I_k – число итераций в HITS алгоритме, N_k – число страниц в базовом наборе.

Число итераций I (вслед за работой[1]) можно оценить, как: $O(N \cdot \frac{h \log h}{k})$, где h – это число правильно найденных авторитетных вершин, всего найдено k авторитетных вершин. Необходимое условие: $O(N \geq k)$.

Далее аналогично первой эвристике: (1) выполнение функции *CalcSimilarVertices* потребует $O(I \cdot M^3)$ операций для вычисления значения степени сходства между M вершинами и (2) $O(N^2)$ для вычисления степени сходства между всеми N вершинами. Суммарная временная сложность будет $O(I \cdot M^3 + N^2 + I_k \cdot N_k^2)$.

Преимущество перед первой эвристикой второй эвристики (вычисляемой соответственно за $O(I \cdot M^3 + N^3)$ и $O(I \cdot M^3 + N^2 + I_k \cdot N_k^2)$ операций) в том, что число обрабатываемых вершин M задаётся напрямую, а не вычисляется (за $O(N^3)$) через параметр L (число общих соседей).

1 Peserico E., Pretto L. The rank convergence of HITS can be slow. 2008. http://arxiv.org/abs/0807.3006.

2.5 Показатели численной оценки семантической близости списка слов

Для экспериментальной оценки алгоритма необходимо научиться численно автоматически оценивать набор слов, сформированный AHITS алгоритмом. Можно указать, по крайней мере, два способа использования такой оценки.

Во-первых, параметры алгоритма влияют и на время работы и на качество результата. Оценка позволит найти золотую середину работы алгоритма (минимизировать время работы и повысить точность, полноту и качество поиска) и выбрать входные параметры алгоритма. Во-вторых, при одних и тех же параметрах необходимо оценить, как сказывается на результатах модификация алгоритма, применение эвристик. В первом, а особенно во втором случае, к системе предъявляется требование: обеспечить пакетную обработку запросов с возможностью табличного вывода результатов удобного для анализа.

Таким образом, разработано несколько показателей численной оценки полученного набора синонимов: SER, ER и коэффициент Спирмена.

SER (Strong Error Rate) – число слов (в процентах от общего числа найденных слов), которые не имеют отношения к исходному слову (определяется либо вручную – экспертом, либо автоматически – по наличию/отсутствию в тезаурусе, например WordNet, Moby[1]).

$$SER(w) = \frac{|Words^{w}_{AHITS} \setminus (Words^{w}_{WordNet} \cup Words^{w}_{Moby})|}{|Words^{w}_{AHITS} \cup Words^{w}_{WordNet} \cup Words^{w}_{Moby}|} \cdot 100 \qquad (2.11)$$

ER (Error Rate) – число слов (в процентах), не являются синонимами (определяется вручную экспертом либо автоматически – по наличию/отсутствию в тезаурусе WordNet).

$$ER(w) = \frac{|Words^{w}_{AHITS} \setminus Words^{w}_{WordNet}|}{|Words^{w}_{AHITS} \cup Words^{w}_{WordNet}|} \cdot 100 \qquad (2.12)$$

Где $Words^{w}_{AHITS}$ – множество слов, найденных с помощью адаптированного HITS алгоритма для слова w, $Words^{w}_{WordNet}$ – список синонимов из тезауруса WordNet для слова w, $Words^{w}_{Moby}$ – список слов

1 Moby Thesaurus List by Grady Ward, см. http://www.dcs.shef.ac.uk/research/ilash/Moby

близких по значению из тезауруса Moby. Очевидно, что $ER \geq SER$, поскольку в *SER* вычитаются слова из обоих тезаурусов: WordNet и Moby.

Открытые вопросы:

1. Сильно ли изменятся значения *SER* и *ER*, если в знаменателе будет стоять только $Words^{w}_{AHITS}$? Вероятно, это сильно связано с числом найденных синонимов адаптированным HITS алгоритмом.

2. Первый вопрос определяет второй: как для упорядоченной последовательности синонимов, возвращаемой алгоритмом (теоретически неограниченной, например, несколько тысяч), определить: где оборвать эту последовательность, обеспечивая некоторое приемлемое качество? Возможный ответ таков. После работы алгоритма у каждой вершины графа определён вес authority, который говорит о степени авторитетности (в нашем случае – синонимичности). Выбрав эмпирически некоторое вещественное *d*, $0 \leq d \leq 1$, можно отсекать множество синонимов, выбирая только те, у которых $authority \geq d$. Либо, вычислив последовательность *SER* и *ER* (где например SER_i соответствует первым i-ым словам результата), оборвать последовательность на *N*-том члене при $SER_N \geq d$.

Коэффициент Спирмена

Самый простой способ сравнить два набора данных в том, чтобы найти количество общих элементов[1].

В [76] метрика Спирмена (Spearman, другое название – Spearman's footrule) используется как для сравнения ранжирования одного и того же набора Интернет страниц разными поисковиками, так и для оценки изменения ранжирования рассматриваемого набора страниц во времени.

Метрика Спирмена[2] позволяет сравнить две ранжировки одного и того же набора из *N* элементов. Каждому элементу назначается ранг от 1 до *N*

1 Данный способ использовался в экспериментах, см. графу *Intersection* в таблице 4.8.

2 В статистике коэффициент корреляции Spearman – это непараметрическая мера корреляции, оценивающая насколько хорошо произвольная монотонная функция может описать отношение между двумя переменными в случае, когда неизвестен характер зависимости переменных (линейный или какой-либо др.). См. http://en.wikipedia.org/wiki/Spearman's_rank_correlation_coefficient

(мера основана на перестановках). Метрика Spearman равна сумме модулей расстояний между i-ми элементами набора.

$$F^S(s_{1,}s_2)=\sum_{i=1}^{S}\left|s_1(i)-s_2(i)\right| \qquad (2.13)$$

Предположим, при поиске синонимов для данного слова и для двух вариантов начальных параметров адаптированного HITS алгоритма I_1 и I_2 (размер корневого набора, число добавляемых страниц и др.) получили два набора слов S_1 и S_2. Задача заключается в том, чтобы выяснить, какие параметры алгоритма дают лучший результат. Это можно сделать, сравнивая множества слов S_1 и S_2 с эталонным множеством синонимов и близких по значению слов для данного слова (выбранных экспертом или найденных с помощью тезауруса). Для сравнения предлагается использовать метрику Спирмена.

Предлагается расширить метрику Спирмена для сравнения списков разной длины[1] следующим способом, отличающимся[2] от расширения предложенного в [76]. Дано два списка: (1) A – более короткий, эталонный, составленный экспертом, (2) B – более длинный, составленный автоматически. В конец списка B добавляются отсутствующие в нём элементы A. Далее применяется формула (2.13), где сравниваются положения в списках общих элементов, S – число общих элементов, а также длина более короткого списка.

Далее метрику Спирмена, расширенную таким способом, будем называть *коэффициентом Спирмена*, так как, вообще говоря, такое расширение не сохраняет свойства метрики (например, может нарушаться свойство симметрии).

Таким образом, с помощью коэффициента Спирмена можно сравнить ранжирование одного и того же набора слов адаптированным HITS алгоритмом при разных параметрах поиска с эталонным списком.

---

1 В экспериментах необходимо будет сравнивать списки разной длины. Один небольшой по длине список постоянной длины (составлены экспертом), другой список (может быть намного более длинным) строится автоматически программой, его длина может меняться от эксперимента к эксперименту.

2 Разница в том, что в работе [76] сравниваются результаты поисковых серверов, ни один из которых не является эталоном для другого. В экспериментах данной работы один из списков является эталонным.

Выводы по главе 2

В данной главе были разработаны: (1) подход поиска СБС, (2) формализация понятия похожие вершины, (3) адаптированный алгоритм HITS с использованием алгоритма кластеризации, (4) предложенный автором алгоритм поиска похожих вершин графа и (5) предложенные автором методы численной оценки наборов синонимов (модификация коэффициента Спирмена (Spearman's footrule) и оценка на основе тезаурусов WordNet и Moby).

HITS алгоритм был адаптирован автором к тем дополнительным возможностям, которые предоставляет рассматриваемый корпус документов, относительно обычных Интернет страниц. Это (1) наличие категорий (классифицирующих документы по их тематической принадлежности), (2) наличие метаинформации в виде ключевых слов (в простейшем случае - это заголовок документа).

В адаптированном HITS алгоритме предложено применить алгоритм иерархической кластеризации к базовому набору страниц для получения групп страниц, соответствующих разным тематическим направлениям. Проведена оценка временной сложности адаптированного HITS алгоритма, см. (2.8). Предложено два варианта объединения результатов работы адаптированного HITS алгоритма и алгоритма кластеризации.

Автором предложены два варианта формализации понятия «похожие вершины» графа. Первый вариант использует понятия авторитетных и хаб-страниц и позволяет формализовать задачу поиска похожих страниц в HITS алгоритме. Во втором варианте получена формула сходства двух вершин a и b, основанная на поиске общих вершин среди соседей вершин a и b.

Предложен алгоритм поиска похожих вершин графа. Выполнена оценка временной сложности данного алгоритма. Предложены две эвристики, позволяющие уменьшить временную сложность алгоритма. В первой эвристике предлагается рассматривать не все пары вершин (как кандидаты на «похожие» вершины), а только те, у которых не менее L

соседей совпадают. Во второй эвристике используются результаты работы HITS алгоритма.

Предложены методы численной оценки наборов синонимов, полученных на выходе адаптированного HITS алгоритма. Оценка позволит выбрать входные параметры алгоритма. Для оценки работы алгоритма с английским корпусом текстов предлагается использовать тезаурусы WordNet и Moby.

Метрика Spearman's footrule (коэффициент Спирмена) обычно используется как для сравнения ранжирования одного и того же набора данных. Коэффициент Спирмена модифицирован для численного сравнения списков слов, отличие от оригинального метода заключается в возможности сравнивать списки разной длины. Предлагается с помощью метрики Spearman's footrule сравнить ранжирование одного и того же набора слов, полученных адаптированным HITS алгоритмом при разных параметрах поиска.

3. Организация программного обеспечения поиска семантически близких слов, автоматической оценки поиска и морфологического анализа слов

Предложенный вниманию читателя во второй главе адаптированный HITS алгоритм алгоритм реализован в программе, названной Synarcher[1]. В рамкам описания архитектуры программы представлены основные классы и методы программы, программный интерфейс доступа к Википедии и особенности реализации модуля визуализации.

В этой главе представлена архитектура и реализация программного модуля системы GATE для удалённого доступа к программе морфологического анализа русского языка (aot.ru) на основе XML-PRC протокола. Данная архитектура представляет способ интеграции приложений написанных на разных языках программирования (например, C++ и Java) посредством XML-RPC протокола. Спроектирована архитектура системы индексирования вики-текстов, включающая GATE и Lemmatizer.

В главе представлена архитектура программной системы оценивания синонимов, позволяющей реализовать метод численной оценки списков семантически близких слов (адаптация метода Spearman's footrule и оценка на основе тезаурусов WordNet и Moby). Данный метод численный оценки описан в подразделе 2.5.

3.1 Архитектура программной системы Synarcher

Достоинствами поискового комплекса Synarcher являются (1) визуализация результатов поиска, (2) возможность уточнения запроса в ходе работы пользователя с программой.

В архитектуру программного комплекса Synarcher (рис. 11) включены такие модули, как: модуль *kleinberg*, предоставляющий доступ к данным Википедии и реализующий алгоритм AHITS, модуль визуализации *TGWikiBrowser*[2]. Последовательность взаимодействия частей системы

1 Поскольку задачу программы можно сформулировать так: SYNonym seARCH.

2 Данный пакет представляет собой модификацию программы *TouchGraph Wiki Browser* (см. http://www.touchgraph.com). Можно отметить популярность данной программы, например она

указаны на рис. 11 числами (1-7). Входными данными для AHITS алгоритма являются слово, заданное пользователем (1-2), и данные Википедии (2). Алгоритм строит список СБС, упорядоченных по весу authority (3), а пользователь получает возможность работать с ними благодаря модулю визуализации TGWikiBrowser (4-5). В ходе работы пользователь уточняет список СБС и запускает алгоритм повторно (6-7).

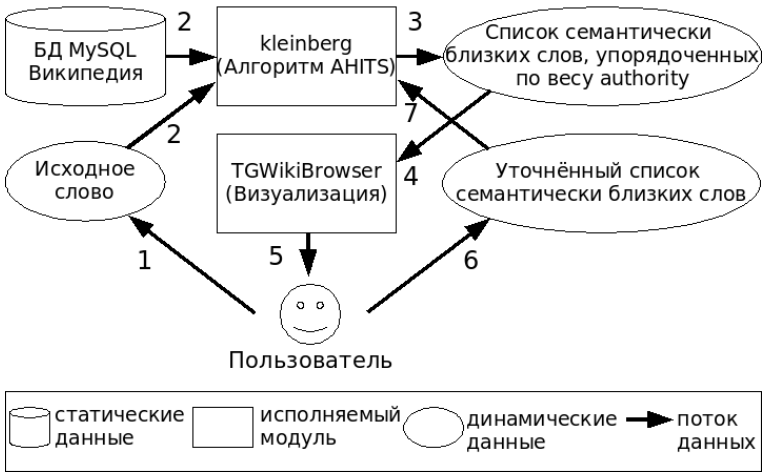


Рис. 11. Архитектура программного комплекса Synarcher

Программа написана на языке Java [68], [8], [115]. Для неусыпного контроля работоспособности разных частей программы использовалась методика экстремального программирования – автоматические тесты модулей[1] [6], [9], [188].

Программа состоит из двух основных частей. Первая часть отвечает за доступ к базе данных Википедия, содержит реализацию алгоритмов поиска синонимов и вспомогательные функции (пакет *kleinberg*). Вторая часть представляет пользовательский интерфейс, посредством которого запускаются функции первой части (пакет *TGWikiBrowser*).

использовалась в прототипе американских учёных [71] для визуального представления отношений RDF. Недостаток программы в том, что разработчики больше не поддерживают данную программу, а также в том, что программа работает не устойчиво при отображении большого числа вершин и рёбер (например, если вершин больше сотни).

1 На июнь 2007 проект kleinberg программы Synarcher содержал 119 тестов, на июнь 2008 – 241 тест.

Модуль алгоритмов

Основные наборы файлов (package) первого проекта – это rfc2229, wikipedia.clustering, wikipedia.kleinberg, wikipedia.sql, wikipedia.util. Их назначение состоит в следующем:

- rfc2229 – клиент к Dict серверам[1], основанный на библиотеке JavaClientForDict (jcfd), разработанной Davor Cengija; расширение данного клиента для удалённого вызова тезаурусов WordNet и Moby;
- wikipedia.clustering – реализация алгоритма кластеризации, описанного в главе 2;
- wikipedia.kleinberg – реализация адаптированного HITS алгоритма и основные структуры данных, используемые в работе этого алгоритма (Article, Category, Node). Также этот пакет содержит вспомогательный класс DumpToGraphViz, позволяющий текущее состояние графа сохранять в специальном формате, который может быть преобразован в форматы PNG, JPEG или SVG программой GraphViz[2];
- wikipedia.sql – это программный интерфейс доступа к Википедии. Пакет позволяет читать данные Википедия, хранимой в базе данных MySQL. Названия классов соответствуют таблицам, с которыми они работают: PageTage, Links, Category. Класс Statistics выдаёт статистическую информацию (число статей, категорий, ссылок) о подключенной Википедии.
- wikipedia.util – пакет содержит вспомогательные классы. Encoding – содержит функции преобразования кодировок, FileWriter – работа с файлами, RandShuffle – обеспечивает случайную перестановку данных в исходном массиве, StringUtil – вспомогательные функции по обработке строковых данных, StringUtilRegular – вспомогательные функции по обработке строковых данных с использованием регулярных выражений [63].

1 Дополнительная информация по Dict серверам и протоколу RFC-2229 на сайте http://www.dict.org

2 См. http://www.graphviz.org

Модуль визуализации: интерфейс и функции

Представление в виде графа результатов поиска синонимов и семантически близких слов основывается на программе TouchGraph WikiBrowser V1.02[1]. Данная программа предназначена для визуализации Wiki страниц. Программа TouchGraph WikiBrowser была существенно модифицирована для того, чтобы обеспечить следующую функциональность:

- подключение к базе данных Википедия, поскольку клиент-серверная архитектура базы данных MySQL позволяет программе Synarcher обращаться к базе данных Википедия, установленной на удалённом компьютере (рис. 13);
- задание параметров адаптированного HITS алгоритма (рис. 14);
- хранение слов, помеченных пользователем, как синонимы, и параметров поиска на компьютере пользователя. Причём запоминаемые параметры делятся на (1) параметры программы в целом (класс BrowserParameters), например, название базы данных с которой работали, последнее искомое слово и (2) параметры поиска отдельного слова (класс ArticleParameters);
- расширение контекстного меню (рис. 15) для более удобной навигации командами, позволяющими спрятать все вершины (Hide all except node), пометить вершину как синоним (Rate synonym), показать категории (Expand Categories).

Экран программы делится на две части вертикальной полосой, с левой стороны три вкладки[2] (англ. *tab*): *Article*, *Database* и *Synonyms*.

Первая вкладка *Article* позволяет просмотреть энциклопедическую статью, соответствующую выбранному слову (рис. 12). Адрес статьи (URL) можно увидеть внизу экрана в строке статуса. Представление статьи Википедии в этом мини браузере (то есть во вкладке *Article*) и, вообще, в Интернет браузере возможен благодаря слаженной работе на стороне сервера таких программ, как: MySQL, Apache, PHP и MediaWiki.

1 См. http://www.touchgraph.com

2 См. http://ru.wikipedia.org/wiki/Вкладка

Вкладка *Database* позволяет: (1) указать кодировку, (2) указать параметры для подключения к базе данных, (3) проверить подключение и получить статистику по базе данных (рис. 13). В верхней части экрана находится выпадающее меню (поле Wikipedia), позволяющее выбрать одну из баз данных Википедия. На данный момент это английская и русская версии энциклопедии.

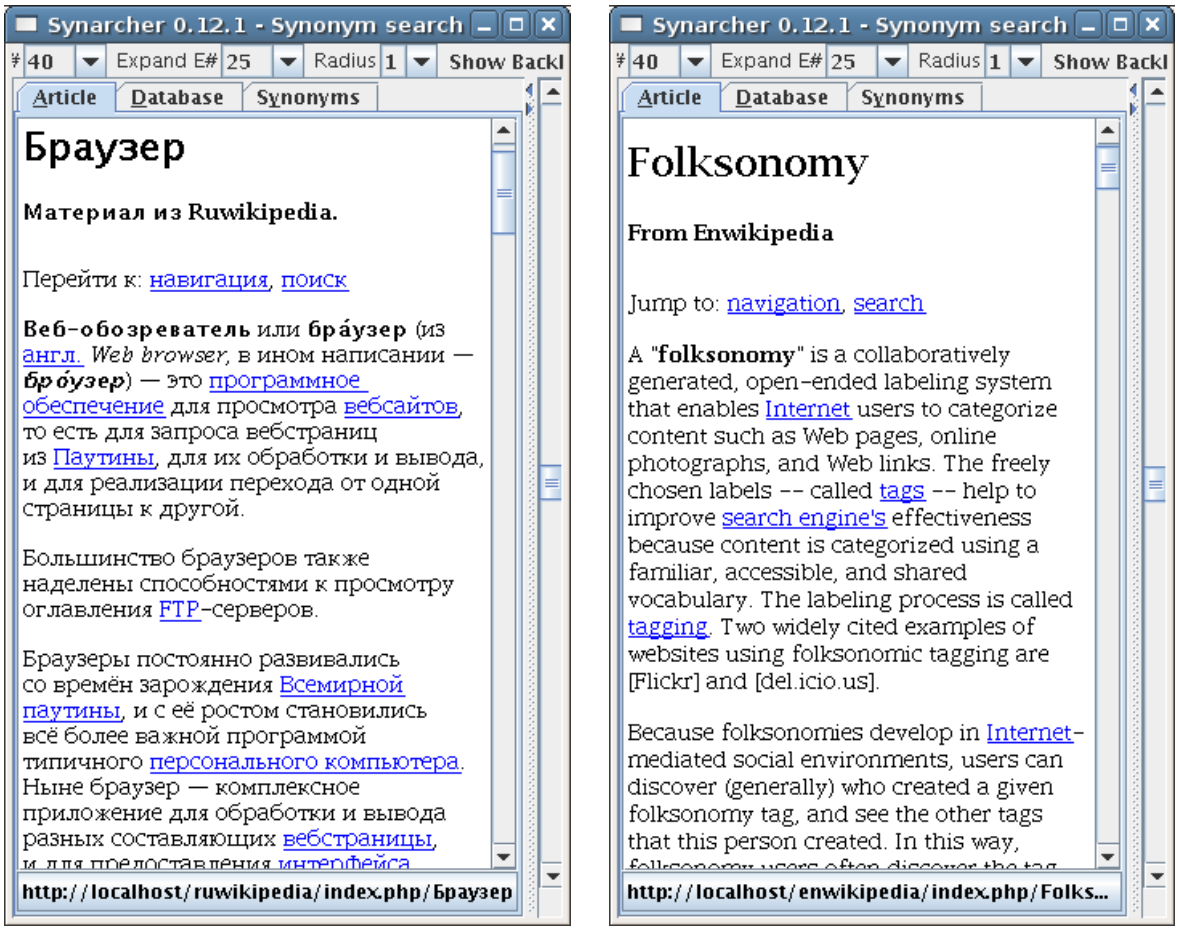


Рис. 12. Отображение статей «Браузер» и «Folksonomy» из Русской и Английской Википедии соответственно во вкладке Article

Возможность указать кодировку (по умолчанию UTF8[1]) призвана решить возможные проблемы, связанные с хранением текстовых данных на стороне пользователя (например, название последней просмотренной статьи). Дополнительная информация по настройке кодировки в программе представлена на странице проекта[2].

1 «Кодировка переменной длины, предназначенная для отображения всех символов стандарта Unicode, при этом совместимая со стандартом ASCII» (http://en.wikipedia.org/wiki/UTF-8)

2 http://synarcher.sourceforge.net

На рис. 13 показаны параметры подключение к БД Русской Википедии, установленной локально (группа параметров «*MySQL database connection parameters*»):

- *Database host* – адрес компьютера на который установлена энциклопедия, в данном случае (рис. 13) она установлена на том же компьютере, что и запущенная программа Synarcher, поэтому указано значение: *localhost*;
- *Database name* – указывается имя базы данных в MySQL и параметры подключения к базе. Здесь база данных названа *ruwiki*, параметры подключения представлены в нижней части экрана в окне Output (в круглых скобках);
- *User, Password* – имя пользователя и пароль под которыми программа подключается к базе данных. Значения определяются настройками MySQL и MediaWiki. Здесь указан пользователь *javawiki*, пароль отсутствует.
- *Wiki url* – URL-адрес, используемый веб браузером для поиска статьи Википедии. Необязательный параметр для работы алгоритмов поиска, нужен для отображения страниц во вкладке Article (рис. 12). Значение определяется настройками Apache и MediaWiki.

При нажатии на кнопку «*Get statistics»* (рис. 13) в окне Output печатается статистика о подключенной базе данных энциклопедии (число статей, ссылок между статьями, категорий, ссылок между категориями, изображений, ссылок на изображения). Возможность получить эту информацию можно использовать для проверки успешного подключения к базе данных.

Во вкладке *Synonyms* (рис. 14) задаются параметры адаптированного HITS алгоритма (дополнительная вкладка *Parameters*), выводятся результаты в табличной и текстовой форме (вкладка *Results*). Слово для поиска задаётся в поле *Word*. Кнопка *Load* позволяет загрузить параметры предыдущего поиска, кнопка «*Search Synonyms»* запускает алгоритм поиска, а кнопка *Draw* отображает (в правой части экрана в виде сетевой структуры) страницы, на которые ссылается заданное слово.

В поле *Log* пользователь (с помощью кнопки «*Browse...*») указывает директорию в которой будут храниться результаты поиска.

Поля «*Root set size*», *increment, «N synonyms», «Eps error»* соответствуют параметрам адаптированного HITS алгоритма: *t, d, N*, *ε* (см. подраздел 2.1 на стр. 68).

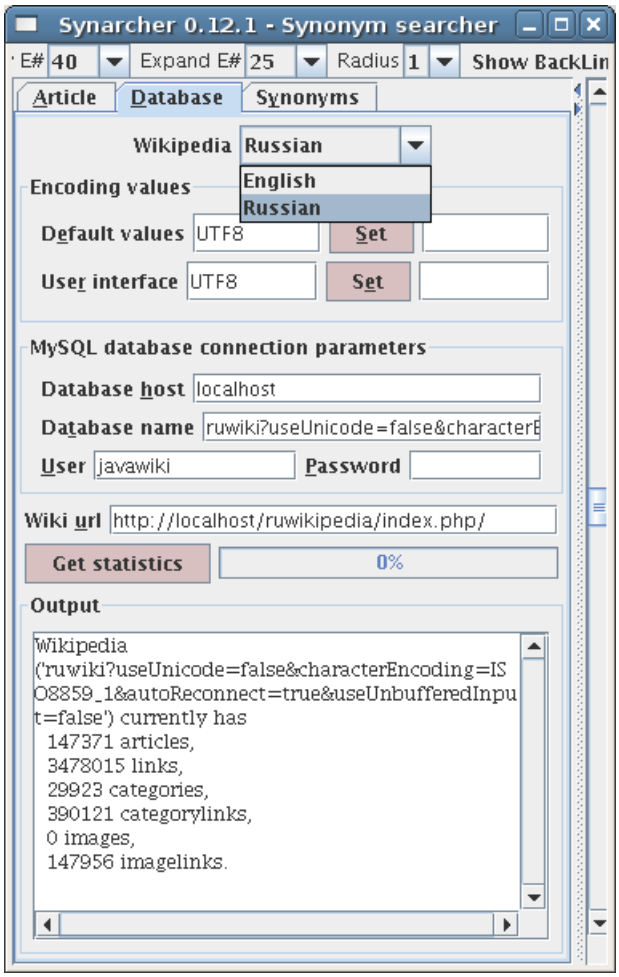


Рис. 13. Вкладка Database

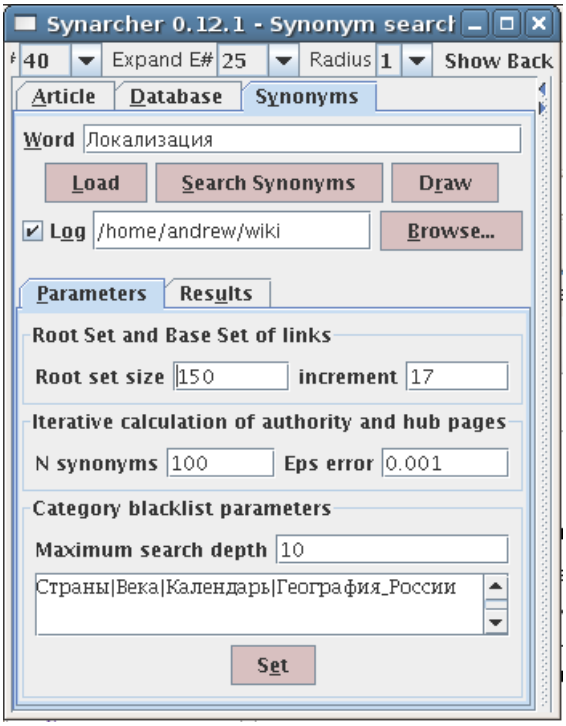


Рис. 14. Задание параметров адаптированного HITS алгоритма

В нижней части экрана (рис. 14) задаётся группа параметров, озаглавленных «*Category blacklist parameters*». Данные параметры определяют такие параметры эвристики[1], как: глубина поиска (поле «*Maximum search depth*») и чёрный список категорий, разделённых вертикальной чертой (на рисунке в этом поле введены категории «*Страны*|*Века*|*Календарь*|*География_Россия*»). После редактирования категорий для сохранения изменений достаточно нажать кнопку *Set*.

На рис. 15 показан пример результатов поиска семантически близких слов для слова *Локализация*. Итак, пользователь вводит слово и параметры алгоритма, нажимает кнопку «*Synonym Search*», после чего система выполняет поиск.

Экран программы разделён на две части вертикальной полосой, с левой стороны (вкладка: *Synonyms, Results*) представлены результаты поиска в виде таблицы и текста, с правой стороны – в виде графа. Вершины графа соответствуют названиям статей энциклопедии (статья – это гипертекстовая страница), рёбра указывают наличие гиперссылок между статьями. Пользователь может пометить найденные слова как синонимы либо с

1 См. подраздел «Эвристика: фильтрация на основе категорий статей» во второй главе.

помощью галочки в таблице (графа «*Rate it*»), либо с помощью контекстного меню, нажав правую клавишу мышки на интересующей вершине и выбрав команду «*Rate synonym*». Другие команды контекстного меню позволяют:

- раскрыть вершину, то есть отобразить соседние вершины (команда «*Expand Node*», см. (рис. 15);
- спрятать соседей, которые связаны только с этой вершиной (команда «*Collapse Node*»);
- спрятать вершину и тех соседей, которые связаны только с этой вершиной (команда «*Hide Node*»);
- спрятать всех соседей, кроме данной вершины (команда «*Hide All Except Node*»).

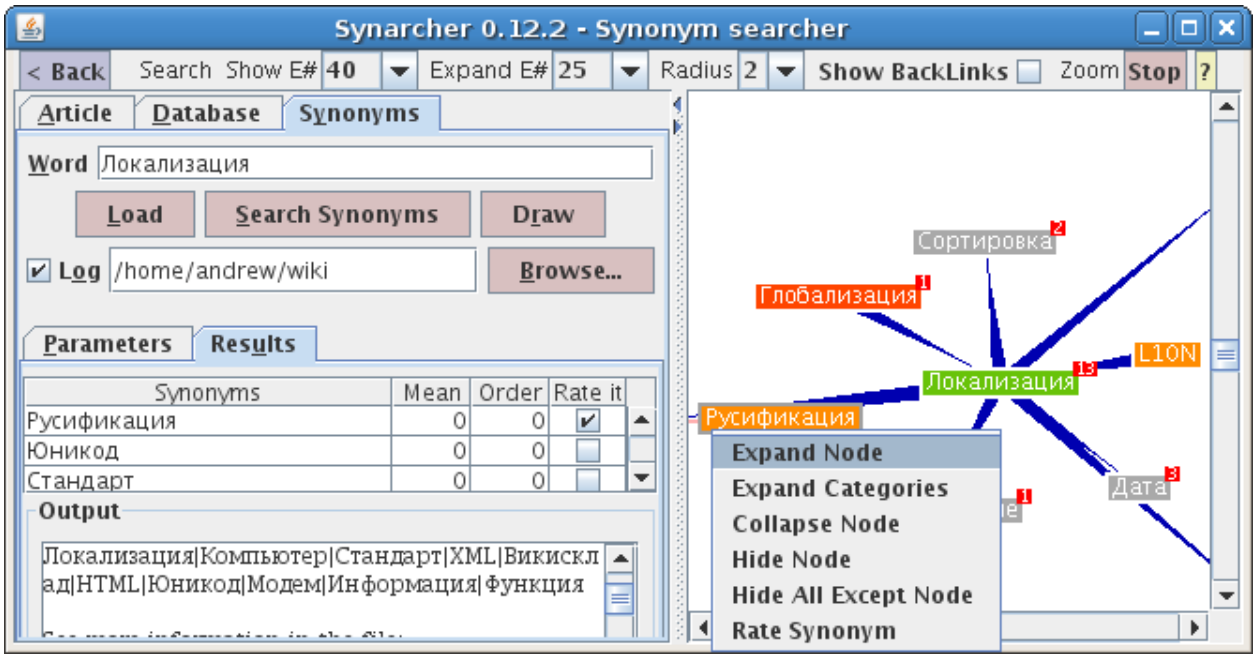


Рис. 15. Результаты поиска семантически близких слов для слова *Локализация* в виде таблицы, текста и графа. Пример контекстного меню справа-внизу

На рис. 15 можно выделить следующие группы вершин[1] (по именам статей, выделенных *курсивом*, им соответствующим):

1. *Локализация* – исходная вершина, заданная пользователем; с неё начинался поиск в адаптированном HITS алгоритме;
2. *Русификация, L10N* – вершины, помеченные пользователем, как синонимы исходного слова *Локализация*;

1 В программе разные группы вершин выделены разным цветом.

3. *Глобализация* – хаб-вершина, то есть вершина, ссылающаяся на многие авторитетные вершины (см. понятие авторитетных и хаб-страниц в разделе «Алгоритм HITS» (стр. 27) и разделе «Вычисление весов authority и hub» стр. 80);

4. Авторитетные страницы (отсутствуют на данном рисунке);

5. *Сортировка*, *Дата* – все остальные вершины (в адаптированном HITS алгоритме они соответствуют вершинам из базового набора).

Аналогичный скриншот, но уже с интерфейсом на русском языке, с результатами поиска для слова «*Интернационализация*» представлен на рис. 16.

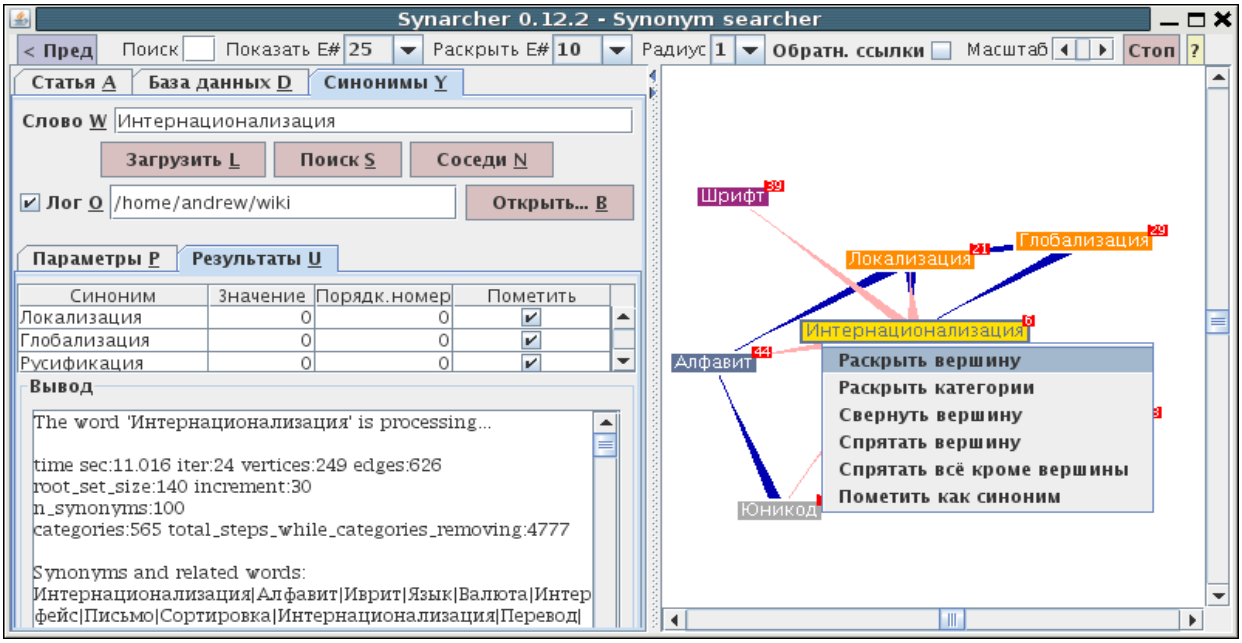


Рис. 16. Результаты поиска семантически близких слов для слова *Интернационализация* в виде таблицы, текста и графа. Пример контекстного меню справа-внизу

Одним из результатов поиска является список категорий, см. вкладку «*Категория C*» на рис. 17. Таблица (слева в центре) содержит список категорий (столбец *Категория*), упорядоченных по числу слов (столбец *Статей*), статьи которых принадлежат этим категориям. При выборе какой-либо категории в таблице обновляется текст – список статей – в окне *Output* (слева внизу). На рис. 17 выделена категория «Воздушные суда» и перечислены названия однословных статей с такой категорией.

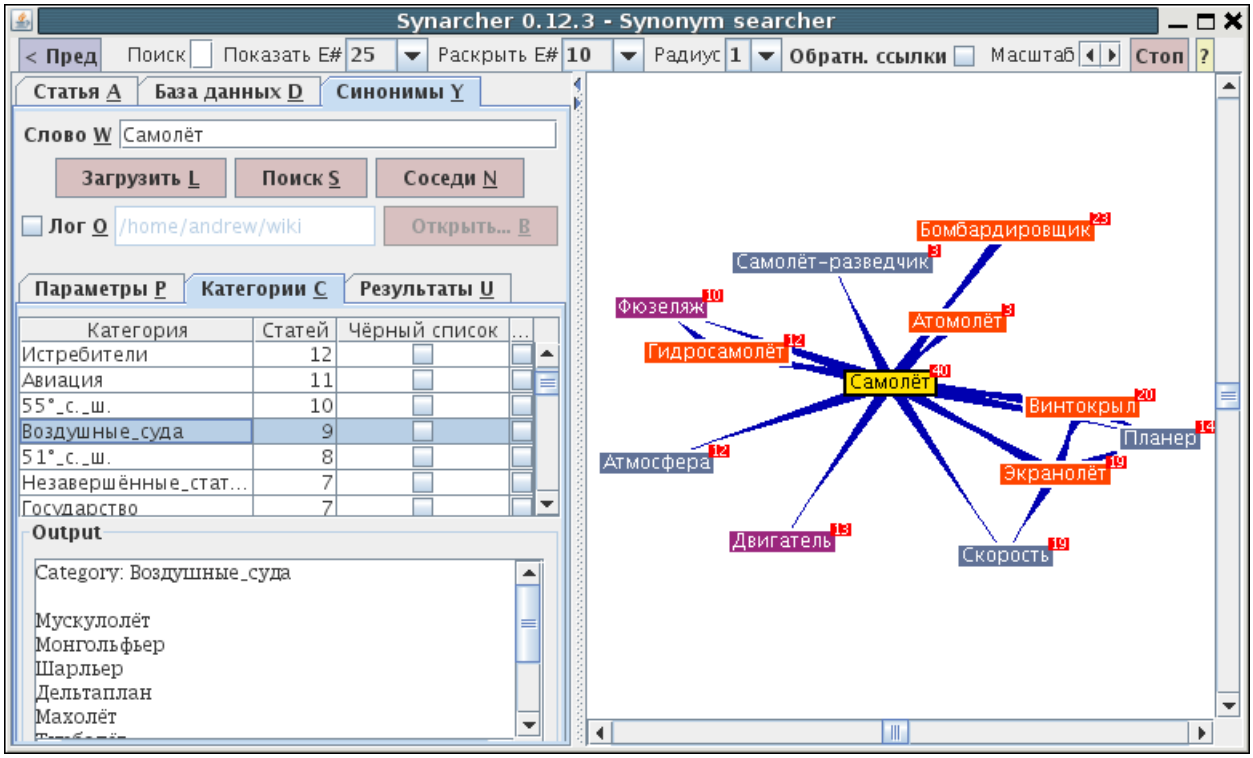


Рис. 17. Таблица со списком категорий (слева в центре), список статей для выделенной категории «Воздушные суда» (слева внизу)

3.2 Архитектура подсистемы GATE для удалённого доступа (на основе XML-RPC протокола) к программе морфологического анализа Lemmatizer

Сформулируем задачу, исходя из доступных ресурсов. Система GATE[1] написана на языке Java, поэтому может работать в среде Linux и Windows. Модуль морфологического анализа *Lemmatizer* написан на языке C++ и может работать либо в среде Linux, либо в CygWin (эмуляция Linux в среде Windows).

Связать эти разнородные системы целесообразно с помощью протокола XML-RPC (рис. 18). Данный протокол позволяет взаимодействовать программным компонентам, (1) написанным на разных языках, (2) расположенным на разных компьютерах.

Таким образом, необходимо расширить модуль морфологического анализа Lemmatizer функциональностью XML-RPC сервера, написанного на C++, назовём его LemServer, для вызова функций которого нужно указать:

1 Проблема отсутствия доступного модуля морфологической обработки русского языка в системе GATE (наподобие модуля Lemmatizer [60]) и сама система описаны выше (стр. 42).

- имя компьютера (указывается, как параметр модуля GATE – *hostLemServer*, см. рис. 27);
- порт сервера (*portLemServer*);
- словарь – русский, английский или немецкий (*dictLemServer*).

Со стороны GATE необходим модуль, назовём его *Russian POS Tagger*, написанный на языке Java, который с одной стороны обеспечивает стандартную функциональность модулей GATE (что даёт, в частности, возможность собирать модули GATE в последовательные цепочки обработчиков, по англ. *pipe*). С другой стороны *Russian POS Tagger* может вызывать функции *Lemmatizer* с помощью XML-RPC клиента, написанного на языке Java, назовём его *aotClient* (программные компоненты, разработанные автором диссертации, выделены чёрной линией на рис. 18).

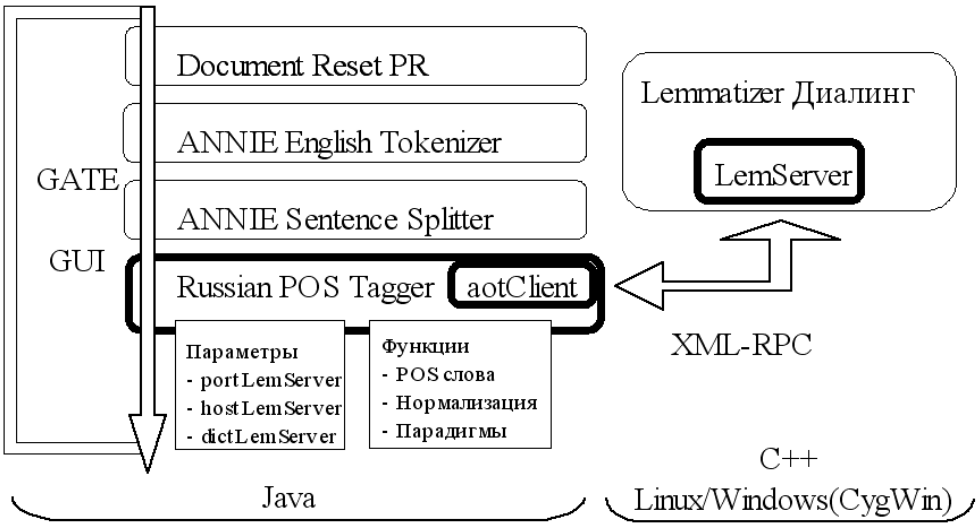


Рис. 18. Архитектура интеграции системы GATE и модуля морфологического анализа Lemmatizer

Таким образом, *Russian POS Tagger* взаимодействует с *Lemmatizer*, вызывая Java-функции *aotClient*, который выполняет XML-RPC запросы к *LemServer*, который, в свою очередь, вызывает C++ функции *Lemmatizer*. *LemServer* – это интерфейс на C++ для вызова функций модуля *Lemmatizer*.

Один их возможных списков модулей системы GATE, последовательно применяемых к тексту или корпусу текстов на русском, английском или немецком языке, включает:

- *Document Reset PR* – инициализация параметров текстов;
- *ANNIE English Tokenizer* – выделение элементов текста (токенизация)

- *ANNIE Sentence Splitter* – выделение предложений;
- *Russian POS Tagger* – морфологическая обработка слов.

Морфологическая обработка включает в себя (1) определение части речи, (2) приведение слова к начальной форме, (3) построение списка парадигм слова. Параметры модуля *Russian POS Tagger*, необходимые для подключения и вызова функций XML-RPC сервера LemServer включают: хост (*portLemServer*), порт (*hostLemServer*) сервера, который может быть запущен на другом компьютере. Поиск может выполняться для русского, английского или немецкого языков, для выбора языка нужно указать параметр *dictLemServer* модуля *Russian POS Tagger*.

3.3 Индексирование вики-текстов: архитектура системы и структура индексной базы данных

Разработанный адаптированный HITS алгоритм (AHITS) [37], [126] выполняет поиск на основе анализа внутренних ссылок Википедии. Многие алгоритмы поиска семантически близких слов (СБС) в Википедии обходятся без полнотекстового поиска (см. табл. 4.6 на стр. 130). Однако экспериментальное сравнение алгоритмов в работах [103], [35] показало, что наилучшие результаты поиска семантически близких слов даёт алгоритм Explicit Semantic Analysis (ESA) [103], использующий именно полнотекстовый поиск.

Это подвигнуло к созданию общедоступной индексной базы данных Википедии (далее WikIDF[1]) и программных средств для генерации БД, что в целом обеспечит полнотекстовый поиск в энциклопедии и в вики-текстах вообще. Вики-текст — это упрощённый язык разметки HTML.[2] Для индексирования требуется преобразовать его в текст на естественном языке. Поиск по ключевым словам не будет использовать символы и теги HTML- и вики-разметки, поэтому они будут удалены в ходе преобразования вики-текста в текст на естественном языке на предварительном этапе индексирования.

Разработанные программные ресурсы (база данных и система

1 WikIDF – аббревиатура, отражающая применение подхода TF-IDF [155], [52] к индексированию вики-текстов.

2 См. http://en.wikipedia.org/wiki/Wikitext.

индексирования) позволят учёным проанализировать полученные индексные БД википедий, а разработчикам поисковых систем воспользоваться уже существующей программой и обеспечить поиск по вики-ресурсам за счёт подключения к построенным индексным базам или генерации новых.

Архитектура системы построения индексной БД вики-текстов

Итак, спроектирована архитектура программной системы индексирования вики-текстов.[1] На рис. 19 показана организация взаимодействия программ GATE [92], Lemmatizer [60] и Synarcher [37], [126]. Полезным результатом работы системы будет индексная БД на уровне записей (англ. «*record level inverted index*»), содержащая список ссылок на документы для каждого слова (точнее — для каждой леммы).[2]

Программной системе требуется задать три группы входных параметров. Во-первых, язык текстов Википедии (один из 254 на 16/01/2008) и один из трёх языков (русский, английский, немецкий) для лемматизации, что определяется наличием трёх баз данных, доступных лемматизатору [60]. Указание языка Википедии необходимо для правильного преобразования текстов в вики-формате в тексты на ЕЯ (рис. 19, функция «Преобразование вики в текст» модуля «Обработчик Википедии»)[3]. Во-вторых, *адрес вики и индексной баз данных*, а именно обычные параметры для подключения к удалённой БД: IP-адрес, имя БД, имя и пароль пользователя. В-третьих, *параметры индексирования*, связанные с ограничениями, накладываемыми пользователем на размер индексной БД, предназначенной для последующего поиска по TF-IDF схеме.[4]

---

1 Фантазии по поводу применения данной архитектуры для (1) тематической вики-индексации и (2) фильтрации потоков текстов см. в работе [167].

2 Об инвертированном файле см. в [52], а также http://en.wikipedia.org/wiki/Inverted_index.

3 Более подробно о преобразование текста см. на стр. 138.

4 Например, ограничение числа связок *слово-страница*. В экспериментах ограничение 1000, см. табл. 4.12.

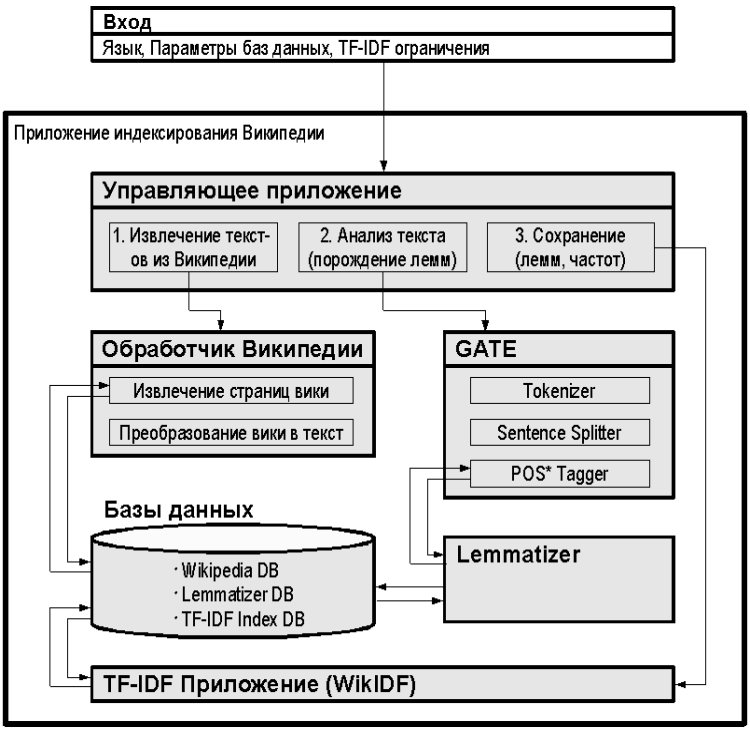


Рис. 19. Архитектура системы индексирования вики-текстов (POS — Part Of Speech)

«Управляющее приложение» выполняет последовательно три шага для каждой статьи (вики-документ), извлекаемой из БД Википедии и преобразуемой в текст на ЕЯ, что и составляет первый шаг. На втором шаге привлекается такой мощный инструмент как GATE [92] вкупе с отечественной разработкой Lemmatizer [60] и объединяющей их программной «прослойкой» RussianPOSTagger[1], конечная цель которых – получение списка лемм и их частоты встречаемости в данной статье.[2] На третьем шаге полученные данные сохраняются в индексную БД[3]: (1) полученные леммы, (2) частота их встречаемости в тексте, (3) указывается, что данные леммы принадлежат данному вики-тексту, (4) увеличивается значение частоты данных лемм в корпусе.

Следует отметить, что две функции модуля «Обработчик Википедии», указанные на рис. 19, а также API доступа к индексной БД («TF-IDF Index

1 Более подробно о программах *Lemmatizer* и *Russian POS Tagger* см.: http://rupostagger.sourceforge.net.

2 Точнее — суммарной частоты всех словоформ данной лексемы в заданной статье (и во всём корпусе) для каждой леммы. Лемма строится по словоформе с помощью программы Lemmatizer. Определение словоформы, леммы и лексемы см. в Русском Викисловаре.

3 Построенные таким способом индексные БД Русской Википедии и Simple Wikipedia доступны по адресу: http://rupostagger.sourceforge.net, см., соответственно, пакеты *idfruwiki* и *idfsimplewiki*.

DB» из модуля «TF-IDF Приложение») реализованы в программе Synarcher[1]. Указание входных параметров и запуск индексации осуществляются с помощью модуля Synarcher: WikIDF[2].

Таблицы и отношения в индексной БД

Принципы построения индексной БД:[3]

1. данные наполняются единожды и далее используются *только для чтения* (поэтому не рассматриваются такие вопросы, как: обновление индекса, добавление записи, поддержка целостности);
2. из викитекста удаляется вики- и HTML- разметка; выполняется лемматизация, леммы слов сохраняются в базу;
3. данные хранятся в несжатом виде.

Число таблиц в индексной базе данных, их наполнение и связи между ними были определены исходя из решаемой задачи: поиск текстов по заданному слову с помощью TF*IDF формулы (см. ниже). Для этого достаточно трёх[4] таблиц и ряда полей (Рис. 20):

1. *term* — таблица содержит леммы слов (поле *lemma*); число документов, содержащих словоформы данной лексемы (*doc_freq*); суммарная частота словоформ данной лексемы по всему корпусу (*corpus_freq*);
2. *page* — список названий проиндексированных документов (поле *page_title* в точности соответствует полю одноимённой таблицы в БД MediaWiki); число слов в документе (*word_count*);
3. *term_page* — таблица, связывающая леммы словоформ, найденных в документах с этими документами.

Постфикс «*_id*» в названии полей таблиц обозначает уникальный

---

1 Эта функциональность доступна начиная с версии *Synarcher 0.12.5*, см. http://synarcher.sourceforge.net

2 WikIDF — это консольное приложение на языке Java. Оно зависит от библиотек программы Synarcher и поставляется в комплекте с последней.

3 См. принципы проектирования индексов поисковых систем: http://en.wikipedia.org/wiki/Index_(search_engine).

4 Четвёртая таблица *related_page* нужна для вспомогательной функции — кэширования похожих страниц, найденных с помощью AHITS алгоритма [126]. Она использовалась при экспериментальной оценке работы AHITS, поскольку слова в тестовом наборе многократно повторялись [35].

идентификатор. Ниже горизонтальной полосы в рамке каждой таблицы перечислены поля, проиндексированные для ускорения поиска. Между полями таблиц задано отношение *один ко многим* — между таблицами *term* и *term_page*, а также *page* и *term_page*.

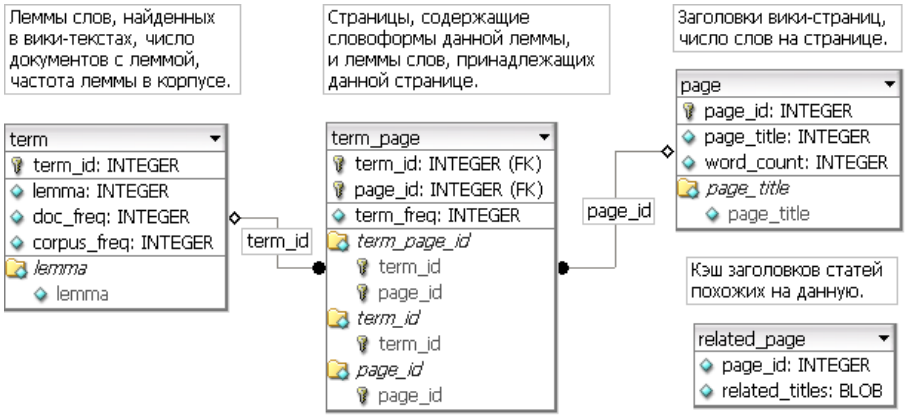


Рис. 20. Таблицы и отношения в индексной базе данных WikIDF[1]

Данная схема БД позволяет получить:

- список лемм слов заданного документа;[2]
- список документов, содержащих словоформы лексемы, заданной своей леммой.[3]

Напомним читателю формулу TF-IDF (3.1), с прицелом на которую и была спроектирована вышеуказанная схема БД (рис. 20). Всего в корпусе D документов, термин (лексема) t_i встречается в DF_i документах (поле индексной базы данных *term.doc_freq*). Для заданного термина t_i вес документа $w(t_i)$ определяется как [155]:

$$w(t_i) = TF_i \cdot idf(t_i) \quad ; \quad idf(t_i) = \log \frac{D}{DF_i} \qquad (3.1)$$

где TF_i — число вхождений термина t_i в документ (поле *term_page.term_freq*), *idf*[4] служит для уменьшения веса высоко частотных слов. Можно нормализовать TF_i, учтя длину документа, то есть разделив на число слов в документе (поле *page.word_count*). Таким образом, значения

1 Для проектирования и визуализации таблиц БД использовалась система визуального проектирования баз данных [17] DBDesigner 4, см. http://www.fabforce.net/dbdesigner4.

2 Точнее: возможно меньше число, чем все леммы. Поскольку для слов, встречающихся больше чем в N документах (здесь тысяча), N+1 связка «слово-документ» не будет записана в таблицу *term_page*.

3 То же ограничение, до 1000 документов здесь.

4 См. замечание об IDF на стр. 72.

полей БД позволяют вычислить обратную частоту термин t_i в корпусе.

Отметим, что в результате построения индексной БД оказалось, что размер индекса составляет 26-38% от размера файла с текстами индексируемого вики-проекта.[1]

3.4 Архитектура программной системы для автоматической оценки списков семантически близких слов

Разработана и далее описана архитектура программной системы для автоматической численной оценки набора списков семантически близких слов на основе тезаурусов английского языка (*WordNet* и *Moby*), доступным с помощью *Dict* серверов. Эти сервера предоставляют программный интерфейс для получения данных о конкретных словарных статьях. Достоинства *Dict* серверов для пользователя:

- это низкие требования к клиенту (не нужно устанавливать словарь локально, словари хранятся на сервере, клиенту нужен только выход в Интернет и программа клиент);
- это возможность регулярного обновления данных на сервере, без обременения пользователю необходимостью отслеживать выход новых версий словаря, самостоятельно его устанавливать.

Для оценки работы адаптированного HITS алгоритма используются формулы, разработанные во второй главе (см. раздел 2.5, стр. 91). Предложена следующая программная архитектура для автоматической оценки (рис. 21).

1 Цифры 26-38% получены как отношение значения поля «Исходный дамп, размер» к «Размер сжатого файла дампа индексной БД» в табл. 4.12 на стр. 144.

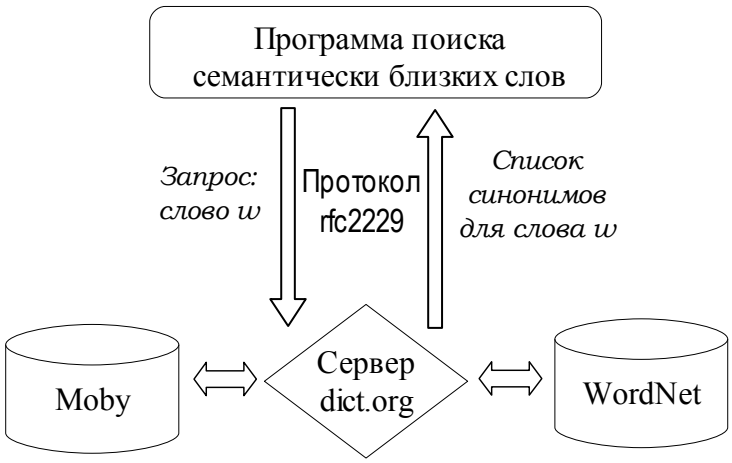


Рис. 21. Архитектура программной системы для автоматической оценки списков семантически близких слов

На основе протокола *rfc2229* система запрашивает список синонимов (из тезауруса *WordNet*) и список семантически близких слов (тезаурус *Moby*) и после этого сравнивает списки, построенные программой поиска семантически близких слов, с эталонными (на основе разработанных формул). Данная архитектура была реализована в качестве прототипа как один из модулей программной системы Synarcher. Связь этого модуля с пользователем через графический интерфейс на данный момент не реализована.

Выводы по главе 3

В данной главе были представлены: (1) архитектура программы Synarcher, реализующей адаптированный HITS алгоритм, (2) модель интеграции системы GATE и модуля морфологического анализа *Lemmatizer* (на основе разработанных автором XML-RPC клиента и сервера), (3) архитектура системы построения индексной базы данных (БД) вики-текстов вместе со структурой таблиц индексной БД и (4) архитектура программной системы оценивания списков семантически близких слов с помощью удалённого доступа к тезаурусам посредством *Dict* сервера.

В рамкам описания архитектуры программы Synarcher представлены основные классы и методы программы, программный интерфейс доступа к Википедии и особенности реализации модуля визуализации.

В рамкам описания архитектуры программы Synarcher представлены основные классы программы и их методы, программный интерфейс доступа к Википедии и особенности реализации модуля визуализации. Программа (1) предоставляет доступ к Википедии, хранимой в базе данных MySQL, размещённой локально или удалённо, (2) позволяет задать параметры адаптированного HITS алгоритма, (3) обеспечивает хранение параметров поиска и слов, помеченных пользователем, как синонимы, на компьютере пользователя.

Модуль визуализации написан на основе кода программы визуализации вики-страниц – *TouchGraph WikiBrowser*. Для более удобной навигации код программы был существенно модифицирован, в контекстное меню были добавлены команды: спрятать все вершины (*Hide all except node*), пометить вершину как синоним (*Rate synonym*), показать категории (*Expand Categories*).

Описаны основные экраны программы, точнее вкладки[1]: (1) вкладка *Article*, позволяющая просмотреть энциклопедическую статью, соответствующую выбранному слову, (2) вкладка *Database*, позволяющая подключиться к базе данных и получить статистику по базе данных,

1 Вкладка – элемент графического интерфейса для переключения между приложения, в данном случае между разными группами входных параметров и результатами. См. http://ru.wikipedia.org/wiki/Вкладка.

(3) вкладке *Synonyms*, на которой задаются параметры адаптированного HITS алгоритма, выводятся результаты поиска в табличной и текстовой форме.

Описан экран, на котором представлен результат поиска семантически близких слов в виде графа. Описаны команды контекстного меню, позволяющие работать с графом. Указаны (с пояснениями) группы вершин, составляющих этот граф.

В этой главе описана модель, позволяющая интегрировать модуль морфологического анализа Lemmatizer[1] в систему GATE. Данная модель представляет способ интеграции приложений написанных на разных языках программирования (например, C++ и Java) посредством XML-RPC протокола.

Описано назначение модулей разработанных автором: (1) GATE модуль – *Russian POS Tagger*, (2) XML-RPC клиент на Java – *aotClient* и (3) XML-RPC сервер на C++ – *LemServer*.

Приведён один их возможных списков модулей системы GATE, включающий модуль *Russian POS Tagger*. Описаны параметры модуля *Russian POS Tagger* для его включения в систему GATE.

В главе спроектирована архитектура системы построения индексной БД вики-текстов. Описаны таблицы и отношения в индексной БД, строимой данной системой.

В данной главе представлена архитектура программной системы оценивания синонимов, позволяющей реализовать метод численной оценки списков семантически близких слов на основе тезаурусов английского языка (WordNet и Moby). Данная система для доступа к тезаурусам использует *Dict* сервера. Указаны достоинства *Dict* серверов для конечного пользователя

1 *Lemmatizer* разработан москвичами (в проекте Диалинг), см. http://www.aot.ru

4. Эксперименты и практическое использование разработанных в диссертации алгоритмов

В этой главе приводятся примеры результатов работы адаптированного HITS алгоритма и сравнение результатов работы с другими алгоритмами. Выполняется оценка алгоритма с помощью коэффициента Спирмена.

Показана работа модуля Russian POS Tagger в составе системы GATE. В качестве проверки работоспособности программного комплекса индексирования вики-текстов построен ряд индексных баз данных, приводится сравнение построенных баз данных по ряду параметров.

Работоспособность и функциональность разработанных программных комплексов (Synarcher и WikIDF) обосновывается успешно работающими unit-модулями (о методике экстремального программирования, а именно автоматических тестовых модулях см. в [6], [9], [188]).

4.1 Экспериментальная оценка работы адаптированного HITS алгоритма

Оценка тестируемого корпуса текстов

Разработанная система тестировалась на двух корпусах: Английская и Русская Википедия. Энциклопедии хранятся в базе данных MySQL. На скорости работы реализованной системы Synarcher сказываются такие параметры энциклопедии, как: число статей, число ссылок, число категорий.

Сервер Википедия (http://en.wikipedia.org) не использовался, поскольку данная реализация поиска синонимов требует значительной вычислительной нагрузки на базу данных (БД). Поэтому обрабатывалась локально установленная БД MySQL Википедия.

Параметры компьютера на котором выполнялись эксперименты: процессор – AMD Athlon XP 2700+, оперативная память – 1 Гб, винчестер – 80 Гб, операционная система – Debian Sarge 3.1.

Английская версия содержит 901 861 энциклопедических статей, 18.3 млн. внутренних перекрёстных ссылок и 1.2 млн. ссылок на категории.

Тестируемая Английская Википедия соответствует онлайн версии от 8 марта 2005 г.[1]

Указание версий дампов Русской Википедии и Simple Wikipedia, использованных для построения индексных БД, даётся в разделе, посвящённом индексированию на стр. 143.

Эксперименты с Английской Википедией

Нужно отметить, что поиск синонимов и семантически близких слов не является полностью автоматическим[2]. Программа Synarcher формирует список слов, который является сырьём для дальнейшей работы эксперта. В построенный автоматически список могут попасть слова весьма далёкие от семантически близких слов, программу можно рассматривать в качестве некоторого фильтра.

Поиск СБС с помощью программы Synarcher выглядит следующим образом. Пользователь задаёт исходное слово, задаёт параметры адаптированного HITS алгоритма. Программа строит список слов, который может содержать семантически близкие слова, и представляет список в виде таблицы и графа пользователю. Используя команды навигации (см. раздел 3.1), пользователь исследует граф и помечает слова (на графе и в таблице), которые, по его мнению, являются семантически близкими исходному слову. Эта информация сохраняется на компьютере пользователя. При повторном поиске эти данные будут учитываться (см описание подхода на стр. 66).

Таким способом автором, с помощью программы Synarcher, были найдены синонимы для слов *Robot* и *Astronaut* (колонка *Synarcher+Эксперт* в таблицах 4.1 и 4.2 соответственно). Итого было найдено 6 синонимов для слова *Robot*, отсутствующих в WordNet: *Android*, *Homunculus*, *Domotics*,

1 Отметим, что для работы адаптированного HITS алгоритма (реализованного в программе Synarcher) необходимо, чтобы таблица *pagelinks* БД Википедия была заполнена. Таблица *pagelinks* хранит информацию о том, какая страница на какую ссылается. Авторы оболочки энциклопедии MediaWiki предлагают несколько способов её заполнения. До 2006 г. эту таблицу вполне успешно заполнял инструмент mwdumper (http://download.wikimedia.org/tools/), написанный на Java. После изменения формата БД Википедии осталось два способа заполнения: с помощью php-скрипта *refreshLinks.php* и (более быстрый способ) с помощью программы Xml2sql (http://meta.wikimedia.org/wiki/Xml2sql).

2 Точно также как результаты поиска любой ИПС могут содержать документы не нужные пользователю, но найденные в виду особенностей исходного набора данных и алгоритма ИПС.

Replicant, *Sentience*, *Parahumans*. Здесь тезаурус Moby не рассматривается, так как он не содержит слово *Robot*.

Для слова *Astronaut* с помощью программы Synarcher было найдено 4 синонима, из которых три отсутствуют в тезаурусах Moby и WordNet: «*Space tourist*», «*Spationaut*» и «*Taikonaut*» (табл. 4.2).

Таблица 4.1

Синонимы для слова *Robot*

Синонимы	***Synarcher+Эксперт***	***WordNet 2.0***
Android[1]	+	
Automaton		+
Golem	+	+
Homunculus	+	
Domotics	+	
Replicant	+	
Sentience	+	
Parahumans	+	

Таблица 4.2

Синонимы для слова *Astronaut*

Синонимы	***Synarcher+Эксперт***	***WordNet 2.0***	***Moby***
Aeronaut			+
Cosmonaut	+	+	+
Pilot			+
Rocket man			+
Rocketeer			+
Space tourist	+		
Spaceman		+	+
Spationaut	+		
Taikonaut	+		

Результаты экспериментов показывают, что с помощью программы Synarcher можно найти синонимы и семантически близкие слова, отсутствующие в современных тезаурусах (например, найден синоним *Spationaut* для слова

1 Значение данного слова см. в энциклопедической статье http://en.wikipedia.org/wiki/Android. Значение прочих слов таблицы можно найти в энциклопедии аналогичным образом.

Astronaut). Тем не менее, некоторые синонимы,представленные в тезаурусах, не были найдены. Например, синоним *Automaton* для слова *Robot* не был найден, хотя такая статья[1] в Википедии существует. Это можно объяснить несовершенством алгоритма и большим количеством статей (901.8 тыс.) среди которых выполнялся поиск.

Эксперименты с Русской Википедией

Русская версия энциклопедии содержит 94 632 энциклопедических статей, 3.4 млн. внутренних перекрёстных ссылок, 29.9 тыс. категорий, 390.1 тыс. ссылок на категории. Тестируемая русская Википедия в данном эксперименте соответствует онлайн версии от 18 июля 2006 г.

Для оценки работы адаптированного HITS алгоритма были выбраны четыре слова, имеющие статьи в Русской Википедии:

1. Слово *жаргон*, имеющее несколько синонимов[2] (*сленг, арго, радиожаргон, феня*)[3];
2. Слово *самолёт*, для которого существуют слова для названий предметов близких по сути (*планер, турболёт*).
3. Многозначное слово *сюжет*. Слово может обозначать[4] (1) содержание, суть случая, происшествия, фильма, рассказа о чём-нибудь, тогда синонимами будут *содержание, киносюжет*, (2) совокупность действий, событий, в которых раскрывается основное содержание художественного произведения, тогда синонимами будут *фабула*, *интрига*.
4. Слово *истина*, обозначающее понятие, не имеющее однозначного определения.

Для этих слов был выполнен поиск семантически близких слов с помощью программы Synarcher. Полный список слов, выданный программой см. в приложении (табл. 1). С помощью эксперта были отобран ряд слов наиболее

1 См. http://en.wikipedia.org/wiki/Automaton

2 Словарь синонимов системы ASIS, ссылка (http://www.lingvoda.ru/dictionaries/)

3 Второе значение слова *жаргон* (дорогой камень красно-желтого цвета, циркон, минерал [24]) на данный момент (18 июля 2006 г.) в русской Википедии не представлено

4 Толковый словарь русского языка (1935-1940 гг.) под редакцией Д.Н.Ушакова. Компьютерное издание, ссылка (http://www.lingvoda.ru/dictionaries/)

близких по значению к искомому слову (табл. 4.3). Порядок слов в графе «*Семантически близкие слова»* в этой таблице является существенным для вычисления коэффициента Спирмена. Данный порядок был получен после опроса 16 респондентов (носителей русского языка) и представляет собой ряд упорядоченный на основе усреднённой оценки экспертов (см. об усреднении в приложении в табл. 1).

Если не указано особо, то здесь и далее поиск выполнялся при следующих параметрах адаптированного HITS алгоритма:

- размер корневого набора:200;
- инкремент: 17;
- чёрный список категорий:Страны|Века|Календарь|География_России|Люди;
- глубина поиска категорий: 10;
- ограничение сверху длины строящегося списка слов:100;
- погрешность для останова итераций: 0.01.

Идея выбора нескольких типов слов (для оценки алгоритма) и привлечения респондентов для упорядочения списков семантически близких слов взята из работы [83].

Таблица 4.3

Семантически близкие слова, полученные экспертом с помощью программы Synarcher и упорядоченные респондентами

Слово	***Семантически близкие слова***
Жаргон	Сленг, Просторечие, Матерщина, Диалект, Арго[1], Эвфемизм
Истина	Факт, Правда, Реальность, Действительность, Знание, Бог, Вера, Авторитет, Догмат
Самолёт[2]	Планер, Турболёт, Автожир, Экранолёт, Экраноплан, Авиация, Транспорт, Штурмовик, Махолёт, Мускулолёт, Дельтаплан, Вертолёт, Винтокрыл,

1 Определение данного слова даётся в энциклопедической статье http://ru.wikipedia.org/wiki/Арго. Определения прочих слов данной таблицы см. аналогичным образом.

2 Викисловарь содержал (на 19.07.2007) такие семантически близкие слова для слова *самолёт* (включая синонимы, гипонимы, гиперонимы, меронимы, холонимы): *аэроплан, авиация, транспорт, штурмовик, экранолёт, экраноплан, моноплан, биплан, планер, махолёт, мускулолёт, дельтаплан, параплан, турболёт, вертолёт, автожир, винтокрыл, эскадрилья, авиапушка, фюзеляж, крыло, двигатель, винт*. Примеры СБС для словосочетания «Беспилотный летательный аппарат» представлены в табл. 4.1.

	Авиапушка, Фюзеляж, Двигатель, Винт
Сюжет	Интрига, Переживание, Конфликт, Трагедия, Коллизия, Противоречие

Экспериментальное сравнение адаптированного с исходным HITS алгоритмом[1]

Для десяти слов и словосочетаний, для которых есть энциклопедические статьи в Русской Википедии (*Автопилот, Аэродром, Беспилотный летательный аппарат, Движитель, Интернационализация, Истина, Пропеллер, Самолёт, Сюжет, Турбина*)[2], проведена серия экспериментов (рис. 22-25) для оценки времени работы и точности поиска адаптированного HITS алгоритма (AHITS) в зависимости от числа категорий (ось абсцисс).

Точность (англ. *precision*) – это отношение числа семантически близких слов, найденных программой, к общему числу найденных программой слов. Семантически близкие слова выбираются экспертом из общего числа слов, найденных программой. Примеры СБС для словосочетания «*Беспилотный летательный аппарат»* представлены в табл. 4.1.

Таблица 4.4

Список семантически близких слов для словосочетания *Беспилотный летательный аппарат*

Синонимы	БПЛА, БЛА
Гиперонимы	летательный аппарат
Гипонимы	спутник, зонд, ракета, автоматическая межпланетная станция
Меронимы	автопилот

Чёрный список категорий (*blacklist*) составляется экспертом и сужает пространство поиска. Например, включение категории «*XX век*» в blacklist

1 Эксперимент проводился на данных, соответствующих онлайн версии Русской Википедии от 20 сентября 2007 г. Данная версия энциклопедии содержит около двухсот тыс. энциклопедических статей, 10.4 млн. внутренних перекрёстных ссылок, 49.8 тыс. категорий, 1.1 млн. ссылок на категории.

2 Для слова *Жаргон* эксперимент не проводился, так как набор слов, найденный программой, слишком мал (11 слов) для того, чтобы применять какую-либо дополнительную фильтрацию. Для слова *Автопилот* точность поиска была низкой (2%) и не менялась при изменении числа фильтруемых категорий. Возможно, это объясняется недостаточным (по сравнению, например, со статьёй *Самолёт*) числом ссылок, связывающих статью *Автопилот* с другими. Результаты поиска СБС для слова *Автопилот* учитывались для оценки времени поиска и не учитывались для оценки суммарной точности поиска.

позволяет отсечь множество документов с заголовками: *1900*, *1901*, *1902* и т. д. В эксперименте для фильтрации выбираются категории с максимальным числом слов, не являющихся семантически близкими заданному слову[1]. На рис. 22 представлены не сами категории, а только их число (здесь от 0 до 15). Число *0* означает, что нет фильтрации категорий.

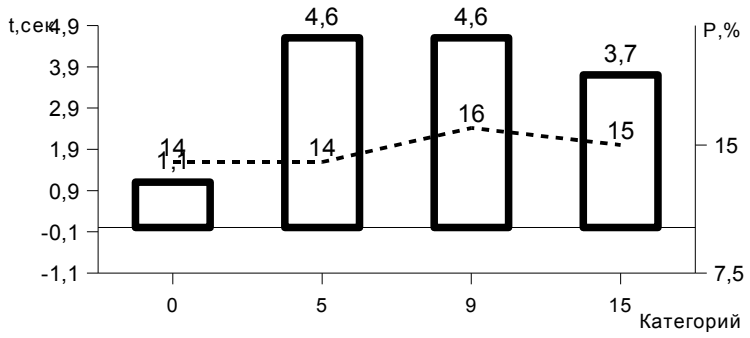


Рис. 22. Изменение времени работы (*t*) и точности поиска (*P*) (пунктирная линия) AHITS алгоритма в зависимости от числа фильтруемых категорий для слова *Истина*

Опишем детально эксперимент и дадим его интерпретацию для слова *Самолёт* (рис. 23). Время работы на рисунках указано с помощью высоты прямоугольника, точность поиска представлена с помощью тонкой пунктирной линии.

1. Категории. Проведено шесть опытов с разным числом категорий: 0, 1, 3, 6, 10, 17. Были выбраны категории с максимальным числом слов, не являющихся релевантными. Такие категории позволяют отсечь большое число статей, заведомо не относящихся к делу. Первая фильтруемая категория (для слова *Самолёт*) называется *Википедия:Избранные_статьи* на неё ссылается 14 найденных слов. Три категории включают в себя вышеуказанную, а также: *Незавершённые_статьи_по_географии*|*Незавершённые_статьи*. Шесть категорий включают (помимо трёх, ещё три) *Химические_элементы*|*Государство*|*Википедия:Статьи_к_викификации*. 10 категорий включают ещё четыре категории *Механика*|*Столицы_Летних_олимпиад*|*Мегаполисы*|*История_Европы*. 17 категорий включают ещё семь *Тюркские_народы*|*Город*|*Локомотивы*|*Города-*

1 Выбирается категория, которой принадлежит больше всего найденных слов. Это возможно узнать с помощью вкладки «Категории С», см. рис. 17 на стр. 106.

государства|Народы_России|Википедия:Хорошие_статьи_о_технике|Дворянство.

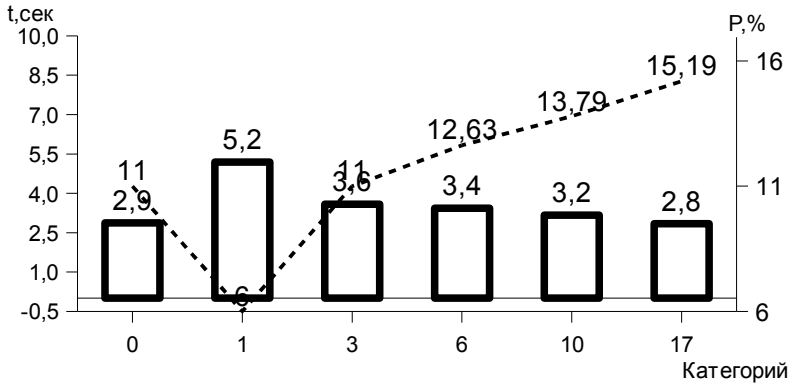


Рис. 23. Изменение времени работы (*t*) и точности поиска (*P*) AHITS алгоритма в зависимости от числа фильтруемых категорий для слова *Самолёт*

Русской Википедии соответствует орграф, содержащий 171 тыс. вершин, 3.4 млн дуг (на 11.05.07). При поиске в графе AHITS алгоритм строит базовый набор с числом вершин 200-800, числом дуг 800-12 000 (для слова *Самолёт*). Указан диапазон вершин и дуг, поскольку изменение фильтруемых категорий меняет число вершин, включаемых в базовый набор. Таким образом, рис. 23 обобщает результаты шести опытов с разными размерами базовых наборов, построенных для слова *Самолёт*.

2. Время работы. При нуле категорий получено почти минимальное время работы алгоритма (2.9 с). Этого следовало ожидать, так как фильтрация категорий требует дополнительных вычислений, то есть времени.

В опытах при увеличении числа категорий (при числе категорий больше нуля) время поиска уменьшается. В данном опыте время снизилось с 5.2 с (максимальное значение при одной категории) до 2.8 с (при 17 категориях). Это объясняется тем, что при увеличении числа фильтруемых категорий пространство поиска сужается. Тогда максимальное время работы алгоритма будет при минимальном числе категорий, то есть при фильтрации по одной категории[1], см. рис. 22-25.

1 Время поиска зависит и от того, какая именно категория выбрана.

3. Точность поиска. На рис. 23 видно, что использование категорий увеличивает точность поиска. Максимальная точность 15.2% получена при 17 категориях, минимальная точность 6% – при одной категориях. В среднем (по пяти опытам с числом категорий: 1, 3, 6, 10, 17) это превышает точность 11%, полученную в случае, когда категории не учитываются на 6.5%.

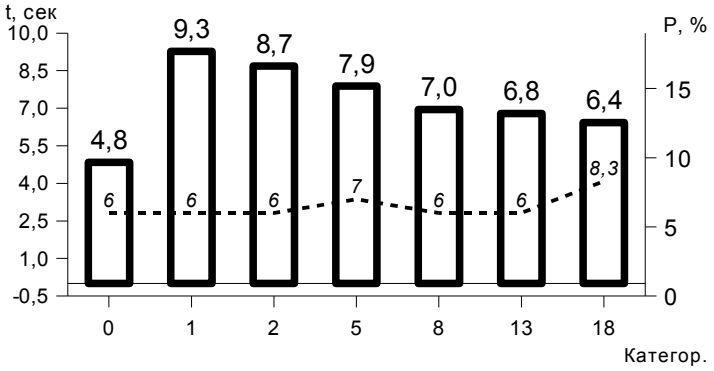


Рис. 24. Изменение времени работы (*t*) и точности поиска (*P*) AHITS алгоритма в зависимости от числа фильтруемых категорий для слова *Сюжет*

Основная разница HITS и AHITS алгоритмов – не учёт и учёт категорий соответственно. При числе категорий ноль (первый вертикальный ряд на рис. 22-25) можно считать, что работа AHITS алгоритма (по скорости и точности поиска) соответствует работе HITS алгоритма. Это позволяет сравнить HITS и AHITS алгоритмы в следующей таблице 4.5.

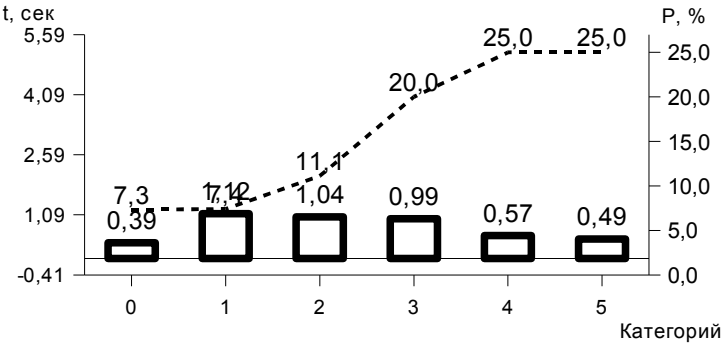


Рис. 25. Изменение времени работы (*t*) и точности поиска (*P*) AHITS алгоритма в зависимости от числа фильтруемых категорий для слова *Интернационализация*

В столбце *HITS* (табл. 4.5) указано время работы алгоритма AHITS без учёта категорий. В столбце *AHITS* дано среднее время работы алгоритма при числе

категорий больше нуля. Значения столбца «*Замедление работы, %*» вычислялось по формуле (AHITS – HITS)/AHITS·100%. Таким образом, усреднение по девяти словам показало, что адаптированный HITS алгоритм работает на 52% медленнее HITS алгоритма.

Таблица 4.5

Сравнение времени работы HITS алгоритма и адаптированного HITS алгоритмов

Слово	***HITS, с***	***AHITS, с***	***Замедление работы, %***
Аэродром	1,19	2,29	48,03
Беспилотный летательный аппарат	0,54	1,1	50,91
Движитель	2,17	5,39	59,74
Интернационализация	0,39	0,84	53,57
Истина	1,11	4,3	74,19
Пропеллер	0,94	2,85	67,05
Самолёт	2,87	3,64	21,15
Сюжет	4,85	7,67	36,77
Турбина	1,63	3,9	58,21
Среднее:	**1,74**	**3,55**	**52,18**

При каждом числе категорий получено своё значение точности (в AHITS алгоритме), поэтому можно указать минимальное (*Min*), среднее (*Avg*) и максимальное (*Max*) значение точности (P_{AHITS}) для каждого слова. Изменение точности вычисляется по формуле $(P_{AHITS} - P_{HITS}) / P_{HITS}$, ·100%, при этом было вычислено изменение точности в худшем (при минимальном значении точности алгоритма AHITS), среднем и лучшем случаях. Таким образом, точность поиска адаптированного HITS алгоритма выше точности поиска HITS алгоритма в худшем на -6.3%, среднем – на 33.3% и в лучшем – на 77.8%. Изменение точности поиска в зависимости от числа фильтруемых категорий представлено на рис. 26.

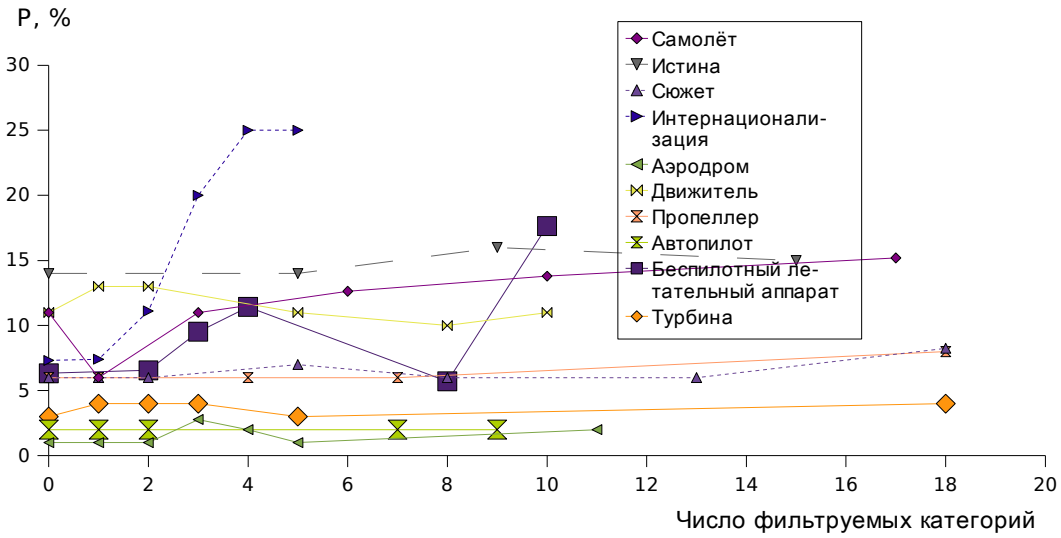


Рис. 26. Изменение точности поиска (*P*) AHITS алгоритма в зависимости от числа фильтруемых категорий

Сравнение результатов работы AHITS алгоритма с другими на основе 353 пар английских слов[1]

Для оценки метрик и алгоритмов, вычисляющих близость значений слов, был использован тестовый набор (англ. *Test Collection*) из 353 пар английских слов, предложенный в работе [100] (далее 353-TC).[2]

Респонденты (13 человек обработали 153 слова, 16 человек – 200 слов) присвоили значения от 0 до 10 семантической близости парам слов, где 0 указывает на то, что слова совершенно не связаны, 10 – слова почти полные синонимы. Критика данного тестового набора, приведённая в работе [117], заключается в том, что:

- не приведена методология составления списка;
- респондентам сложнее давать оценку от 0 до 10, чем на более привычной шкале от 0 до 4.

Достоинство данного тестового набора в том, что он

- превосходит другие тестовые наборы по размеру[3];

1 Эксперимент (по оценке работы AHITS алгоритма и адаптированной метрики Резника (res_{hypo}) проводился на данных, соответствующих онлайн версии English Wikipedia от 27 мая 2007 и 2 мая 2006г., а также Simple Wikipedia от 11 августа 2007 и 9 сенября 2007, подробности см. в [35].

2 Данные доступны: http://www.cs.technion.ac.il/~gabr/resources/data/wordsim353/wordsim353.html

– позволяет оценивать семантическую близость (включающую, например, отношение антонимии), а не семантическое сходство (только синонимия)[1].

Подробное описание экспериментов по оценке результатов поиска СБС в Английской и Simple[2] ВП приведено в работе [35], [36]. Основные результаты данной подглавы связаны с классификацией методов поиска СБС и оценкой методов.

Классификация метрик и алгоритмов поиска СБС, предложенная в [173], расширена (1) адаптированным HITS алгоритмом, основанном на анализе веб-ссылок, (2) алгоритмом WLVM [134] и (3) явным указанием отдельной группы методов, полагающихся на частотность слов в корпусе.[3] Таким образом, предложена следующая классификация (табл. 4.6) метрик и алгоритмов поиска СБС, основанных на учёте (i) расстояния в таксономии, (ii) анализа веб-ссылок, (iii) частотности слов в корпусе, (iv) совпадения (перекрытия) текстов. Следует уточнить, что метрика Резника *res* учитывает одновременно и частотность слов и свойства (не расстояние) концептов в таксономии.

Проведены эксперименты по вычислению корреляции результатов работы алгоритма AHITS и метрики Резника, адаптированной в работе [173] к Википедии, res_{hypo} с оценкой семантической близости пар английских слов, выполненной респондентами. Результаты указаны в столбцах *AHITS* и res_{hypo} в табл. 4.6, то есть курсивом выделены значения, рассчитанные самостоятельно. Данные для других метрик и алгоритмов в основном взяты из работы [173], в ней также описаны метрики *jaccard*, *text,* res_{hypo}. Использованы экспериментальные данные таких работ, как [117] (*jarmasz*), [100] (поисковик IntelliZap и алгоритм *LSA*), [103] (алгоритм *ESA*), [134] (алгоритмом WLVM). Представление об остальных метриках можно

3 Тесты на синонимию: 80 вопросов теста TOEFL, 50 вопросов ESL [177] и 300 вопросов Reader's Digest Word Power Game [117].

1 Разница понятий semantic similarity и semantic relatedness описана на стр. 185. AHITS алгоритм позволяет находить семантически близкие слова (*semantic relatedness*).

2 Простая Английская Википедия, см. http://simple.wikipedia.org

3 Ещё одна классификация метрик семантической близости представлена в работе [148] (стр. 5).

получить из работ: [186] метрика *wup*, ([99], стр. 265-283) метрика *lch*, [151] метрика *res*, [74] метрика *lesk*.

Табл. 4.6 содержит значения корреляции тестовой коллекцией 353-TC и результатов, полученных с помощью указанных метрик и алгоритмов. Получены лучшие результаты при поиске с учётом:

- *расстояния в таксономии* – 0.48, метрика *lch* ([99], стр. 265-283) для Английской Википедии;
- *анализ ссылок* – 0.45, алгоритм *WLVM* [134] для Английской Википедии (при автоматическом разрешении «дизамбигов»);
- *частотности слов в корпусе* – 0.75, алгоритм *ESA* [103] для Английской Википедии;
- *перекрытия текстов – 0.21,* метрика *lesk* [74] для тезауруса WordNet.

Вне рассмотрения оставлен алгоритм Green [145] (поиск в Википедии), поскольку нет данных о его тестировании с помощью коллекции 353-TC.

Таблица 4.6

Классификация алгоритмов и корреляция результатов с данными респондентов (на данных тестового набора 353-TC, без пропусков)

	Расстояние в таксономии				Анализ ссылок		Частотность слов в корпусе				Перекрытие текстов	
Набор данных	**wup**	**lch**	**res $_{hypo}$**	**jarm asz**	**AHITS**	**WLVM**	**jaccard**	**res**	**LSA**	**ESA**	**lesk**	**text**
WordNet	0.3	0.34	–	–	–		–	0.34		–	0.21	–
Wiki-pedia[1]	0.47	**0.48**	*0.33-0.36* 0.37[2]	–	*0.38-0.39*	0.45-0.72[3]	–	–[4]		**0.75**	0.2	0.19
Simple Wikipedia	–	–	*0.37*	–	*0.31-0.33*		–	–		–	–	–
Другие	–	–	–	Тезаурус Роже 0.539[5]	–		Google 0.18	–	Intelli Zap 0.56	–	–	–

Таким образом, оценка корреляции результатов поиска СБС с тестовым набором 353-TC показала, что алгоритм AHITS даёт несколько лучший результат (0.38-0.39), чем адаптированная метрика Резника (0.33-0.36) и хуже, чем алгоритм WLVM (0.45-0.72) на данных Английской Википедии. В экспериментах с Википедией на английском упрощённом языке получен значительный разброс значений корреляции для AHITS: от 0.15 до 0.4.[6]

Для оценки поисковых алгоритмов на русских словах предложен общедоступный тестовый набор.[7]

1 Английская Википедия, см. http://en.wikipedia.org

2 0.33-0.36 – о получении этих данных см. [35], [36], 0.37 взято из [173].

3 Коэффициент корреляции Спирмена с эталонным набором равен 0.45 при автоматическом разрешении многозначных статей и 0.72 — при ручном. Подробнее об WLVM см. на стр. 30.

4 Сомнения по поводу того, чтобы считать эквивалентными метрики *res* [151] и *res $_{hypo}$* [173] изложены в работе [35] на стр. 3.

5 0.539, см. [117], стр. 4. Значение 0.55 в работе [103] - это, вероятно, опечатка.

6 Подробное описание экспериментов см. в работе [35], (10 стр.), краткое в [36] (4 стр.).

7 См. http://ru.wikipedia.org/wiki/Участник:AKA_MBG/Wordsim

Пример оценки эвристики с помощью коэффициента Спирмена[1]

Одна из эвристик поиска похожих статей энциклопедии Википедия программой Synarcher заключается в том, чтобы пропускать и не включать в корневой и в базовый набор те энциклопедические статьи, названия которых содержат пробелы, то есть названия, состоящие более чем из одного слова. Для оценки эффекта этой эвристики на качество поиска был использован коэффициент Спирмена (табл. 4.7).

В этой таблице столбец *F* – это значение коэффициента Спирмена. Данный коэффициент получается при сравнении списка построенного программой (длина этого списка указана в столбце *N*) и списка построенного экспертом и упорядоченного респондентами (табл. 4.3). В столбце «*Набор слов»* в таблице 4.7 указаны те слова, выбранные экспертом, которые вошли в список, построенный автоматически программой Synarcher. В конце каждого слова стоит число, соответствующая порядковому номеру слова в автоматически построенном списке. Чем меньше эти номера и чем больше слов в столбце «*Набор слов»,* тем более похож автоматически построенный список на список семантически близких слов, построенный экспертом. В этом случае значение коэффициента Спирмена будет меньше.

Применение эвристики не изменило результат поиска для слова *истина*. 900 – это максимальное значение коэффициента Спирмена для списков из 100 и 9 слов, то есть их пересечение пусто. Это можно объяснить большим количеством статей, которые связаны со статьёй *Истина*: на неё ссылается 45 статей.

Преимущество и недостаток данной эвристики в сужении пространства поиска. Этим объясняется тот факт, что для слова *Жаргон* при включённой эвристике (1) синоним *Диалект* был пропущен, (2) общее число полученных семантически близких слов снизилось с 48 до 11 (табл. 4.7).

1 Эксперимент проводился на данных, соответствующих онлайн версии Русской Википедии от 18 июля 2006 г.

Таблица 4.7

Оценка влияние эвристики учёта статей (не содержащих в заголовке пробелов[1]) на результаты поиска

	Без эвристики			*С эвристикой*		
Слово	F	N	Набор слов	F	N	Набор слов
Жаргон	129	48	Арго8,Сленг11,Эвфемизм 19,Диалект28,Матерщина 36,Просторечие42	27	11	Арго1,Матерщина2,Эвфемизм 3,Просторечие4,Сленг8
Истина	900	100	Нет	900	100	Нет
Самолёт	161	100	Планер5,Автожир9,Экран олёт12,Турболёт13,Экран оплан41,Конвертоплан96	48	78	Планер2,Автожир4,Турболёт6, Экраноплан7,Экранолёт9,Конв ертоплан35
Сюжет	547	100	Трагедия50	446	95	Трагедия12,Интрига57

Данный опыт показал, что в целом применение эвристики (не учитывать статьи с пробелами) понижает значение коэффициента Спирмена (табл. 4.7), то есть строится список, более близкий к списку эксперта. Это был ожидаемый результат, поскольку список семантически близких слов, построенный экспертом (табл. 4.3), содержит однословные понятия, то есть слова без пробелов.

Применение коэффициента Спирмена для оценки параметров адаптированного HITS алгоритма[2]

Было проведено 66 опытов для каждого из четырёх слов: *жаргон, истина, самолёт, сюжет*. Менялись такие входные параметры адаптированного HITS алгоритма, как: размер корневого набора страниц (от 10 до 510 с шагом 50), инкремент (от 10 до 60 с шагом 10)[3]. Чёрный список категорий был тот же, что и в других экспериментах: *Страны*|*Века*|*Календарь*|*География_России*|*Люди*. Погрешность для останова итераций:

1 То есть заголовки статей состоят из одного слова.

2 Эксперимент проводился на данных, соответствующих онлайн версии Русской Википедии от 18 июля 2006 г.

3 Для этого была написана подпрограмма с вложенным циклом, в теле цикла которой вызывался адаптированный HITS алгоритм.

0.01. Усреднённые значения выходных параметров алгоритма приведены в таблице 4.8.

Таблица 4.8

Средние значения выходных параметров адаптированного HITS алгоритма

Слово	F	Inter-section	N_{expert}	N_{auto}	time (мин)	iter	vertices	edges	$S_{category}$
Жаргон	22.4	5.7	6	15.0	5.6	30.2	155.4	393.4	19855.2
Истина	900.0	0	9	100	19.9	19.2	458.8	2631.0	73257
Самолёт	50.2	6.0	6	86.8	7.4	12.6	144.0	547.0	29252.2
Сюжет	426.2	1.9	6	90.1	32.3	14.2	849.0	4381.4	119069.2

Графа *Intersection* указывает среднее число общих слов двух списков: (1) списка построенного экспертом и упорядоченного респондентами, см. таблицу 4.3 (размер этого списка указан в графе N_{expert}) и (2) списка, автоматически построенного программой (размер этого списка см. в графе N_{auto}).

Графа $S_{category}$ показывает число шагов по дереву категорий, для того чтобы выяснить, какие статьи нужно удалить / добавить в зависимости от содержимого входного параметра алгоритма: *чёрный список категорий*. Этот параметр, также как и параметры: *time* (время выполнения поиска), *iter* (число итераций для вычислений весов *hub* и *authority* страниц), позволяет косвенно судить о временных затратах алгоритма.

Параметры *vertices* и *edges* (число вершин и рёбер в базовом наборе страниц соответственно) позволяют судить о порядке размера оперативной памяти, необходимой для вычислений.

Эксперименты показали быструю сходимость итеративных вычислений (порядка 20, 30 шагов), см. графу *iter* в таблице 4.8. Аналогичная скорость сходимости указана в [125] в экспериментах по поиску похожих интернет страниц.

Графа *time* таблицы 4.8 указывает среднее время обработки поискового запроса. Полчаса (для слова *сюжет*) это чрезвычайно много для того, чтобы говорить об онлайн версии системы поиска. Необходимы эвристики, позволяющие ускорить время поиска, либо позволяющие выполнять какую-

либо предобработку поиска. Это особенно важно, если учесть, что размер Английской Википедии на порядок больше Русской.

В таблице 4.9 указаны минимальное (F_{min}), максимальное (F_{max}), среднее значение (F_{avg}) и стандартное отклонение (F_{stdev}) коэффициента Спирмена для той же серии опытов, что и в предыдущей таблице.

Таблица 4.9

Значения коэффициента Спирмена в серии опытов построения списков семантически близких слов

Слово	F_{min}	F_{max}	F_{avg}	F_{stdev}
Жаргон	20	30	22.36	2.75
Самолёт	45	59	50.21	4.41
Сюжет	60	479	426.18	95.97

Из таблицы 4.9 можно сделать вывод, что для некоторых слов (*жаргон, самолёт*) качество результата поиска достаточно стабильно[1] (значение стандартного отклонения коэффициента Спирмена 2.75 и 4.41 соответственно). В этом случае перед пользователем не стоит такой нетривиальной[2] задачи, как выбор входных параметров адаптированного HITS алгоритма.

Для многозначного слова *сюжет* всё сложнее. Такое высокое значение стандартного отклонения коэффициента Спирмена (95.97) указывает, что наличие в автоматически построенном списке тех слов, которые являются семантически близкими, в большей степени зависит от входных параметров алгоритма. Возможно, это связано с большей употребимостью слова *сюжет* в текстах энциклопедических статей, с большим количеством ссылок на эту статью (среднее число вершин – 849.0, рёбер – 4381.4 в базовом наборе для слова *сюжет*, см. таблицу 4.8). Для таких слов пользователю программы, вероятно, придётся не раз менять параметры программы поиска, чтобы найти как можно больше семантически близких слов.

1 Под качеством результата поиска понимается число тех слов, которые одновременно есть (1) и в автоматически создаваемом программой списке, (2) и в списке семантически близких слов, составленном экспертом.

2 То есть неформализованной на данный момент.

4.2 Сессия нормализации слов на основе модуля Russian POS Tagger, как одного из этапов автоматической обработки текстов в системе GATE

Ниже приведён пример результата работы модуля *Lemmatizer* (разработанного в проекте Диалинг [59]) для слова *рама.* Результат включает в себя (1) лемму слова, (2) часть речи (С,Г,П,...), (3) информацию о словоформе в кодах Ancode[1], (4) граммему[2], (5) уникальный идентификатор, (6) список словоформ:

«+ {РАМ, С, "дфст,лок,но", ("мр,рд,ед",) } Id=69343 \nAll forms: РАМ РАМА РАМУ РАМ РАМОМ РАМЕ РАМЫ РАМОВ РАМАМ РАМЫ РАМАМИ РАМАХ

+ {РАМА, С, "но", (\"жр,им,ед\",) } Id=98067 Accented=РА'МА\nAll forms: РАМА РАМЫ РАМЕ РАМУ РАМОЙ РАМОЮ РАМЕ РАМЫ РАМ РАМАМ РАМЫ РАМАМИ РАМАХ»

Заметим, что одной словоформе может соответствовать много морфологических интерпретаций (в данном случае две интерпретации для слова *рама*). И результат работы для слова *доброго*:

«+ {ДОБРЫЙ, П, \"кач\", (\"но,од,мр,рд,ед\",\"од,мр,вн,ед\",\"но,од,ср,рд,ед\",) } Id=138557 Accented=ДО'БРОГО\n

All forms: ДОБРЫЙ ДОБРОГО ДОБРОМУ ДОБРОГО ДОБРЫЙ ДОБРЫМ ДОБРОМ ДОБРАЯ ДОБРОЙ ДОБРОЙ ДОБРУЮ ДОБРОЙ ДОБРОЮ ДОБРОЙ ДОБРОЕ ДОБРОГО ДОБРОМУ ДОБРОЕ ДОБРЫМ ДОБРОМ ДОБРЫЕ ДОБРЫХ ДОБРЫМ ДОБРЫХ ДОБРЫЕ ДОБРЫМИ ДОБРЫХ ДОБР ДОБРА ДОБРО ДОБРЫ ДОБРЫ ДОБРЕЕ ПОДОБРЕЕ ДОБРЕЙ ПОДОБРЕЙ \n\n»

Более подробно о работе морфологического модуля *Lemmatizer* см. в работах [59], [60].

Благодаря разработанному автором модулю *Russian POS Tagger*, эта информация теперь доступна как в системе GATE, так и в отдельном приложении на языке Java. Для подключения *Russian POS Tagger* необходимо установить модуль морфологического анализа *Lemmatizer*, систему GATE и программу *Russian POS Tagger* (см. инструкции на сайте http://rupostagger.sourceforge.net).

Для инициализации *Russian POS Tagger* и подключения к XML-RPC серверу LemServer указаны следующие параметры:

- выбран английский словарь (*dictLemServer=ENGLISH*);
- выбрана Unicode кодировка (*encoding=UTF-8*);

---

1 Все возможные морфологические интерпретации хранятся в таблице Ancodes в Lemmatizer. Ключом является поле Ancode («аношкинский код») [60].

2 «Граммема – это элементарный морфологический описатель, относящий словоформу к какому-то морфологическому классу» [59]. Например, словоформе *рама* с леммой РАМ будет приписан следующий набор граммем: «мр,рд,ед».

- XML-RPC сервер *LemServer*, а значит, и *Lemmatizer* находятся на машине student (*hostLemServer=student*);
- XML-RPC сервер слушает порт 8000 (*portLemServer=8000*).

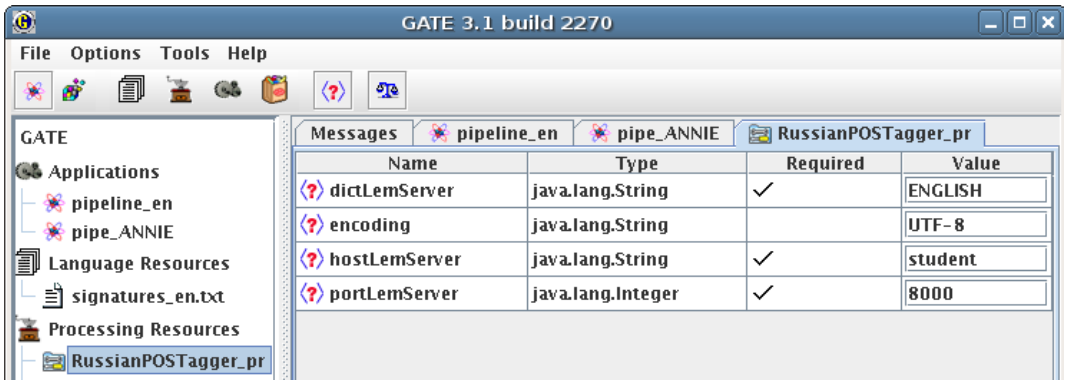


Рис. 27. Параметры GATE модуля *Russian POS Tagger*

Для запуска модуля в GATE нужно создать приложение (*pipeline_en* на рис. 27 и рис. 28) и назначить ему последовательность обработчиков (*Processing Resources*). Также нужно создать текстовый ресурс (*Language Resource*), например, текстовый файл *signatures_en.txt*.

На рис. 28 показано, что приложению *pipeline_en* присвоена последовательность из четырёх обработчиков (Document Reset PR, ANNIE English Tokenizer, ANNIE Sentence Splitter, Russian POS Tagger)[1] и каждому из обработчиков присвоен текстовый ресурс *signatures_en.txt*. На рисунке показано присвоение текстового ресурса свойству *document* модуля *Russian POS Tagger* и отмечено, что это необходимый параметр (required).

После запуска приложения *pipeline_en* к текстовому ресурсу будут последовательно применены указанные обработчики, передавая от одного другому, как эстафетную палочку, те наборы аннотаций (*annotation sets*), которые они строят в течение своей работы. Наборы аннотаций предыдущих модулей содержат данные, необходимые для работы последующих. В результате работы приложения (включающего модуль *Russian POS Tagger*) среди прочих будут построены два набора аннотаций: Wordform и Paradigm. Они содержат данные, приведённые вначале этого раздела: лемму слова, часть речи и др., полученные от модуля морфологического анализа *Lemmatizer*. Эти данные представлены пользователю в графической среде GATE (рис. 29).

1 Более подробно эти модули GATE описаны в гл. 3 на стр. 106.

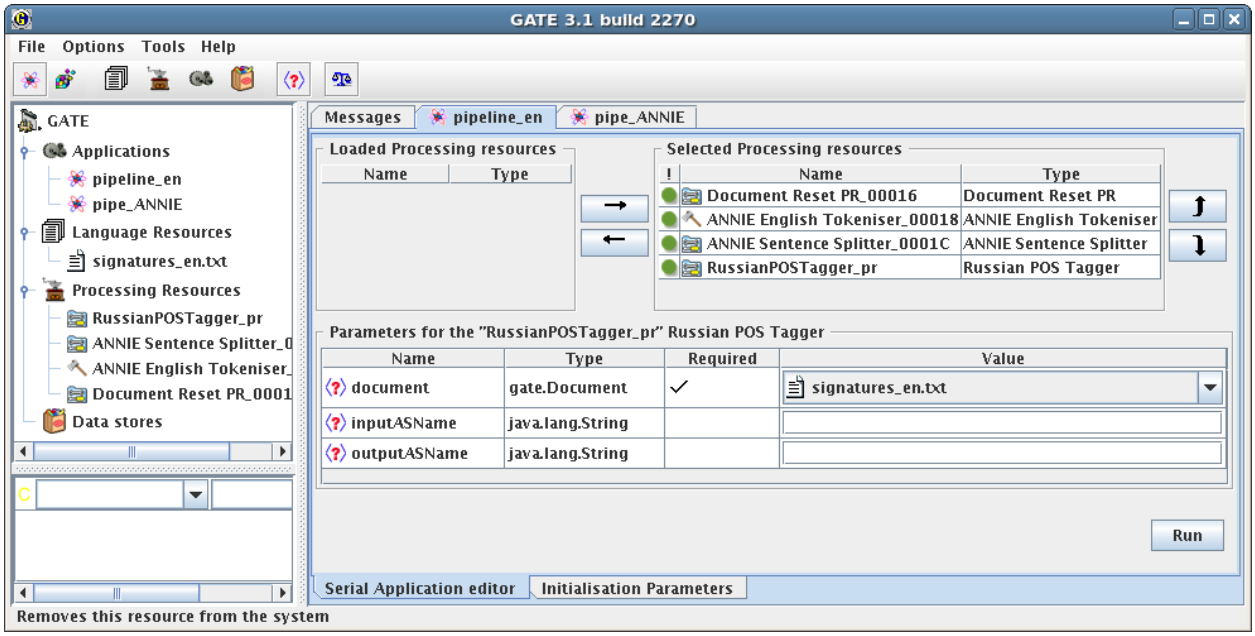


Рис. 28. Определение последовательности обработчиков в GATE

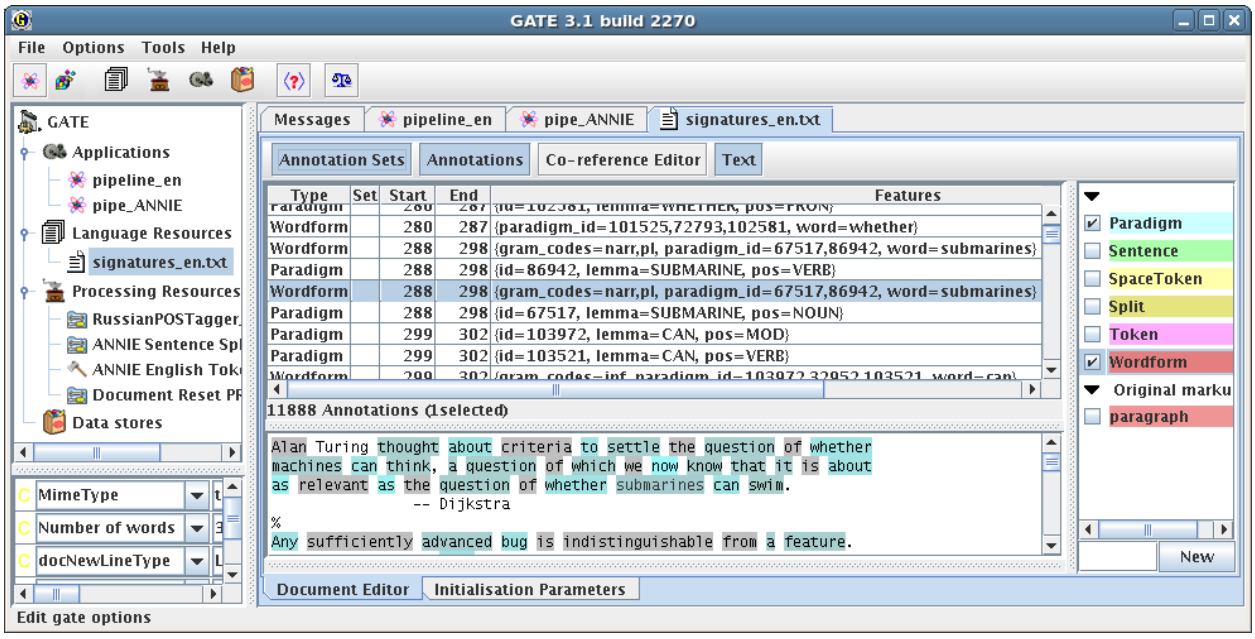


Рис. 29. Результат построения аннотаций *Paradigm* и *Wordform*

4.3 Индексирование вики-текста: инструментарий и эксперименты

Архитектура системы индексирования и структура таблиц индексной базы данных (БД) описаны в третьей главе.

Преобразование вики-текста с помощью регулярных выражений

Тексты Википедии написаны по правилам вики-разметки. Существует насущная необходимость в преобразовании вики-текста, а именно в удалении либо «раскрытии» тегов вики (то есть извлечении текстовой части). Если опустить данный шаг, то в сотню наиболее частых слов индексной БД попадают специальные теги, например «ref», «nbsp», «br» и др.[1] В ходе работы возникали вопросы: «как и какие элементы разметки обрабатывать?». Представим, аналогично работе [14] (стр. 20), возникшие вопросы и принятые решения в табл. 4.10. Для наиболее интересных, но могущих быть записанными в одну строку преобразований в таблице приведены регулярные выражения [63].

1 Собственно, анализ самых частых терминов, полученных в индексной БД, позволял находить элементы вики-разметки, требующие обработки. Код переписывался и база генерировалась вновь.

Таблица 4.10

Решения по парсингу вики-текста

N	**Вопросы** **Исходный текст**	**Ответы** **Преобразованный текст**

1	Заголовки (подписи) рисунков
	`[[Изображение:Через-тернии-к-звёздам.jpg|thumb|«Через тернии к звёздам»]]`
	`[[Image:Asimov.jpg|thumb|180px|right|[[Isaac Asimov]] with his [[typewriter]].]]`
		Оставить (извлечь)
		`«Через тернии к звёздам»`
		
		`[[Isaac Asimov]] with his [[typewriter]].`

2	Интервики	Оставить или удалить определяется пользователем (параметр `b_remove_not_expand_iwiki`).
3	Название категорий	Удалить.
	Регулярное выражение (RE): **\[\[Категория:.+?\]\]**	
4	Шаблоны; цитаты[1]; таблицы	Удалить.
5	Курсивный шрифт и «жирное» написание. `'''italic''' ''bold''`	Знаки выделения (апострофы) удаляются. `italic bold`

6	Внутренняя ссылка
	`[[w:Wikipedia:Interwikimedia_links|text to expand]]`
	`[[run]]`
	`[[Russian language|Russian]]`
	`в [[космос|космическом пространстве]].`
		Оставить текст, видимый пользователю, удалить скрытый текст.
		`text to expand`
		
		`run`
		`Russian`
		`в космическом пространстве.`

	RE: внутренняя ссылка без вертикальной черты: **\[\[([^:|]+?)\]\]**	

7	Внешняя ссылка
	`[http://example.com Russian]`
	`[http://www.hedpe.ru сайт hedpe.ru – русский фан-сайт]`
		Оставить текст, видимый пользователю, удалить сами гиперссылки.
		`Russian`
		`сайт – русский фан-сайт`

	RE: Имя сайта (без пробелов), содержащее точку '.' хотя бы раз, кроме последнего символа: **(\A|\s)\S+?[.]\S+?[^.]([\s,!?]|\z)**	
8	Примечание. `word1<ref>Ref text.</ref> – word2.`	Раскрывать, переносить в конец текста. `word1 word2.\n\nRef text.`

1 Имеются в виду короткие цитаты: *{{цитата|текст}}*, см. http://ru.wikipedia.org/wiki/Википедия:Шаблоны/Форматирование.

Для преобразования текстов в вики-формате в тексты на ЕЯ последовательно выполняются следующие шаги, которые можно разбить на две группы: (i) удаление и (ii) преобразование текста.[1]

(i) Удаляются такие теги (вместе с текстом внутри них):

1. HTML комментарии (`<!-- ... -->`);
2. теги выключения форматирования (`<pre>...</pre>`)[2];
3. теги исходных кодов `<source>` и `<code>`.

(ii) Выполняются преобразования вики-тегов:

4. Извлекается текст примечаний `<ref>`, добавляется в конец текста;
5. Удаляется двойные фигурные скобки и текст внутри них (`{{шаблон}}`);[3]
6. Удаляются таблицы и текст (`{| table 1 \n {| A table in the table 1 \n|}|}`).
7. Удаляется знак ударения в текстах на русском языке (например, *Ко́тор*);
8. Удаляется тройной апостроф, окружающий текст и обозначающий `'''жирное выделение'''`; текст остаётся;
9. Удаляется двойной апостроф, обозначающий `''наклонное начертание''`; текст остаётся;
10. Из тега изображения извлекается его название, прочие элементы удаляются;
11. Обрабатываются двойные квадратные скобки (раскрываются внутренние ссылки, удаляются интервики и категории);
12. Обрабатываются одинарные квадратные скобки, обрамляющие гиперссылки: ссылка удаляется, текст остаётся;
13. Удаляются символы (заменяются на пробел), противопоказанные XML парсеру:[4] `<`, `>`, `&`, `&quot;`; удаляются также их «XML-безопасные» аналоги: `&lt;`, `&gt;`, `&amp;`, `&quot;`; а также: `&#039;`, ` `, `&ndash;`, `&mdash;` символы `<br />`,`<br/>`,`<br>` заменяются символом перевода каретки.

---

1 См. код функции *wikipedia.text.WikiParser.convertWikiToText* в программе Synarcher.

2 Поскольку теги <pre> обычно «оборачивают» исходный код программ, не содержащий текстов на ЕЯ, см. http://en.wikipedia.org/wiki/Wikipedia:How_to_edit_a_page#Character_formatting.

3 Данная подфункция вызывается дважды, чтобы удалить `{{шаблон в {{шаблоне}}}}`. Более глубокие вложения в данной версии не учитываются.

4 Имеется в виду парсер протокола XML-RPC системы RuPOSTagger.

Данный преобразователь вики-текста воплощён в виде одного из Java-пакетов программной системы Synarcher [126]. Для замены элементов текста широко использовались регулярные выражения [63] языка Java. В табл. 4.11 приведён фрагмент статьи Русской Википедии «Через тернии к звёздам (фильм)». Показан результат комплексного преобразования текста по всем вышеуказанным правилам.

Таблица 4.11

Пример преобразования вики-текста[1]

Исходный текст в вики-разметке	**Преобразованный текст**
{{Фильм | РусНаз = Через тернии к звёздам }} [[Изображение:Через-тернии-к-звёздам 2.jpg|thumb|«Через тернии к звёздам»]] '''«Че́рез те́рнии к звёздам»''' – [[научная фантастика|научно-фантастический]] двухсерийный фильм [[режиссёр]]а [[Викторов, Ричард Николаевич|Ричарда Викторова]] по сценарию [[Кир Булычёв|Кира Булычёва]]. == Сюжет == {{сюжет}} [[XXIII]] век. [[Звездолёт]] дальней разведки обнаруживает в [[космос]]е погибший корабль неизвестного происхождения, на нём – гуманоидных существ, искусственно выведенных путём клонирования. Одна девушка оказывается жива, её доставляют на [[Земля (планета)|Землю]], где [[учёный]] Сергей Лебедев поселяет её в своём доме. == В ролях == * [[Елена Метёлкина]] – ''Нийя'' == Ссылки == {{викицитатник}} * [http://ternii.film.ru/ Официальный сайт фильма] [[Категория:Киностудия им. М. Горького]] [[en:Per Aspera Ad Astra (film)]]	«Через тернии к звёздам» «Через тернии к звёздам» научно-фантастический двухсерийный фильм режиссёра Ричарда Викторова по сценарию Кира Булычёва. == Сюжет == XXIII век. Звездолёт дальней разведки обнаруживает в космосе погибший корабль неизвестного происхождения, на нём гуманоидных существ, искусственно выведенных путём клонирования. Одна девушка оказывается жива, её доставляют на Землю, где учёный Сергей Лебедев поселяет её в своём доме. == В ролях == * Елена Метёлкина Нийя == Ссылки == * Официальный сайт фильма

1 См. http://ru.wikipedia.org/wiki/Через_тернии_к_звёздам_(фильм).

API индексной базы данных вики

Укажем существующие программные интерфейсы (API) для работы с данными ВП:

- FUTEF API для поиска в Английской Википедии с учётом категорий ВП.[1] Поисковик реализован как веб-сервис на основе Yahoo!, результат возвращается в виде Javascript объекта JSON;[2]
- интерфейс для вычисления семантического сходства слов в ВП [149]. Здесь запрос идёт из Java через XML-RPC к Perl-процедуре, затем посредством MediaWiki выполняется обращение к БД;
- интерфейс к Википедии и Викисловарю [189]. Проведены эксперименты по извлечению данных из Английского и Немецкого Викисловаря. Главный недостаток программы — лицензия — «только для исследовательских целей».
- набор интерфейсов для работы с данными ВП, преобразованными в XML формат.[3]

Поскольку структура индексной БД отличается от схемы БД MediaWiki (для работы с которой уже написано достаточное количество необходимых функций в программе Synarcher), постольку возникла необходимость в разработке «сопряжения» для программного управления индексом. Итак, разработан программный интерфейс для работы с базой данных WikIDF.

I.) Интерфейс верхнего уровня позволяет:

1. получить список терминов для данной вики-страницы, упорядоченный по значению TF-IDF;
2. получить список документов, содержащих словоформы лексемы по заданной лемме; документы упорядочены по значению частоты термина (TF).

II.) Функции низкого уровня для работы с отдельными таблицами индексной БД (рис. 20) реализованы в пакете `wikipedia.sql_idf` программы Synarcher.

---

1 См. http://api.futef.com/apidocs.html.

2 См. http://json.org/json-ru.html.

3 См. http://modis.ispras.ru/sedna/ и http://wikixmldb.dyndns.org/help/use-cases/.

Эксперименты по построению индексных баз данных

Разработанная программная система индексирования вики-текстов позволила построить индексные БД для Simple English[1] (далее SEW) и Русской[2] (далее RW) википедий и провести эксперименты. Статистическая информация об исходных БД, о парсинге и о размерах полученных БД представлены в табл. 4.12.

В двух столбцах («RW / SEW 07» и «RW / SEW 08») указано во сколько раз параметры русского корпуса превосходят английский по дампам от 2007 года (SEW от 9 и RW от 20 сентября) и от 2008 года (SEW от 14 и RW от 20 февраля). Данными, характеризующими корпус текстов Русской Википедии, можно назвать большое количество лексем (1.43 млн) и общего числа слов (32.93 млн). Размер Русской Википедии примерно на порядок больше Английской (столбец «RW/SEW 08»): статей больше в 9.5 раз, лексем — в 9.6, всего слов — в 14.4 раза.

Значения следующих двух столбцов («SEW 08/07 %» и «RW 08/07 %») указывают, насколько выросли (по сравнению с собой же) английский и русский корпуса за пять месяцев с сентября 2007 до февраля 2008 гг.

В последнем столбце (SEW↑ /RW↑) указано насколько быстрее шёл рост английского корпуса по сравнению с русским (отношение предыдущих двух столбцов), а именно: на 12% быстрее появлялись статьи и на 6% быстрее пополнялся лексикон Википедии на английском упрощённом языке.

---

1 1000 наиболее частотных слов, полученных по текстам Википедии на упрощённом английском языке, см. http://simple.wiktionary.org/wiki/User:AKA_MBG/English_Simple_Wikipedia_20080214_freq_wordlist. (14.02.2008)

2 1000 наиболее частотных слов, полученных по текстам Русской Википедии (20 февраля 2008), см. http://ru.wiktionary.org/wiki/Конкорданс:Русскоязычная_Википедия/20080220.

Таблица 4.12

Статистика по Русской Википедии и Simple Wikipedia, парсингу и сгенерированным индексным базам данных

БД Википедии	Simple English (SEW 08)	Russian (RW 08)		RW/ SEW 07	RW/ SEW 08		SEW 08/07 %	RW 08/07 %	SEW↑ /RW↑ %
База данных Википедии									
Исходный дамп, дата	14 фев. 2008	20 фев. 2008		–	–		–	–	–
Исходный дамп, размер, МБ[1]	21.11	240.38		15.9	14.4		40	26	10
Статей, тыс.	25.22	239.29		10.7	**9.5**		31	17	**12**
Парсинг									
Парсинг всего, ч	3.63	69.26		15.1	19.1		4	32	-21
Парсинг одной страницы, сек	0.52	1.04		1.42	2.01		-20	13	-30
Индексная база данных Википедии									
Лексем в корпусе, млн	0.149	**1.43**		10.2	**9.6**		23	16	**6**
Лексема-страница (<=1000 для слова)[2], млн	1.65	15.71		10.1	9.5		24	16	6
Слов в корпусе, млн	2.28	**32.93**		15.1	**14.4**		29	23	5
Размер сжатого файла дампа индексной БД, МБ	7.15	77.5		11.5	10.8		25	17	6

Чтобы время парсинга, приведённое в табл. 4.12, имело смысл, укажем параметры рабочего компьютера и версии двух основных программ: ОС Debian 4.0 etch, ядро Linux 2.6.22.4, процессор AMD 2.6 ГГц, 1 ГБ RAM, Java SE 1.6.0_03, MySQL 5.0.51a-3.

Теперь обратимся к такому интересному вопросу корпусной лингвистики как распределение частот слов, упорядоченных по своей частоте, и проверим выполнение гипотезы Ципфа для текстов Википедии.[3]

---

1 Размер файла «...-pages-articles.xml.bz2», содержащего тексты статей.

2 Число связок «слово-статья» в корпусе, учёт не более 1000 для одного слова. Число *1000* здесь — это один из входных параметров программного комплекса построения индексной БД, см. «TF-IDF contraints» на рис. 1.

3 Следует признать, что к данному вопросу авторов привлёк рисунок с распределением частот слов в Английской Википедии за 2006 г., см. http://en.wikipedia.org/wiki/Zipf%27s_law#Related_laws.

Проверка выполнения закона Ципфа для вики-текстов

Эмпирический закон Ципфа говорит о том, что частота употребления слова в корпусе обратно пропорциональна его рангу в списке упорядоченных по частоте слов этого корпуса [131] (стр. 23), то есть второе по частоте слово будет употребляться в текстах в два раза реже чем первое, третье — в три раза и так далее.

Другая формулировка закона Ципфа гласит: если построить список слов, отранжировав слова по уменьшению их частоты встречаемости в некотором *достаточно большом* тексте, и нарисовать график логарифма частот слов в зависимости от логарифма порядкового номера в списке, то получится прямая [155]. См. также построение аппроксимирующих расчётных ранговых распределений частот появления слов (РРЧС) в тексте А. С. Пушкина “Медный всадник” в работе [4].

На рис. 30 слова упорядочены (по убыванию частоты) вдоль оси абсцисс, вдоль оси ординат отложена частота слов. Кривая, составленная из знаков «+», построена по данным корпуса текстов Русской Википедии «RW 08». С помощью метода наименьших квадратов пакета Scilab [91] были построены и нарисованы *аппроксимирующие кривые* y_{100}^{RW} по первым ста наиболее частотным словам корпуса (см. рис. 30, кривая розового цвета, длинный пунктир) и $y_{10\text{K}}^{RW}$ по первым 10 тыс. слов (голубая линия, пунктир с точкой):

$$y_{100}^{RW}(x)=\frac{\mathrm{e}^{14.51}}{x^{0.819}} \quad ; \quad y_{10\text{K}}^{RW}(x)=\frac{\mathrm{e}^{16.13}}{x^{1.048}} \tag{4.1}$$

Знакам «X» на рис. 30 соответствуют данные Википедии «SEW 08». Аналогично нарисованы аппроксимирующие кривые: y_{100}^{SEW} (зелёного цвета, точечный пунктир) и $y_{10\text{K}}^{SEW}$ (красного цвета, пунктир с двумя точками):

$$y_{100}^{SEW}(x)=\frac{\mathrm{e}^{12.83}}{x^{0.974}} \quad ; \quad y_{10\text{K}}^{SEW}(x)=\frac{\mathrm{e}^{14.29}}{x^{1.174}} \tag{4.2}$$

Рис. 30 показывает, что закон Ципфа в целом выполняется для текстов википедий, то есть кривую на рисунке с логарифмическим масштабом вполне можно аппроксимировать прямой. При этом данные Simple Wikipedia (0.20)[1] соответствуют данному закону немного лучше корпуса русских текстов (0.23). Что довольно-таки странно, поскольку размер Русской Википедии на порядок больше (табл. 4.12). Такое пристрастие выполнения закона к текстам на английском упрощённом языке, по сравнению с русским, можно объяснить либо особенностью упрощённого языка, либо разницей между русским и английским языками. Для окончательного выяснения вопроса нужно решить задачу промышленного масштаба, а именно: построить индексную БД для English Wikipedia.

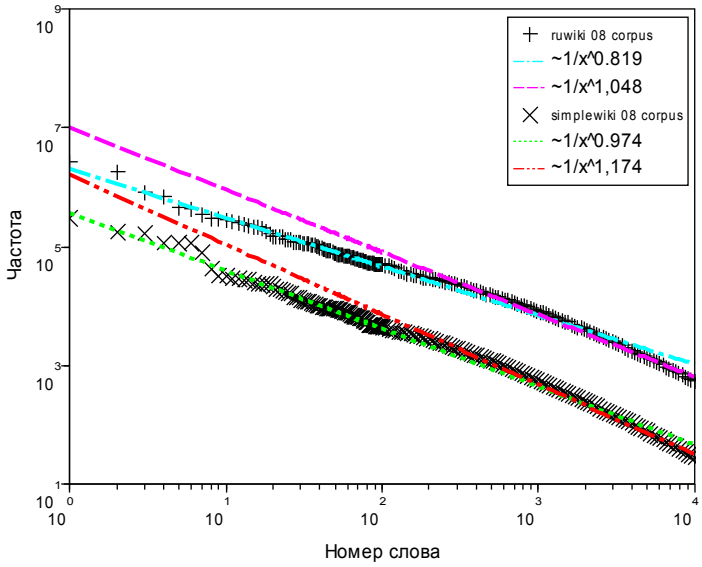


Рис. 30. Линейная зависимость убывания частоты слов от ранга в частотном списке слов (в масштабе логарифм-логарифм) для Русской Википедии и Simple Wikipedia на февраль 2008 г.

На рис. 31 представлено распределение частоты слов в текстах двух википедий *в два момента времени*, то есть на рисунке приведены данные для тех же четырёх корпусов (индексных БД), что представлены в табл. 4.12. Перечислим значения пяти кривых (сверху вниз) на рис. 31:

1 0.20 — разница между степенью наклона аппроксимирующих прямых по ста (0.974) и тысяче (1.174) слов.

– «*ruwiki 08 corpus*» (линия красного цвета, пунктир) — частота слов в корпусе Русской Википедии на 20.02.2008;

– «*07 corpus*» (фиолетовый цвет, пунктир: две точки, один штрих) — частота слов в корпусе Русской Википедии на 20.09.2007;

– «*ruwiki 07 doc*» (серый цвет, широкая полоса) — число документов, содержащих лексемы тех же слов, что указанны на графике «*07 corpus*» (Русская Википедия, 20.09.2007);[1]

– «*simplewiki 08 corpus*» (фиолетовый цвет, непрерывная линия) — частота слов в корпусе Simple Wikipedia на 14.02.2008;

– «*simplewiki 07 corpus*» (зелёный цвет, пунктир) — частота слов в корпусе Simple Wikipedia на 09.09.2007.

График на рис. 31 справа тот же, что и слева, за исключением того, что прологарифмирована не только частота слов в корпусе и число документов (с этим же словом), но также и число слов (ранг в списке). Графики показывают, что характер зависимостей (и выполнение закона Ципфа) сохранился за истекшие полгода в обеих википедиях.

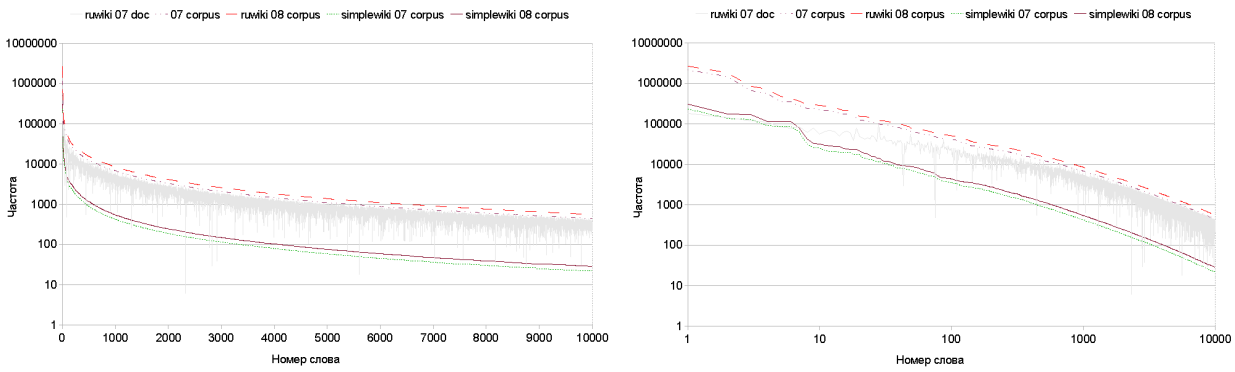


Рис. 31. Распределение частоты десяти тысяч наиболее употребимых слов в вики-корпусах на русском (ruwiki) и английском упрощённом (simplewiki) языках за 2007 и 2008 гг. (слева); проверка выполнения закона Ципфа для данных корпусов (справа)

Предложим читателю ещё ряд экспериментов, которые можно выполнить, пользуясь данными индексной БД:

а) взять первую 1000 слов по частоте в корпусе, упорядочить по числу

1 Можно построить аналогичный график, если отсортировать слова не по частоте слов в корпусе, а по числу документов, содержащих слова. В этом случае взаимно поменяется характер кривых «07 corpus» и «ruwiki 07 doc».

документов, построить график;

б) найти число слов с частотой 1, 2, 3.. 10..1000 слов в корпусе (привести в таблице и построить гистограмму);

в) найти число слов длиной 1 буква, 2, 3.. 30 (таблица и гистограмма);

г) сравнить изменение ранга слов (по популярности, то есть частоте употребления) в корпусе со временем (слово, ранг, повышение/понижение — на сколько), (подсказка: нужно построить две индексных БД для разных по времени дампов википедий); указать часто употребимые слова, ранг которых изменился максимально;

д) найти число различных лексем в документе; среднее и максимальное число по всем документам; то же, но нормированное на число слов в документе; вычислить список первых десяти самых «пёстрых» документов, то есть богатых своим лексиконом, а именно: содержащих максимальное значение отношения числа уникальных лексем к числу слов в документе; обратная задача: построить упорядоченный список самых «нудных», т.е. длинных, но бедных лексиконом документов.

4.4 Эксперименты в проекте «Контекстно-зависимый поиск документов в проблемно-ориентированных корпусах»

В проекте «Контекстно-зависимый поиск документов в проблемно-ориентированных корпусах» предлагается решение задача поиска похожих документов на основе онтологий[1].

Задача поиска похожих документов решается в два этапа: индексирование и поиск на основе индекса.

1. *Индексирование* документов относительно онтологии заключается в (i) определении фрагмента онтологии, соответствующего документу (классы, атрибуты, отношения между ними, см. рис. 34), (ii) вычислении сходства (*similarity*) между документом и данным фрагментом онтологии. В базе данных

1 По классификации, предложенной в работе [118], приложение, разработаннное в данном проекте, относится к классу «ontology-based search». Определение онтологии см. в [107], [181]. Введение и обзор онтологий также представлены в отечественной монографии [61].

сохраняется тройка <*A''*, *Doc* ^{ID}, *Sim*>, где *A''* – фрагмент онтологии, *Doc* ^{ID} – идентификатор документа, *Sim* – степень сходства. Предложено два варианта индексирования: на основе категорий (рис. 32) и полнотекстовое (рис. 33). При полнотекстовом индексировании используется модуль *Russian POS Tagger* (см., раздел 3.2 на стр. 106).

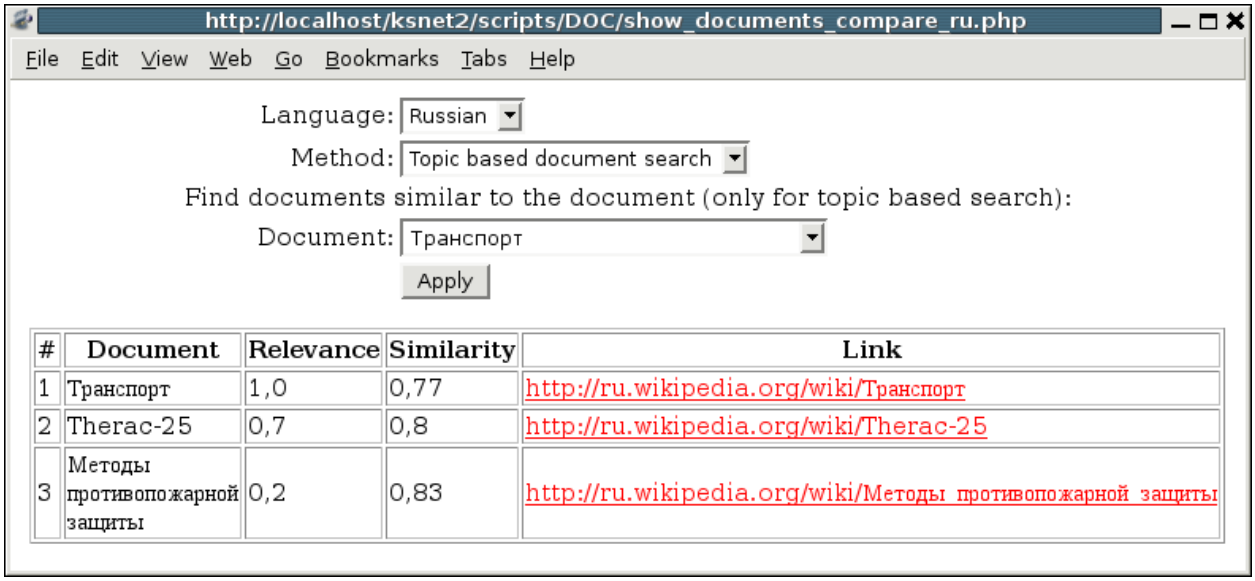
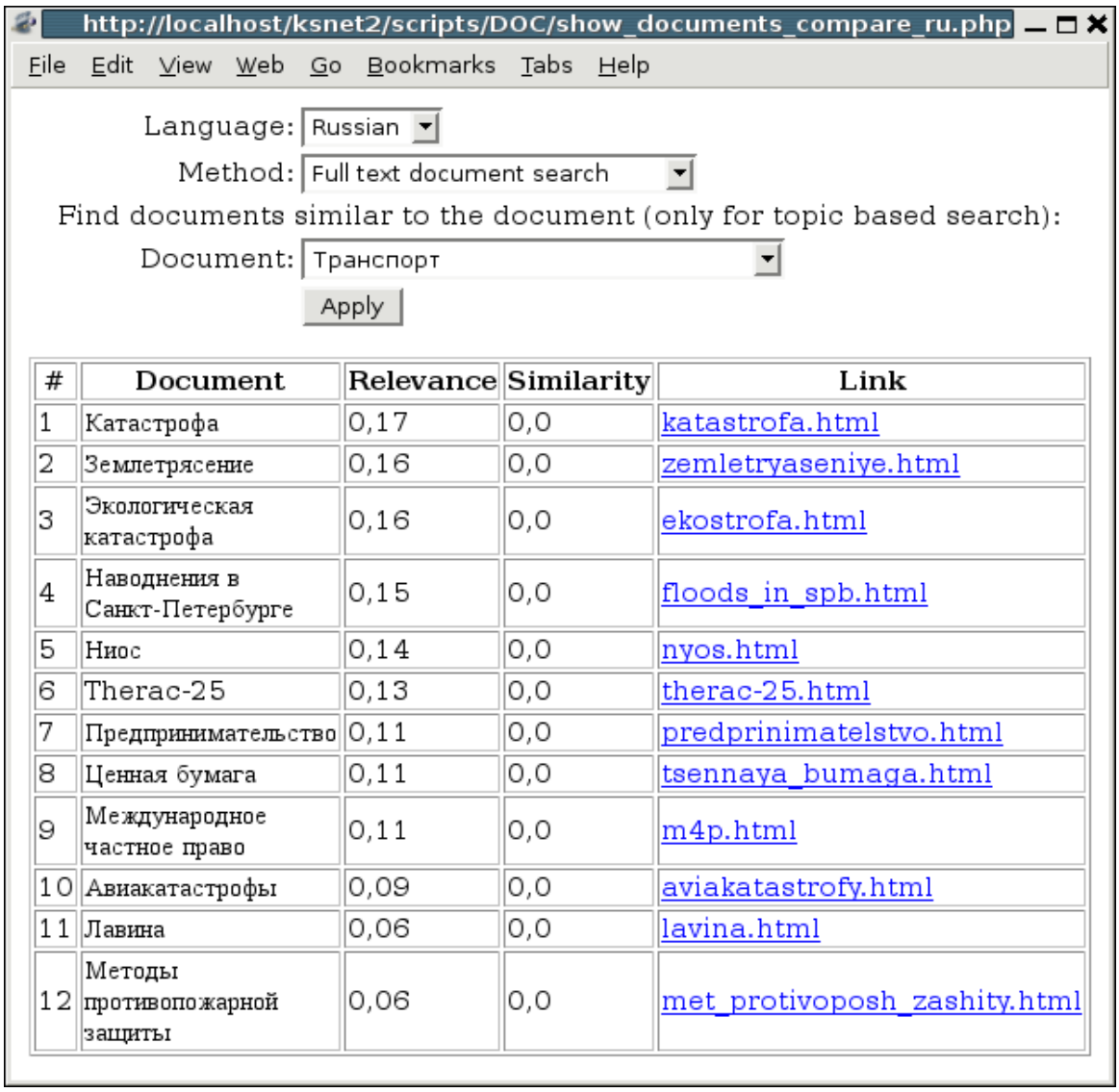


Рис. 32. Список документов, похожих на документ «Транспорт», полученный с помощью алгоритма поиска на основе категорий

2. Поиск похожих документов на основе индекса заключается в следующей последовательности шагов. По исходному документу выбирают фрагмент онтологии из базы данных индекса. Для данного фрагмента находят похожие фрагменты, сравнивая его с фрагментами, хранящимися в индексе. Упорядоченному (по степени сходства) списку похожих фрагментов соответствует список документов, который возвращают, как упорядоченный список похожих документов (рис. 32). Документы на рис. 32 упорядочены по графе релевантность (*relevance*).



#	Document	Relevance	Similarity	Link
1	Катастрофа	0,17	0,0	katastrofa.html
2	Землетрясение	0,16	0,0	zemletryaseniye.html
3	Экологическая катастрофа	0,16	0,0	ekostrofa.html
4	Наводнения в Санкт-Петербурге	0,15	0,0	floods_in_spb.html
5	Ниос	0,14	0,0	nyos.html
6	Therac-25	0,13	0,0	therac-25.html
7	Предпринимательство	0,11	0,0	predprinimatelstvo.html
8	Ценная бумага	0,11	0,0	tsennaya_bumaga.html
9	Международное частное право	0,11	0,0	m4p.html
10	Авиакатастрофы	0,09	0,0	aviakatastrofy.html
11	Лавина	0,06	0,0	lavina.html
12	Методы противопожарной защиты	0,06	0,0	met_protivoposh_zashity.html

Рис. 33. Список документов, похожих на документ «Транспорт», полученный с помощью алгоритма полнотекстового поиска

Для решения данных задач автором были спроектированы и реализованы: (i) алгоритм индексирования на основе категорий Википедии, (ii) алгоритм сравнения графов, (iii) алгоритм выбора минимального связного набора вершин в графе (рис. 35).

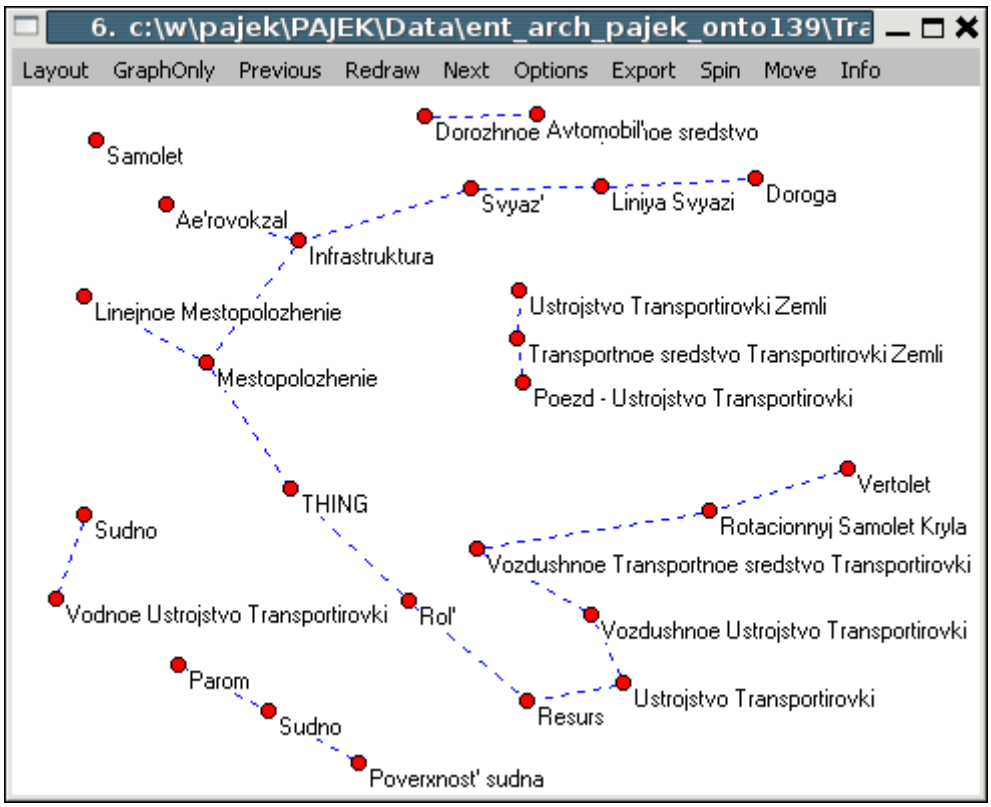


Рис. 34. Фрагмент онтологии, соответствующий документу «Транспорт»

Алгоритм сравнения графов и алгоритм выбора минимального связного набора вершин реализованы в виде модулей библиотеки Java Universal Network / Graph Framework (JUNG) [144]. JUNG – это программная библиотека с открытым исходным кодом, обеспечивающая средства для управления, анализа и визуализации таких данных, которые можно представить в виде графа.

Для демонстрации алгоритма сравнения графов автором создано приложение Java WebStart, позволяющее сравнивать графы[1]. Данное приложение можно также рассматривать как пример интеграции библиотеки JUNG и технологии WebStart[2].

Для визуального отображения графов в целях отладки (см., например, рис. 34, 35) использовался формат и программа *Pajek*, разработанные словенскими учёными [78].

1 Приложение и код программы доступны по адресу: http://whinger.narod.ru/soft/edoc.jung/index.html.

2 См. http://java.sun.com/products/javawebstart, http://mindprod.com/jgloss/javawebstart.html.

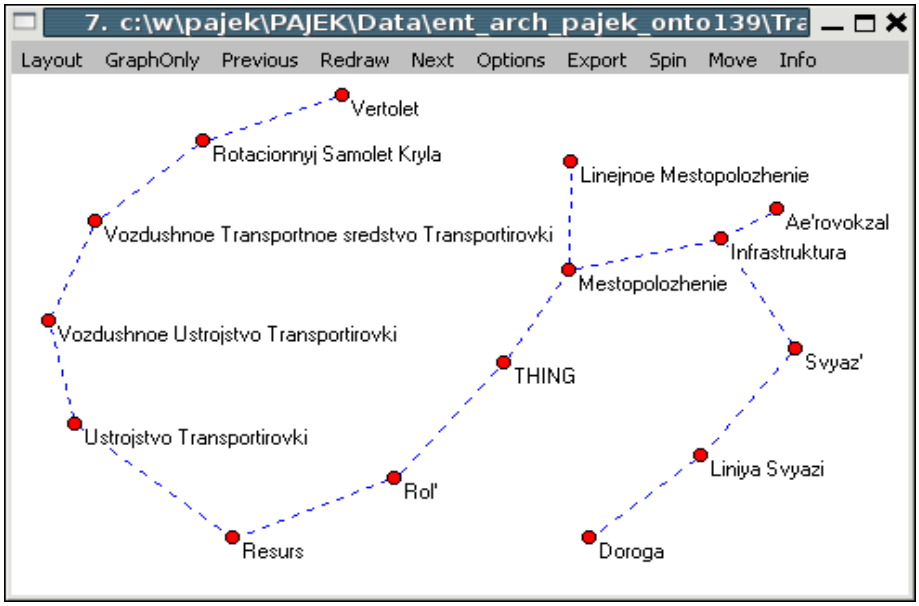


Рис. 35. Из фрагмента онтологии, соответствующего документу «Транспорт», выбран связный граф

Выводы по главе 4

В этой главе (1) описаны эксперименты поиска синонимов в Английской и Русской Википедии с помощью адаптированного HITS алгоритма, (2) дано описание (с точки зрения пользователя) сессии поиска синонимов в программы Synarcher (реализующей адаптированный HITS алгоритм), (3) численно проверена польза предложенных поисковых эвристик, (4) проведены эксперименты по построению индексных баз данных вики-текстов, (5) получено подтверждение выполнения закона Ципфа для текстов двух Википедий.

Эксперименты позволяют сделать вывод, что программа Synarcher позволяет находить синонимы и семантически близкие слова в Английской Википедии, отсутствующие в современных тезаурусах WordNet, Moby (например, найден синоним *Spationaut* для слова *Astronaut*). Тем не менее некоторые синонимы, представленные в тезаурусах, не были найдены. Таким образом, можно улучшить алгоритм, используя данные тезаурусов.

Эксперименты показывают, что работа AHITS алгоритма медленнее HITS алгоритма в среднем на 52%, а точность поиска AHITS алгоритма выше на 33%.

Выполнена численная оценка (с помощью коэффициента Спирмена) влияния эвристики на качество строящегося автоматически списка семантически близких слов. Суть эвристики в том, чтобы не включать в корневой и в базовый набор те энциклопедические статьи, названия которых содержат пробелы.

Предложенная модификация коэффициента Спирмена позволила провести эксперименты для оценки чувствительности результатов адаптированного HITS алгоритма к параметрам поиска. Для ряда слов из Русской Википедии (*Жаргон*, *Самолёт*) качество результата поиска[1] было достаточно стабильным (значение стандартного отклонения коэффициента Спирмена 2.75 и 4.41 соответственно), что избавляет пользователя от необходимости

1 Под качеством результата поиска понимается число тех слов, которые одновременно есть (1) и в автоматически создаваемом программой списке, (2) и в списке семантически близких слов, составленном экспертом.

тщательно подбирать параметры поиска. Для более часто употребимого (в данном корпусе текстов) слова *Сюжет* качество результата оказалось в большей степени зависимым от входных параметров алгоритма (значение стандартного отклонения коэффициента Спирмена 95.97).

Описаны данные морфологического анализа Lemmatizer, доступные в модуле Russian POS Tagger. Представлен пример инициализации модуля Russian POS Tagger, указаны параметры для его подключения к XML-RPC серверу LemServer. Показан способ подключения и результаты работы модуля Russian POS Tagger в составе системы GATE.

Описаны эксперименты по построению индексных баз данных Русской Википедии и Википедии на английском упрощённом языке. Данная работа не является первой, применяющей модуль Lemmatizer к текстам Википедии.[1] Тем не менее вкладом работы является: описание достаточно законченной системы индексирования вики-текстов (вероятно, первой общедоступной), а также индексные базы двух википедий, доступные для проведения исследований или включения в поисковые системы.

Проведены эксперименты, подтверждающие выполнение эмпирического закона Ципфа для текстов Русской Википедии и Википедии на английском упрощённом языке.

1 Методика построения индексной базы. 23.10.2006. http://ru.wikipedia.org/wiki/Википедия:Частотный_словник

Заключение

Одной из важных современных задач является задача поиска информации. Подзадачей является поиск похожих объектов, который кроме поиска похожих текстовых документов включает задачу поиска семантически близких слов, задачу поиска похожих вершин графа и др. С другой стороны большую популярность приобретает новый формат интернет страниц – вики. Всё это подвигло нас к решению такой теоретически занятной и имеющей большое практическое значение задаче как создание математического и программного обеспечения для поиска семантически близких слова на основе рейтинга вики-текстов.

Анализ методов поиска синонимов и методов поиска похожих интернет страниц показал, что HITS алгоритм наиболее подходит для поиска похожих документов в корпусах текстов специальной структуры (с гиперссылками и категориями). HITS алгоритма, изначально предназначенный для поиска похожих страниц в Интернете, был адаптирован для поиска наиболее похожих документов в корпусе текстов специальной структуры с использованием алгоритма кластеризации. Данный алгоритм был реализован в программном продукте *Synarcher* с визуализацией результатов поиска и с возможностями интерактивного поиска. Эксперименты показали возможность находить синонимы и семантически близкие слова в Английской Википедии, отсутствующие в современных тезаурусах WordNet, Moby.

В работе предложен итеративный алгоритм поиска похожих вершин графа. Предложены эвристики и проведена оценка временной сложности алгоритма.

Спроектирована архитектура программной системы оценивания степени синонимичности набора слов на основе тезаурусов (WordNet и Moby). Предложены способы оценки семантической близости для списков, строящихся автоматически.

Коэффициент Спирмена модифицирован для численного сравнения списков слов (отличие от оригинального метода заключается в возможности

сравнивать списки разной длины) и применён в экспериментальной части работы для оценки качества поиска семантически близких слов в энциклопедии Русская Википедия. Спроектирована клиент-серверная архитектура программного комплекса поиска семантически близких слов с возможностью оценки списков слов на основе удалённого доступа к тезаурусам (WordNet, Moby)

Предложена и реализована в виде программы интеграция распределённых программных компонент в рамках системы GATE, а именно: подключен модуль морфологического анализа русского языка (на основе XML-PRC протокола). Плюс данной части работы в том, что это один из шагов по созданию модулей обработки текстов на русском языке в системе GATE. Это по определению (инфраструктуры GATE) приведёт к созданию переносимых, совместимых, обладающих визуальным интерфейсом[1] модулей по обработке текстов на естественном языке.

Разработана архитектура системы индексирования вики-текстов, включающая программные модули GATE и Lemmatizer. Реализован программный комплекс индексации текстов Википедии на трёх языках: русский, английский, немецкий. Выполнено индексирование Русской Википедии и Википедии на английском упрощённом языке, построены индексные базы для них, выполнено сравнение основных показателей баз данных (число слов, лексем). На основе этих баз выполнена проверка, подтверждающая выполнение закона Ципфа для текстов Русской Википедии и Википедии на английском упрощённом языке.

Предложенное решение задачи автоматического поиска синонимов может использоваться в поисковых системах (расширение / переформулировка запроса с помощью тезаурусов), в системах машинного перевода, при составлении словарей синонимов.

1 Визуальный интерфейс GATE позволяет: (1) связывать модули друг с другом, (2) задавать параметры, (3) представлять результаты работы модулей.

Список источников литературы

[1]. Азарова И.В., Митрофанова О.А., Синопальникова А.А. Компьютерный тезаурус русского языка типа WordNet // Компьютерная лингвистика и интеллектуальные технологии. Труды международной конференции «Диалог 2003». Протвино, 2003. – С. 1-6. http://www.dialog-21.ru/materials/?id=56248

[2]. Александров В.В. Интеллект и компьютер. СПб.: «Анатолия», 2004. – 285 с. – ISBN 5-314-00080-6. http://www.sial.iias.spb.su/issue.html

[3]. Александров В.В., Андреева Н.А., Кулешов С.В. Системное моделирование. Методы построения информационно-логистических систем / Учеб. пособие. СПб.: Изд-во Политехн. ун-та, 2006. – 95 с.. http://sial.iias.spb.su/files/semsys.pdf

[4]. Александров В.В., Арсентьева А.В., Семенков А.В. Структурный анализ диалога. Препринт № 80. Л.: ЛНИВЦ, 1983. – 50 с..

[5]. Ахо А.В., Хопкрофт Д., Ульман Д.Д. Структуры данных и алгоритмы. : Пер. с англ. М.: Издательский дом "Вильямс", 2003. – 384 с. – ISBN 5-8459-0122-7.

[6]. Бек К. Экстремальное программирование. СПб.: Питер, 2002. – 224 с. – ISBN 5-94723-032-1.

[7]. Берков В.П. Двуязычная лексикография: Учебник. СПб.: Изд-во С.-Петербургского ун-та, 1996. – 248 с. – ISBN 5-288-01643-7.

[8]. Блох Дж. Java. Эффективное программирование. М.: Лори, 2002. – 224 с. – ISBN 5-85582-169-2.

[9]. Бобровский С. Технологии Пентагона на службе российских программистов. Программная инженерия. СПб.: Питер, 2003. – 222 с. – ISBN 5-318-00103-3.

[10]. Браславский П.И. Автоматические операции с запросами к машинам поиска интернета на основе тезауруса: подходы и оценки // Компьютерная лингвистика и интеллектуальные технологии. Труды международной конференции «Диалог 2004». Верхневолжский, 2004. – С. 79-84. http://www.dialog-21.ru/archive/2004/braslavskij.htm

[11]. Браславский П.И., Соколов Е.А. Автоматическое извлечение терминологии с использованием поисковых машин Интернета // Компьютерная лингвистика и интеллектуальные технологии. Труды международной конференции

«Диалог 2007». Бекасово, 2007. – С. 89-94. http://www.dialog-21.ru/dialog2007/materials/pdf/14.pdf

[12]. Брукс Ф. Мифический человеко-месяц или как создаются программные системы: Пер. с англ. СПб.: Символ-Плюс., 1999. – 304 с. – ISBN 5-93286-005-7.

[13]. Вайсфельд М. Объектно-ориентированный подход: Java, .Net, C++. Второе издание / Пер. с англ.. М.: КУДИЦ-ОБРАЗ, 2005. – 336 с. – ISBN 5-9579-0045-1, 0-672-32611-6.

[14]. Вахитова Д. Создание корпуса текстов по корпусной лингвистике. 2006. http://matling.spb.ru/files/kurs/Vahitova_Corpus.doc

[15]. Вишняков Р.Ю. Интеллектуальные информационно-поисковые системы. Лингвистический анализ // *Перспективные информационные технологии и интеллектуальные системы*. – 2006. – Т.**4** (28). – С. 37-42. http://pitis.tsure.ru/files28/05.pdf

[16]. Выготский Л.С. Мышление и речь. Общая психология. Тексты. Раздел 3. Вып. 1. Познавательные процессы: виды и развитие. Часть 2 / Под общей редакцией Петухова В.В. М., 1997. – С. 87.

[17]. Гаврилова Т.А., Хорошевский В.Ф. Базы знаний интеллектуальных систем. Санкт-Петербург, Питер, 2000. – 384 с. – ISBN 5-272-00071-4.

[18]. Голуб И.Б. Стилистика русского языка. Учеб. пособие, 2002. http://www.hi-edu.ru/x-books/xbook028/01/

[19]. Горбатов В.А. Фундаментальные основы дискретной математики. Информационная математика. М.: Наука. Физматлит, 1999. – 544 с. – ISBN 5-02-015238-2.

[20]. Грин Д., Кнут Д. Математические методы анализа алгоритмов: Пер. с англ. М.: Мир, 1987. – 120 с.. http://www.proklondike.com/file/CompScience/Grin_Knut_-_MatMetodi_v_analize_algoritmov.rar

[21]. Грунтов И.А. «Каталог семантических переходов» – база данных по типологии семантических изменений // Компьютерная лингвистика и интеллектуальные технологии. Труды международной конференции «Диалог 2007». Бекасово, 2007. – С. 157-161. http://www.dialog-21.ru/dialog2007/materials/pdf/23.pdf

[22]. Гулин А., Маслов М., Сегалович И. Алгоритм текстового ранжирования Яндекса на РОМИП-2006 // Труды РОМИП'2006, октябрь, 2006 . http://download.yandex.ru/company/03_yandex.pdf

[23]. Гэри М., Джонсон Д. Вычислительные машины и труднорешаемые задачи: Пер. с англ. М.: Мир, 1982. – 416 с..

[24]. Даль В.И. Толковый словарь живого, великорусского языка. М., 1955. Т. 1.

[25]. Дунаев С. Java для Internet в Windows и Linux. М.: ДИАЛОГ-МИФИ, 2004. – 496 с. – ISBN 5-86404-188-2.

[26]. Дюбуа П. MySQL, 3-е изд.: Пер. с англ. М.: Издательский дом "Вильямс", 2007. – 1168 с. – ISBN 5-8459-1119-2.

[27]. Жуков В.П., Сидоренко М.И., Шкляров В.Т. Словарь фразеологических синонимов русского языка: Около 730 синоним. рядов/Под ред. В.П. Жукова. М.: Рус. яз., 1987. – 448 с. – ISBN 5-17-027498-X.

[28]. Зыков А.А. Основы теории графов. М.: Наука. Гл. ред. физ.-мат. лит., 1987. – 384 с. – ISBN 5-9502-0057-8.

[29]. Иванов Б.Н. Дискретная математика. Алгоритмы и программы: Учеб. пособие. М.: Лаборатория Базовых Знаний, 2003. – 288 с. – ISBN 5-93208-093-0.

[30]. Искусственный интеллект. - в 3-х кн. Кн. 2. Модели и методы: Справочник / Под ред. Д. А. Поспелова. М.: Радио и связь, 1990. – 304 с.

[31]. Кожунова О.С. Применение правдоподобных рассуждений дсм-метода для пополнения семантического словаря // Компьютерная лингвистика и интеллектуальные технологии. Труды международной конференции «Диалог 2006». Бекасово, 2006. – С. 243-247.

[32]. Кормен Т., Лейзерсон Ч., Ривест Р. Алгоритмы: построение и анализ. М.: МЦНМО, 1999. – 960 с.5-900916-37-5.

[33]. Крижановский А.А. Вопросы реализации проблемно-ориентированных агентов интеграции знаний // Труды СПИИРАН / Под ред. Р.М. Юсупова вып. 1, Т. 3 – СПб.: СПИИРАН, 2002. – С. 31-40. http://whinger.narod.ru/paper/index.html

[34]. Крижановский А.А. Автоматизированное построение списков семантически близких слов на основе рейтинга текстов в корпусе с гиперссылками и категориями // Компьютерная лингвистика и интеллектуальные технологии.

Труды международной конференции «Диалог 2006». Бекасово, 2006. – С. 297-302. http://arxiv.org/abs/cs.IR/0606128

[35]. Крижановский А.А. Оценка результатов поиска семантически близких слов в Википедии: Information Content и адаптированный HITS алгоритм // Вики-конференция 2007. Тезисы докладов. Санкт-Петербург, 2007 . http://arxiv.org/abs/0710.0169

[36]. Крижановский А.А. Оценка результатов поиска семантически близких слов в Википедии // Труды СПИИРАН. Вып. 5. — СПб.: Наука, 2007 – С. 113–116.

[37]. Крижановский А.А. Автоматизированный поиск семантически близких слов на примере авиационной терминологии // *Автоматизация в промышленности*. – 2008. – Т.**4** (64). – С. 16-20. http://whinger.narod.ru/paper/avia_terms_search_by_AHITS_08.pdf

[38]. Кристофидес Н. Теория графов: алгоритмический подход. М.: Мир, 1978. – 215 с.. http://nehudlit.ru/1/45/

[39]. Кулешов С.В. Разработка автоматизированной системы семантического анализа и построения визуальных динамических глоссариев: Автореф. ... дис. канд. техн. наук. – СПб., 2005. – 20 с.

[40]. Ландэ Д.В., Григорьев А.Н., Дармохвал А.Т. Взаимосвязь понятий в документах - совместное появление или контекстная близость? // Компьютерная лингвистика и интеллектуальные технологии. Труды международной конференции «Диалог 2007». Бекасово, 2007 . http://www.dialog-21.ru/dialog2007/materials/pdf/LandeD1.pdf

[41]. Леонтьева Н.Н. Автоматическое понимание текстов: системы, модели, ресурсы: учеб. пособие для студ. лингв. фак. вузов / Нина Николаевна Леонтьева. М.: Издательский центр "Академия", 2006. – 304 с. – ISBN 5-7695-1842-1.

[42]. Макконнелл Дж. Основы современных алгоритмов. 2-е дополненное издание. М.: Техносфера, 2004. – 368 с. – ISBN 5-94836-005-9.

[43]. Мандель И.Д. Кластерный анализ. М.: Финансы и статистика, 1988. – 176 с. – ISBN 5-279-00050-7.

[44]. Меликян Х. Гиперсемантика. "дальше..." - Блокнот недовольного программиста. 2004. http://www.melikyan.com/dalshe/articles/hypersemantics.html

[45]. Нечепуренко М.И., Попков В.К., Майнагашев С.М. и др. Алгоритмы и программы решения задач на графах и сетях. Новосибирск: Наука. Сиб. отд-ние, 1990. – 515 с. – ISBN 5-02-028614-1.

[46]. Новиков Ф.А. Дискретная математика для программистов. Учебник для вузов. 2-е изд. СПб.: Питер, 2004. – 364 с. – ISBN 5-94723-741-5.

[47]. Ножов И.М. Морфологическая и синтаксическая обработка текста (модели и программы): Дис. ... канд. техн. наук. – М., 2003. http://www.aot.ru/technology.html

[48]. Полонников Р.И. Основные концепции общей теории информации. СПб.: Наука, 2006. – 203 с. – ISBN 5-02-025082-1.

[49]. Препарата Ф., Шеймос М. Вычислительная геометрия: Введение: Пер. с англ. М.: Мир, 1989. – 478 с. – ISBN 5-03-001041-6.

[50]. Рахилина Е.В., Кобрицов Б.П., Кустова Г.И., Ляшевская О.Н., Шеманаева О.Ю. Многозначность как прикладная проблема: лексико-семантическая разметка в национальном корпусе русского языка // Компьютерная лингвистика и интеллектуальные технологии. Труды международной конференции «Диалог 2006». Бекасово, 2006. – С. 445-450. http://www.dialog-21.ru/dialog2006/materials/pdf/Rakhilina.pdf

[51]. Репьев А.П. По-ВААЛ-яем дурака, господа! 2007. http://www.repiev.ru/articles/VAAL.htm

[52]. Сегалович И.В. Как работают поисковые системы. http://www.dialog-21.ru/trends/?id=15539&f=1

[53]. Семенов Ю.А. Современные поисковые системы. 2006. http://book.itep.ru/4/45/retr4514.htm

[54]. Семикин В.А. Семантическая модель контента образовательных электронных изданий : Автореф. ... канд. техн. наук: 05.13.18. — Тюмень, 2004. – 21 с. http://orel3.rsl.ru/dissert/Semikin.pdf

[55]. Сичинава Д.В. К задаче создания корпусов русского языка в Интернете, 2001. http://corpora.narod.ru/article.html

[56]. Словарь синонимов русского языка: В 2 т. / АН СССР, Институт русского языка; Под ред. А. П. Евгеньевой. Л.: Наука, Ленинградское отделение, 1970.

[57]. Смирнов А.В., Пашкин М.П., Шилов Н.Г., Крижановский А.А. Реализация взаимодействия агентов в системе логистики знаний «Интеграция» // Информатизация и связь. – 2003. – № 1-2. – С. 74-78.

[58]. Смирнов А.В., Пашкин М.П., Шилов Н.Г., Левашова Т.В., Крижановский А.А. Формирование контекста задачи для интеллектуальной поддержки принятия решений. Фундаментальные основы информационных технологий и систем // Труды Института системного анализа РАН. – М.: ИСА РАН, 2004. – Т. 9. – С. 125-188.

[59]. Сокирко А.В. Семантические словари в автоматической обработке текста (по материалам системы ДИАЛИНГ): Дис. ... канд. техн. наук. – М., 2001. http://www.aot.ru

[60]. Сокирко А.В. Морфологические модули на сайте www.aot.ru // Компьютерная лингвистика и интеллектуальные технологии. Труды международной конференции «Диалог 2004». «Верхневолжский», 2004. – С. 559-564. http://www.aot.ru/docs/sokirko/Dialog2004.htm

[61]. Соловьев В.Д., Добров Б.В., Иванов В.В., Лукашевич Н.В. Онтологии и тезаурусы: Учебное пособие.. Казань, Москва: Казанский государственный университет, МГУ им. М.В. Ломоносова, 2006. – 157 с.. http://window.edu.ru/window_catalog/files/r41722/ot_2006_posobie.pdf

[62]. Торвальдс Л., Даймонд Д. Ради удовольствия. М.: Изд-во ЭКСМО-Пресс, 2002. – 288 с. – ISBN 5-04-009285-7.

[63]. Фридл Дж. Регулярные выражения. Библиотека программиста. СПб.: Питер, 2001. – 352 с. – ISBN 5-272-00331-4.

[64]. Хорошевский В.Ф. Оценка систем извлечения информации из текстов на естественном языке: кто виноват, что делать // Десятая национальная конференция по искусственному интеллекту с международным участием (КИИ-2006). г. Обнинск, 2006. – С. 464-478. http://www.raai.org/resurs/papers/kii-2006/doklad/Khoroshevsky.rar

[65]. Чупановская М. Н. Репрезентация противоположности в семантике производных антонимов (на материале словарей русского языка) : Автореф. ... канд. филол. наук: 10.02.01. — Новосибирск, 2007. — 21 с. http://dissovet.nsu.ru/node/67

[66]. Шемакин Ю.И. Начала компьютерной лингвистики: Учеб. пособие. М.: Иэд-во МГОУ, А/О "Росвузнаука", 1992. – 81 с.– ISBN 5-7045-0132-X. http://sysen.rags.ru

[67]. Шемакин Ю.И. Семантика самоорганизующихся систем. М.: Академический Проект, 2003. – 176 с. – ISBN 5-8291-0168-8. http://sysen.rags.ru

[68]. Шилдт Г. Java 2, v5.0 (Tiger) . Новые возможности: Пер. с англ. СПб.: БХВ-Петербург, 2005. – 208 с. – ISBN 5-94157-643-9, 0-07-225854-3.

[69]. Adler B.T., Benterou J., Chatterjee K., Alfaro L., Pye I., Raman V. Assigning trust to Wikipedia content, Technical Report N. UCSC-CRL-07-08, School of Engineering, University of California, Santa Cruz, CA, USA, 2007. http://www.soe.ucsc.edu/~luca/papers/07/trust-techrep.html

[70]. Albert R., Barabasi A.-L. Statistical mechanics of complex networks. – Reviews of Modern Physics, 2002. – Vol. 74, No. 47. http://arxiv.org/abs/cond-mat/0106096

[71]. Aleman-Meza B., Halaschek C., Arpinar I.B., Ramakrishnan C., Sheth A. Ranking complex relationships on the Semantic Web. – IEEE Internet Computing, 2005. – Vol. 9, No. 3, pp. 37-44. http://lsdis.cs.uga.edu/library/download/AHARS05-Ranking-IC.pdf

[72]. Andrews P., Rajman M. Thematic annotation: extracting concepts out of documents. 2004. http://arxiv.org/abs/cs/0412117

[73]. Avrachenkov K., Litvak N., Son Pham K. Distribution of PageRank mass among principle components of the Web. 2007. http://arxiv.org/abs/0709.2016

[74]. Banerjee S., Pedersen T. An Adapted Lesk algorithm for word sense disambiguation using WordNet. In *Proceedings of the Third International Conference on Intelligent Text Processing and Computational Linguistics (CICLING-02).* Mexico City, February, 2002. http://www.d.umn.edu/~tpederse/Pubs/cicling2002-b.pdf

[75]. Banerjee S., Pedersen T. Extended gloss overlaps as a measure of semantic relatedness. In *Proceedings of the Eighteenth International Joint Conference on Artificial Intelligence IJCAI-2003.* Acapulco, Mexico, August, 2003. http://www.d.umn.edu/~tpederse/Pubs/ijcai03.pdf

[76]. Bar-Ilan J., Mat-Hassan M., Levene M. Methods for comparing rankings of search engine results. 2005. http://arxiv.org/abs/cs/0505039

[77]. Barthelemy M., Chow E., Eliassi-Rad T. Knowledge representation issues in semantic graphs for relationship detection. In *AI Technologies for Homeland Security: Papers from the 2005 AAAI Spring Symposium, AAAI Press*, 2005. – pp. 91-98 http://arxiv.org/abs/cs/0504072

[78]. Batagelj V., Mrvar A. Analysis of large networks with Pajek. In *Pajek workshop at XXVI Sunbelt Conference*Vancouver, BC, Canada, 25-30 April, 2006. http://vlado.fmf.uni-lj.si/pub/networks/Doc/Sunbelt/sunbeltXXVI.pdf

[79]. Bayardo R., Ma Y., Srikant R. Scaling up all pairs similarity search. In *Proc. of the 16th Int'l Conf. on the World Wide Web*, 2007. http://labs.google.com/papers.html

[80]. Bellomi F., Bonato R. Network analysis for Wikipedia. Wikimania 2005. http://www.fran.it/blog/2005/08/network-analisis-for-wikipedia.html

[81]. Bharat K., Henzinger M. Improved algorithms for topic distillation in a hyperlinked environment. In *Proc. 21st International ACM SIGIR Conference on Research and Development in Information Retrieval (SIGIR 98)*, 1998. – pp. 104-111 ftp://ftp.digital.com/pub/DEC/SRC/publications/monika/sigir98.pdf

[82]. Biber D., Conrad S., Reppen R. Corpus linguistics: investigating language structure and use (Cambridge approaches to linguistics). – Cambridge university press, 2002. – 300 pp. – ISBN 0-521-499577.

[83]. Blondel V., Gajardo A., Heymans M., Senellart P., Dooren P. A measure of similarity between graph vertices: applications to synonym extraction and web searching. – SIAM Review, 2004. – Vol. 46, No. 4, pp. 647-666. http://www.inma.ucl.ac.be/~blondel/publications/areas.html

[84]. Blondel V., Senellart P. Automatic extraction of synonyms in a dictionary. In *Proceedings of the SIAM Workshop on Text Mining.* Arlington (Texas, USA), 2002. http://www.inma.ucl.ac.be/~blondel/publications/areas.html

[85]. Brin S., Page L. The anatomy of a large-scale hypertextual Web search engine. 1998. http://www-db.stanford.edu/~backrub/google.html

[86]. Buckley C., Salton G., Allan J., Singhal A. Automatic query expansion using SMART - TREC-3. In *An Overview of the Third Text Retrieval Conference (TREC 3), D.K. Harman, editor. National Institute of Science and Technology. Special Publication.* Gaithersburg, MD, 1995. – pp. 69-80 http://www.cs.cornell.edu/Info/People/singhal/papers/trec3.ps

[87]. Budanitsky A., Hirst G. Semantic distance in WordNet: An experimental, application-oriented evaluation of five measures. In *Workshop on WordNet and Other Lexical Resources, Second meeting of the North American Chapter of the Association for Computational Linguistics.* Pittsburgh, 2001. http://ftp.cs.toronto.edu/pub/gh/Budanitsky+Hirst-2001.pdf

[88]. Cabezas C., Resnik P. Using WSD techniques for lexical selection in statistical machine translation. Technical report CS-TR-4736/LAMP-TR-124/UMIACS-TR-2005-42, July 2005. http://lampsrv01.umiacs.umd.edu/pubs/TechReports/LAMP_124/LAMP_124.pdf

[89]. Calado P., Ribeiro-Neto B. A., Ziviani N., Moura E. S., Silva I. Local versus global link information in the Web. – ACM Transactions on Office and Information Systems (TOIS), 2003. – Vol. 21, No. 1, pp. 42-63. http://homepages.dcc.ufmg.br/~berthier/books_journal_papers/acm_tois_2003.pdf

[90]. Calderan M. Semantic Similarity Library, Technical Report #DIT-06-036, University of Trento, 2006. http://www.dit.unitn.it/~accord/Publications/Semantic%20Similarity%20Library%20Thesis%20Proposal.pdf

[91]. Campbell S., Chancelier J.-P., Nikoukhah R. Modeling and Simulation in Scilab/Scicos. – Springer, 2006. – 313 pp. – ISBN 978-0-387-27802-5.

[92]. Cunningham H., Maynard D., Bontcheva K., Tablan V., Ursu C., Dimitrov M., Dowman M., Aswani N., Roberts I. Developing language processing components with GATE (user's guide), Technical report, University of Sheffield, U.K., 2005. http://www.gate.ac.uk

[93]. Deng Y. The Metadata Architecture for Data Management in Web-based Choropleth Maps. 2002. http://www.cs.umd.edu/hcil/census/JavaProto/metadata.pdf

[94]. Denoyer L., Gallinari P. The Wikipedia XML corpus. In *Advances in XML Information Retrieval and Evaluation: Fifth Workshop of the INitiative for the Evaluation of XML Retrieval (INEX'06)*. Germany, Dagstuhl, 2007. http://www.sigir.org/forum/2006J/2006j_sigirforum_denoyer.pdf

[95]. Ding G., Wang B., Bai S. Robust track: using query expansion and rankfusion to improve effectiveness and robustness of ad hoc information retrieval. 2005. http://trec.nist.gov/pubs/trec14/t14_proceedings.html

[96]. Dorogovtsev S. N., Goltsev A. V., Mendes J. F. F. Critical phenomena in complex networks. 2007. http://arxiv.org/abs/0705.0010

[97]. Doshi P., Thomas C. Inexact Matching of Ontology Graphs Using Expectation-Maximization. In *Proceedings of AAAI, Special track on AI and the Web*, 2006. http://cs.uga.edu/~pdoshi/

[98]. Efimenko I.V., Khoroshevsky V.F., Klintsov V.P. Ontosminer family: multilingual IE systems. In *9-th International Conference "Speech and Computer" SPECOM'2004*. Russia, St. Petersburg, September 20-22, 2004. – pp. 716-720

[99]. Fellbaum C. WordNet: an electronic lexical database. – MIT Press, Cambridge, Massachusetts, 1998. – 423 pp. – ISBN 0-262-06197-X.

[100]. Finkelstein L., Gabrilovich E., Matias Y., Rivlin E., Solan Z., Wolfman G., Ruppin E. Placing search in context: the concept revisited. – ACM Transactions on Information Systems, 2002. – Vol. 20, No. 1, pp. 116-131. http://www.cs.technion.ac.il/~gabr/papers/tois_context.pdf

[101]. Fonseca B. M., Golgher P. B., Possas B., Ribeiro-Neto B., Ziviani N. Concept-based interactive query expansion. In *ACM Conference on Information and Knowledge Management (CIKM)*, 2005. – pp. 696-703 http://homepages.dcc.ufmg.br/~berthier/conference_papers/cikm_2005a.pdf

[102]. Fortunato S., Boguna M., Flammini A., Menczer F. How to make the top ten: Approximating PageRank from in-degree. 2005. http://arxiv.org/abs/cs/0511016

[103]. Gabrilovich E., Markovitch S. Computing semantic relatedness using Wikipedia-based Explicit Semantic Analysis. In *Proceedings of The 20th International Joint Conference on Artificial Intelligence (IJCAI)*. Hyderabad, India, January, 2007. http://www.cs.technion.ac.il/~gabr/papers/ijcai-2007-sim.pdf

[104]. Gayo-Avello D. Building Chinese lexicons from scratch by unsupervised short document self-segmentation. 2004. http://arxiv.org/abs/cs/0411074

[105]. Gentner D. Structure-mapping: A theoretical framework for analogy. – Cognitive Science, 1983. – Vol. 7, No. 1, pp. 155-170. http://www.psych.northwestern.edu/psych/people/faculty/gentner/allpubs.htm

[106]. Geraci F., Pellegrini M. Dynamic user-defined similarity searching in semi-structured text retrieval. 2007. http://arxiv.org/abs/0705.4606

[107]. Gruber T. R. A translation approach to portable ontologies. – Knowledge Acquisition, 1993. – Vol. 5, No. 2, pp. 199-220. http://tomgruber.org/writing/ontolingua-kaj-1993.htm

[108]. Guy M., Tonkin E. Tidying up taggs?. – D-Lib Magazine, 2006. – Vol. 12, No. 1.

[109]. Hamon T., Nazarenko A., Poibeau T., Aubin S., Deriviere J. A robust linguistic platform for efficient and domain specific web content analysis. In *Proceedings of RIAO*, 2007. http://arxiv.org/abs/0706.4375

[110]. Harabagiu S., Moldovan D. A marker-propagation algorithm for text coherence. In *Working Notes of the Workshop on Parallel Processing at the 14th International Joint Conference on Artificial Intelligence.* Montreal, 1995. – pp. 76 - 86 http://www.seas.smu.edu/~sanda/papers/parai.ps.gz

[111]. Hayes C., Avesani P., Veeramachaneni S. An analysis of the use of tags in a blog recommender system. In *Proc. 20th International Joint Conference on Artificial*

Intelligence (IJCAI-07). India, Hyderabad, January 6-12, 2007. – pp. 2772-2778 http://www.ijcai.org/papers07/Papers/IJCAI07-445.pdf

[112]. Hearst M., Karadi C. Cat-a-Cone: an interactive interface for specifying searches and viewing retrieval results using a large category hierarchy. In *Proc. of ACM SIGIR*, 1997. – pp. 246-255

[113]. Hillman D. Using Dublin Core. 2006. http://dublincore.org/documents/usageguide/

[114]. Holloway T., Bozicevic M., Borner K. Analyzing and visualizing the semantic Coverage of Wikipedia and its Authors. 2005. http://arxiv.org/abs/cs/0512085

[115]. Horstmann C. S., Cornell G. Core Java™ 2 volume I - fundamentals, Seventh Edition. – Prentice Hall PTR, 2004. – 784 pp. – ISBN 0-13-148202-5. http://horstmann.com/corejava.html

[116]. Iria J., Ciravegna F. Relation Extraction for Mining the Semantic Web. In *Proceedings of the Dagstuhl Seminar on Machine Learning for the Semantic Web.* Dagstuhl, Germany, February 13-18, 2005. http://tyne.shef.ac.uk/t-rex

[117]. Jarmasz M, Szpakowicz S. Roget's Thesaurus and semantic similarity. In *Proceedings of Conference on Recent Advances in Natural Language Processing (RANLP 2003)*. Borovets, Bulgaria, September, 2003. – pp. 212-219 http://www.nzdl.org/ELKB/

[118]. Jasper R., Uschold M. A framework for understanding and classifying ontology applications. In *Twelfth Workshop on Knowledge Acquisition Modeling and Management KAW'99*, 1999. http://sern.ucalgary.ca/KSI/KAW/KAW99/papers/Uschold2/final-ont-apn-fmk.pdf

[119]. Jeh G., Widom J. SimRank: a measure of structural-context similarity. In *Proceedings of the Multi-Relational Data Mining Workshop, 8th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining.*, 2002. http:// glenjeh.googlepages.com/index.html

[120]. Jiang J. J., Conrath D. W. Semantic similarity based on corpus statistics and lexical taxonomy. In *Proceedings of International Conference Research on Computational Linguistics (ROCLING X)*. Taiwan, 1997. http://xxx.lanl.gov/abs/cmp-lg/9709008

[121]. Karov Y., Edelman S. Learning similarity-based word sense disambiguation from sparse data. To appear in the Fourth Workshop on Very Large Corpora, 1996, Copenhagen. http://xxx.lanl.gov/abs/cmp-lg/9605009

[122]. Karypis G., Han E.-H., Kumar V. Chameleon: a hierarchical clustering algorithm using dynamic modeling. – IEEE Computer: Special Issue on Data Analysis and Mining, 1999. – Vol. 32, No. 8. http://www-users.cs.umn.edu/~karypis/publications/Papers/PDF/chameleon.pdf

[123]. Kashyap V., Ramakrishnan C., Thomas C., Sheth A. TaxaMiner: an experimental framework for automated taxonomy bootstrapping. – International Journal of Web and Grid Services, Special Issue on Semantic Web and Mining Reasoning, 2005. – Vol. 1, No. 2, pp. 240-266. http://lsdis.cs.uga.edu/~amit

[124]. Khoroshevsky V., Efimenko I., Drobyazko G., Kananykina P., Klintsov V., Lisitsin D., Seledkin V., Starostin A., Vorobyov V. Ontos Solutions for semantic web: text mining, navigation and analytics. In *(Gorodetsky, V., Zhang, C., Skormin, V.A., Cao, L. eds.) Autonomous Intelligent Systems: Agents and Data Mining: International Workshop, AIS-ADM 2007*. Springer-Verlag Berlin Heidelberg, Lecture Notes in Artificial Intelligence, Vol. 4476, 2007. – pp. 11-27 http://ontosearch.com

[125]. Kleinberg J. Authoritative sources in a hyperlinked environment. – Journal of the ACM, 1999. – Vol. 5, No. 46, pp. 604-632. http://www.cs.cornell.edu/home/kleinber

[126]. Krizhanovsky A. Synonym search in Wikipedia: Synarcher. In *11-th International Conference "Speech and Computer" SPECOM'2006.* Russia, St. Petersburg, June 25-29, 2006. – pp. 474-477 http://arxiv.org/abs/cs/0606097

[127]. Kroski E. The Hive Mind: Folksonomies and User-Based Tagging. 2006. http://infotangle.blogsome.com/2005/12/07/the-hive-mind-folksonomies-and-user-based-tagging

[128]. Lin D. An information-theoretic definition of similarity. In *Proceedings of International Conference on Machine Learning,* Madison, Wisconsin, July, 1998. http://www.cs.ualberta.ca/~lindek/papers.htm

[129]. Maguitman A. G., Menczer F., Roinestad H., Vespignani A. Algorithmic Detection of Semantic Similarity. 2005. http://www2005.org/cdrom/contents.htm

[130]. Mahadevan P., Krioukov D., Fall K., Vahdat A. A basis for systematic analysis of network topologies. 2006. http://arxiv.org/abs/cs/0605007

[131]. Manning C. D., Schutze H. Foundations of Statistical Natural Language Processing. – The MIT Press, 1999. – 620 pp. – ISBN 0262133601. http://nlp.stanford.edu/fsnlp/

[132]. Melnik S., Garcia-Molina H., Rahm E. Similarity flooding: a Versatile graph matching algorithm and its application to schema matching. In *18th ICDE*. San Jose CA, 2002. http://research.microsoft.com/~melnik/publications.html

[133]. Meyer C.F. English corpus linguistics: An introduction. – Cambridge: Cambridge University Press, 2004. – 168 pp. – ISBN 0-521-00490-X.

[134]. Milne D. Computing Semantic Relatedness using Wikipedia Link Structure. // New Zealand Computer Science Research Student Conference (NZCSRSC'2007). Hamilton, New Zealand. . http://www.cs.waikato.ac.nz/~dnk2/publications/nzcsrsc07.pdf

[135]. Milne D., Medelyan O., Witten I. H. Mining domain-specific thesauri from Wikipedia: a case study. In *Proc. of the International Conference on Web Intelligence (IEEE/WIC/ACM WI'2006)*. Hong Kong, 2006. http://www.cs.waikato.ac.nz/~olena/publications/milne_wikipedia_final.pdf

[136]. Mizzaro S., Robertson S. HITS hits TREC - exploring IR evaluation results with network analysis. In *SIGIR 2007. ACM*, 2007. – pp. 479-486 http://www.soi.city.ac.uk/~ser/papers/MizzaroRobertsonSIGIR07.pdf

[137]. Mladenic D., Grobelnik M. Mapping documents onto web page ontology. – In Berendt B., Hotho A., Mladenic D., Van Someren M.W., Spiliopoulou M., Stumme G. (Eds), Web Mining: From Web to Semantic Web, Lecture Notes in Artificial Intelligence: Lecture Notes in Computer Science, 2004. – Vol. 3209, No. , pp. 77-96. http://eprints.pascal-network.org/archive/00000840/

[138]. Montoyo A., Palomar M., Rigau G. Method for WordNet enrichment using WSD. In *Proceedings of 4th International Conference on Text Speech and Dialogue TSD'2001. Selezna Ruda - Spieak, Czech Republic. Published in Lecture Notes in Artificial Intelligence 2166, Springer-Verlag*, 2001. http://www.lsi.upc.es/~nlp/papers/2001/tsd01-mpr.ps.gz

[139]. Muller C., Gurevych I., Muhlhauser M. Integrating semantic knowledge into text similarity and information retrieval. In *Proceedings of the First IEEE International Conference on Semantic Computing (ICSC)*, 2007. – pp. (to appear) http://proffs.tk.informatik.tu-darmstadt.de/TK/abstracts.php3?lang=en&paperID=575

[140]. Muller C., Meuthrath B., Baumgrass A. Analyzing wiki-based networks to improve knowledge processes in organizations. – Journal of Universal Computer

Science, 2008. – Vol. 14, No. 4, pp. 526-545. http://www.jucs.org/jucs_14_4/analyzing_wiki_based_networks

[141]. Nazarenko A., Alphonse E., Deriviere J., Hamon T., Vauvert G. The ALVIS Format for Linguistically Annotated Documents. In *Proceedings of the fifth international conference on Language Resources and Evaluation, LREC*, 2006. – pp. 1782-1786 http://arxiv.org/abs/cs/0609136

[142]. Newman M. E. J. The structure and function of complex networks. 2003. http://arxiv.org/abs/cond-mat/0303516

[143]. North Carolina ECHO (Exploring Cultural Heritage Online: Guidelines for Digitization). Ed.: K. M. Wisser, 2006. http://www.ncecho.org/Guide/metadata.htm

[144]. O'Madadhain J., Fisher D., Smyth P., White S., Boey Y.-B. Analysis and visualization of network data using JUNG (preprint). – Journal of Statistical Software. pp. 1-35. 2007. http://jung.sourceforge.net/doc/JUNG_journal.pdf

[145]. Ollivier Y., Senellart P. Finding related pages using Green measures: an illustration with Wikipedia. In *Association for the Advancement of Artificial Intelligence.*Vancouver, Canada, 2007. http://pierre.senellart.com/publications/ollivier2006finding.pdf

[146]. Pantel P., Lin D. Word-for-word glossing with contextually similar words. In *Proceedings of ANLP-NAACL 2000.* Seattle, Washington, May, 2000. – pp. 75-85 http://www.cs.ualberta.ca/~lindek/papers.htm

[147]. Pedersen T. Computational approaches to measuring the similarity of short contexts: a review of applications and methods. – South Asian Language Review (to appear), 2008. – Vol. 1, No. 1. http://arxiv.org/abs/0806.3787

[148]. Pedersen T., Pakhomov S., Patwardhan S., Chute C. Measures of semantic similarity and relatedness in the biomedical domain. – Journal of Biomedical Informatics, 2007. – Vol. 40, No. 3, pp. 288-299. http://www.d.umn.edu/~tpederse/Pubs/jbi2007.pdf

[149]. Ponzetto S. P., Strube M. An API for measuring the relatedness of words in Wikipedia. In *Companion Volume to the Proceedings of the 45th Annual Meeting of the Association for Computational Linguistics.* Prague, Czech Republic, 23-30 June, 2007. http://www.eml-research.de/english/homes/ponzetto/pubs/acl07.pdf

[150]. Ponzetto S., Strube M. Exploiting semantic role labeling, WordNet and Wikipedia for coreference resolution. In *Proceedings of the Human Language*

Technology Conference of the North American Chapter or the Associaton for Computational Linguistics (HLT-NAACL 06). New York City, N.Y., June 4-9, 2006. – pp. 192-199 http://www.eml-research.de/english/research/nlp/publications.php

[151]. Resnik P. Disambiguating noun groupings with respect to WordNet senses. In *Proceedings of the 3rd Workshop on Very Large Corpora.* MIT, June, 1995. http:// xxx.lanl.gov/abs/cmp-lg/9511006

[152]. Resnik P. Semantic similarity in a taxonomy: an information-based measure and its application to problems of ambiguity in natural language. – Journal of Artificial Intelligence Research (JAIR), 1999. – Vol. 11, No. , pp. 95-130. http://www.cs.washington.edu/research/jair/abstracts/resnik99a.html

[153]. Resnik P., Yarowsky D. Distinguishing systems and distinguishing senses: new evaluation methods for word sense disambiguation. – Natural Language Engineering, 2000. – Vol. 5, No. 2, pp. 113-133. http://www.cs.jhu.edu/~yarowsky/pubs.html

[154]. Rigau G., Atserias J., Agirre E. Combining unsupervised lexical knowledge methods for word sense disambiguation. In *Proceedings of joint 35th Annual Meeting of the Association for Computational Linguistics and 8th Conference of the European Chapter of the Association for Computational Linguistics ACL/EACL'97,* Madrid, Spain, 1997. http://www.lsi.upc.es/~nlp/papers/1997/acl97-raa.ps.gz

[155]. Robertson S. Understanding inverse document frequency: on theoretical arguments for IDF. – Journal of Documentation, 2004. – Vol. 60, No. , pp. 503-520. http://www.soi.city.ac.uk/~ser/idfpapers/Robertson_idf_JDoc.pdf

[156]. Robertson S., Zaragoza H. On rank-based effectiveness measures and optimization. – Information Retrieval, 2007. – Vol. 10, No. , pp. 321-339. http://www.soi.city.ac.uk/~ser/papers/new_optimisation_final.pdf

[157]. Rosenzweig R. Can history be open source? Wikipedia and the future of the past. – The Journal of American History, 2006. – Vol. 93, No. 1, pp. 17-46. http://chnm.gmu.edu/resources/essays/d/42

[158]. Ruiz-Casado M., Alfonseca E., Castells P. Automatic assignment of Wikipedia encyclopedic entries to WordNet synsets. 2005. http://arantxa.ii.uam.es/~castells/publications/awic05.pdf

[159]. Sahami M., Heilman T. D. A web-based kernel function for measuring the similarity of short text snippets. In *Proceedings of the 15th International World Wide Web Conference (WWW)*, 2006. http://robotics.stanford.edu/users/sahami/papers-dir/www2006.pdf

[160]. Schmitz C., Hotho A., Jäschke R., Stumme G. Mining association rules in folksonomies. In *Proc. IFCS 2006 Conference*. Ljubljana, July, 2006. – pp. 261-270 http://www.kde.cs.uni-kassel.de/hotho/pub/2006/schmitz2006asso_ifcs.pdf

[161]. Schone P. Toward knowledge-free induction of machine-readable dictionaries. Ph.D., University of Colorado at Boulder, 2001. http://hometown.aol.com/boisebound/family/publications/DPFV.pdf.gz

[162]. Serrano M.A., Maguitman A., Boguna M., Fortunato S., Vespignani A. Decoding the structure of the WWW: facts versus sampling biases. 2006. http://arxiv.org/abs/cs/0511035

[163]. Shi Z., Gu B., Popowich F., Sarkar A. Synonym-based expansion and boosting-based re-ranking: a two-phase approach for genomic information retrieval. Simon Fraser University, 2005. http://trec.nist.gov/pubs/trec14/t14_proceedings.html

[164]. Shvaiko P., Euzenat J. A Survey of schema-based matching approaches. Journal on Data Semantics, 2005. http://www.ontologymatching.org

[165]. Sima J., Schaeffer S.E. On the NP-completeness of some graph cluster measures. 2005. http://arxiv.org/abs/cs/0506100

[166]. Sinha R., Mihalcea R. Unsupervised graph-based word sense disambiguation using measures of word semantic similarity. In *Proceedings of the IEEE International Conference on Semantic Computing (ICSC)*. Irvine, CA, September, 2007. http://www.cs.unt.edu/~rada/papers/mihalcea.naacl07.pdf

[167]. Smirnov A., Krizhanovsky A. Information filtering based on wiki index database. In *Proceedings of the 8th International FLINS Conference on Computational Intelligence in Decision and Control.* Spain, Madrid, September 21 – 24, 2008. http://arxiv.org/abs/0804.2354

[168]. Smirnov A., Krizhanovsky A., Roy R., Kerr C. A multi-agent system architecture for requirements management in the extended enterprise. In *Proceedings of CE2004: 11th ISPE International Conference on Concurrent Engineering, Research and Applications*, Beijing, China, July, 2004. – pp. 235-240 http://whinger.narod.ru/paper/index.html

[169]. Smirnov A., Levashova T., Pashkin M., Chilov N., Krizhanovsky A., Kashevnik A., Komarova A. Context-sensitive access to e-document corpus // Труды международной конференции «Корпусная лингвистика–2006». – СПб.: Изд-во С.-Петерб. ун-та, 2006. – С. 360-364. http://arxiv.org/abs/cs/0610058

[170]. Smirnov A., Pashkin M., Chilov N., Levashova T., Krizhanovsky A. High-level business intelligence service in networked organizations. In *Abstracts of e-Business Research Forum eBRF 2003*. Tampere, Finland, 2003. – pp. 37-39

[171]. Smirnov A., Pashkin M., Chilov N., Levashova T., Krizhanovsky A. Free text user request processing in the system "KSNet". In *Proceedings of the 9th International Conference "Speech and Computer"*. St.Petersburg, Russia, 2004. – pp. 662-665

[172]. Smirnov A., Pashkin M., Chilov N., Levashova T., Krizhanovsky A., Kashevnik A. Ontology-based users and requests clustering in customer service management system. In *(Gorodetsky, V., Liu, J., Skormin, V., eds.) Autonomous Intelligent Systems: Agents and Data Mining: International Workshop, AIS-ADM 2005*. Springer-Verlag GmbH, Lecture Notes in Computer Science, Vol. 3505, 2005. – pp. 231-246 http://arxiv.org/abs/cs.IR/0501077

[173]. Strube M., Ponzetto S. WikiRelate! Computing semantic relatedness using Wikipedia. In *Proceedings of the 21st National Conference on Artificial Intelligence (AAAI 06)*. Boston, Mass., July 16-20, 2006. http://www.eml-research.de/english/research/nlp/publications.php

[174]. Survey of text mining: clustering, classification, and retrieval, M. Berry (Ed.). – Springer-Verlag, New York, 2003. – 244 pp. – ISBN 0-387-955631.

[175]. Teich E., Fankhauser P. WordNet for lexical cohesion analysis. In *Proceedings of the Second Global WordNet Conference.* Brno, Czech Republic, January 20-23, 2004. – pp. 326-331 http://www.fi.muni.cz/gwc2004/proc/77.pdf

[176]. Thom J. A., Pehcevski J., Vercoustre A.-M. Use of Wikipedia categories in entity ranking. In *12th Australasian Document Computing Symposium (ADCS'07)*, 2007. http://arxiv.org/abs/0711.2917

[177]. Turney P.D. Mining the Web for synonyms: PMI-IR versus LSA on TOEFL. In *Proceedings of the Twelfth European Conference on Machine Learning (ECML-2001).* Freiburg, Germany, 2001. – pp. 491-502 http://arxiv.org/abs/cs.LG/0212033

[178]. Turney P.D. Expressing implicit semantic relations without supervision. In *Proceedings of the 21st International Conference on Computational Linguistics and 44th Annual Meeting of the Association for Computational Linguistics (ACL-06)*. Sydney, Australia, 2006. – pp. 313-320 http://arxiv.org/abs/cs/0607120

[179]. Turney P.D. Similarity of semantic relations. – Computational Linguistics, 2006. – Vol. 32, No. 3, pp. 379-416. http://arxiv.org/abs/cs/0608100

[180]. Turney P.D., Littman M.L., Bigham J., Shnayder V. Combining independent modules to solve multiple-choice synonym and analogy problems. In *Proceedings of the International Conference on Recent Advances in Natural Language Processing (RANLP-03)*. Borovets, Bulgaria, 2003. – pp. 482-489 http://arxiv.org/abs/cs.CL/0309035

[181]. Uschold M., Gruninger M. Ontologies: principles, methods, and applications. – Knowledge Engineering Review, 1996. – Vol. 11, No. 2, pp. 93-155. http://citeseer.ist.psu.edu/uschold96ontologie.html

[182]. Vercoustre A.-M., Thom J. A., Pehcevski J. Entity ranking in Wikipedia. In *Proceedings of the 23rd Annual ACM Symposium on Applied Computing (SAC08)*, 2008. http://arxiv.org/abs/0711.3128

[183]. Volkel M., Krotzsch M., Vrandecic D., Haller H., Studer R. Semantic Wikipedia. In *Proceedings of the 15th International Conference on World Wide Web. WWW '06. ACM Press, New York, NY*. Edinburgh, Scotland, May 23 - 26, 2006. – pp. 585-594 http://www2006.org/programme/item.php?id=4039

[184]. Voss J. Collaborative thesaurus tagging the wikipedia way. Collaborative Web Tagging Workshop. 2006. http://arxiv.org/abs/cs/0604036

[185]. Widdows D. and Dorow B. A graph model for unsupervised lexical acquisition. In *19th International Conference on Computational Linguistics.*Taipei, 2002. – pp. 1093-1099 http://infomap.stanford.edu/graphs/

[186]. Wu Z., Palmer M. Verb semantics and lexical selection. In *Proc. of ACL-94*, 1994. – pp. 133-138 http://acl.ldc.upenn.edu/P/P94/P94-1019.pdf

[187]. Yarowsky D. Unsupervised word sense disambiguation rivaling supervised methods. In *Proceedings of the 33rd Annual Meeting of the Association for Computational Linguistics.* Cambridge, MA, 1995. – pp. 189-196 http://www.cs.jhu.edu/~yarowsky/pubs.html

[188]. Zaidman A., Rompaey B., Demeyer S., Deursen A. On how developers test open source software systems. 2007. http://arxiv.org/abs/0705.3616

[189]. Zesch T., Mueller C., Gurevych I. Extracting lexical semantic knowledge from Wikipedia and Wiktionary. In *Proceedings of the Conference on Language Resources and Evaluation (LREC)*, 2008. http://elara.tk.informatik.tu-darmstadt.de/publications/2008/lrec08_camera_ready.pdf

[190]. Zhdanova A., Shvaiko P. Community-driven ontology matching. In *Proceedings of ESWC'06, LNCS 4011*, 2006. – pp. 34-49 http://dit.unitn.it/~knowdive/index.php?idx=pubs

Приложение 1. Список наиболее употребительных сокращений

АОТ	– автоматическое обработка текста
БД	– база данных
ВП	– Википедия
ИПС	– информационно-поисковая система
ПО	– предметная область
СБС	– семантически близкие слова
AHITS	– Adapted Hyperlink-Induced Topic Selection
IE	– Information Extraction
SEW	– Simple English Wikipedia
RW	– Russian Wikipedia
WSD	– Word Sense Disambiguation

Приложение 2. Акты внедрения

РОССИЙСКАЯ АКАДЕМИЯ НАУК

Учреждение Российской академии наук
Санкт-Петербургский институт
информатики и автоматизации РАН
(СПИИРАН)

199178, Санкт-Петербург, 14 линия, 39
Телефон: (812)328-33-11
Факс: (812)328-44-50
E-mail: spiiran@iias.spb.su
http://www.spiiras.nw.ru

УТВЕРЖДАЮ
Директор СПИИРАН
д.т.н. член-корреспондент РАН
Р.М. Юсупов
«__» 2008 г.

А К Т

об использовании результатов кандидатской диссертационной работы соискателя Крижановского А. А. «Математическое и программное обеспечение построения списков семантически близких слов на основе рейтинга вики-текстов» в международном проекте “Information modelling for multi-lingual system development across the Extended Enterprise and Multi-agent systems”, совместный исследовательский проект с университетом Крэнфилда (Cranfield University), г. Крэнфилд, Великобритания.

Комиссия в составе: председателя Б.В. Соколова, членов комиссии: И. П. Поднозовой, Д. В. Бакурадзе, рассмотрев представленные материалы:

1. Диссертационную работу Крижановского А. А.
2. Отчеты по выполнению международном проекта “Information modelling for multi-lingual system development across the Extended Enterprise and Multi-agent systems”

установила, что:

1. Основные положения диссертационной работы Крижановского А. А. были использованы при проведении плановых научно-исследовательских работ в ходе международного проекта “Information modelling for multi-lingual system development across the Extended Enterprise and Multi-agent systems”.
2. В рамках проекта построена клиент-серверная архитектура программного комплекса поиска семантически близких слов с возможностью оценки списков слов на основе удалённого доступа к тезаурусам (WordNet, Moby).

Председатель комиссии,
Зам. директора по научной работе, д.т.н. проф. Б.В. Соколов

Члены комиссии
Помощник по международным связям И.П. Поднозова
Ученый секретарь к.т.н. с.н.с Д.В. Бакурадзе

РОССИЙСКАЯ АКАДЕМИЯ НАУК

Учреждение Российской академии наук
Санкт-Петербургский институт
информатики и автоматизации РАН
(СПИИРАН)

199178, Санкт-Петербург, 14 линия, 39
Телефон: (812)328-33-11
Факс: (812)328-44-50
E-mail: spiiran@iias.spb.su
http://www.spiiras.nw.ru

УТВЕРЖДАЮ
Директор СПИИРАН
д.т.н. член-корреспондент РАН
Р.М. Юсупов
«__» 2008 г.

А К Т

об использовании результатов кандидатской диссертационной работы соискателя Крижановского А. А. «Математическое и программное обеспечение построения списков семантически близких слов на основе рейтинга вики-текстов» в проекте «Интеллектуальный доступ к каталогам и документам», нацеленном на создание системы поддержки клиентов, реализованной для немецкой промышленной компании Фесто.

Комиссия в составе: председателя Б. В. Соколова, членов комиссии: И. П. Поднозовой и Д. В. Бакурадзе, рассмотрев представленные материалы:

1. Диссертационную работу Крижановского А. А.
2. Отчеты по выполнению проекта “Интеллектуальный доступ к каталогам и документам”

установила, что:

1. Основные положения диссертационной работы Крижановского А. А. были использованы при проведении плановых научно-исследовательских работ в ходе проекта “Интеллектуальный доступ к каталогам и документам”.
2. В рамках проекта использована клиент-серверная архитектура программного комплекса для поиска семантически близких слов на основе данных энциклопедии Википедия.

Председатель комиссии,
Зам. директора по научной работе, д.т.н. проф. Б.В. Соколов

Члены комиссии
Помощник по международным связям И.П. Поднозова
Ученый секретарь к.т.н. с.н.с Д.В. Бакурадзе

РОССИЙСКАЯ АКАДЕМИЯ НАУК

Учреждение Российской академии наук
Санкт-Петербургский институт
информатики и автоматизации РАН
(СПИИРАН)

199178, Санкт-Петербург, 14 линия, 39
Телефон: (812)328-33-11
Факс: (812)328-44-50
E-mail: spiiran@iias.spb.su
http://www.spiiras.nw.ru

УТВЕРЖДАЮ
Директор СПИИРАН
д.т.н. член-корреспондент РАН
Р.М. Юсупов
«___» 2008 г.

А К Т

об использовании результатов кандидатской диссертационной работы соискателя Крижановского А. А. «Математическое и программное обеспечение построения списков семантически близких слов на основе рейтинга вики-текстов» в исследовательском проекте CRDF № RUM2-1554-ST-05 «Онтолого-управляемая интеграция информации из разнородных источников для принятия решений» (задача 1)

Комиссия в составе: председателя Б. В. Соколова, членов комиссии: И. П. Поднозовой и Д. В. Бакурадзе, рассмотрев представленные материалы:

1. Диссертационную работу Крижановского А. А.
2. Отчеты по международному проекту CRDF № RUM2-1554-ST-05

установила, что:

1. Основные положения диссертационной работы Крижановского А. А. были использованы при проведении плановых научно-исследовательских работ в ходе международного проекта CRDF № RUM2-1554-ST-05 (задача 1).
2. Адаптированный HITS алгоритм для поиска семантически близких слов был использован в системе KSNet для расширения / переформулировки запросов пользователя.
3. Коэффициент Спирмена модифицирован для численного сравнения списков слов, что позволило осуществить эксперименты и проверить правильность и эффективность предложенных решений в рамках данного проекта.

Председатель комиссии,
Зам. директора по научной работе, д.т.н. проф. Б.В. Соколов

Члены комиссии
Помощник по международным связям И.П. Поднозова
Ученый секретарь к.т.н. с.н.с Д.В. Бакурадзе

Приложение 3. Экспериментальные данные программы Synarcher

Полный список семантически близких слов, построенный программой Synarcher представлен в табл. 1. Поиск выполнялся при следующих параметрах:

- размер корневого набора:200;
- инкремент:17;
- чёрный список категорий:Страны|Века|Календарь|География_России|Люди;
- ограничение сверху длины строящегося списка слов:100;
- погрешность для останова итераций: 0.01.

Таблица 1

Полный список[1] семантически близких слов, построенный программой Synarcher

Жаргон Слово|Арго|Матерщина|Эвфемизм|Просторечие|ЗЫ|Ака|Диалектология|Сленг|Франц.| Аниме
Истина[2] Философия|Религия|Математика|Христианство|Искусство|Физика|Логика|Теология| Химия|Биология|История|Медицина|Натурфилософия|Мифология|Идеология|Экономика| Механика|Теория|Психология|Филология|Мировоззрение|Современность|Постмодерн| Мистицизм|Вселенная|Викиновости|США|Диалектика|Астрономия|Космология|Гипотеза| Право|Демокрит|Социология|Информатика|Магия|Гносеология|Астрофизика|Космогония| Богословие|Космос|Эмпиризм|Атом|Экология|Абстракция|Агностицизм|Алгебра| Лингвистика|Схоластика|Мораль|Дедукция|Образование|Эксперимент|Антропология| Средневековье|Каббала|Материаловедение|Техника|Язык|Гравитация|Хаос|Геометрия| Криптография|Геология|СССР|Звезда|Оптика|Алхимия|Лженаука|Кибернетика| Архитектура|Электрон|Астрология|Иммунология|Фрактал|Пространство-время| Псевдонаука|Возрождение|Марксизм-ленинизм|Индукция|Космонавтика|Робототехника| Галактика|Нейтрон|Бионика|Парапсихология|Политология|Радиоактивность|Технология| ДНК|Электротехника|Компьютер|Полупроводник|Нумерология|Электроника| Портал:Наука|Культурология|Нанотехнология|Шизофрения|Свет
Самолёт Вертолёт|Аэростат|Планер|Мускулолёт|Автожир|Винтокрыл|Турболёт|Экраноплан|

1 Полный список, то есть без фильтрации экспертом

2 Использовалась программа Synarcher версии 0.12.1

Махолёт|Экранолёт|Викисклад|Авиация|Атмосфера|Воздух|Водород|Винт|СССР|Гелий|США|Газ|Аэропорт|Фарнборо|Пулемёт|DARPA|Давление|Сибирь|Радио|Киловатт|Движитель|А-50|Дирижабль|Шарльер|Монгольфьер|Велосипед|Двигатель|Конвертоплан|Педаль|ОКБ|Вектор|Тангаж|Крен|Пожар|Артиллерия|Эверест|Фенестрон|Спецназ|Разведка|Медицина|Ка-50|Москва|ИКАО|Феодосия|Boeing|Ан-225|Мореходность|Каспийск|ИМО|Амфибия|Дельтаплан|Параплан|Катапульта|Ива|Шёлк|Фюзеляж|Ла-Манш|Икар|Бензин|Инфраструктура|Скорость|Самолет|ПВО|Космонавтика|Керосин|Судно|Энергия|Гироскоп|Корабль|Техника
Сюжет Философия|Наука|Искусство|Религия|История|Идеология|Бог|Христианство|Литература|Культура|Поэзия|Античность|Трагедия|Эпос|Илиада|Поэт|Одиссея|Символизм|Аллегория|Ирония|Общество|Мировоззрение|Драма|Католицизм|Фольклор|Романтизм|Личность|Грех|Гёте|Мистика|Символ|Человечество|Персонаж|Евангелие|Имя|Катастрофа|Фантастика|Пролетариат|Притча|Бессмертие|Эстетика|Познание|Абстракция|Викицитатник|Эмоция|Семиотика|Скульптура|Абстракционизм|Документ|Художник|Письменность|Цвет|Мифология|Натурфилософия|Образ|Реализм|Мотив|Интрига|Шекспир|Сатира|Сказка|Агиография|Проза|Антропоморфизм|Жанр|Метод|Автор|Текст|Знак|XVII|Миф|Событие|Атеизм|Аноним|Неоязычество|Викисклад|Живопись|Фотография|Кинематограф|Метафора|Реклама|Изображение|Орнамент|Герой|Ритм|Комедия|Кино|Драматургия|Шиллер|Трилогия|Басня|Компьютер|Минерал|Саундтрек|Баба-Яга

Приложение 4. Упорядочение списков с помощью респондентов

Задача упорядочения списка семантически близких слов была решена с помощью привлечения респондентов, носителей русского языка. Им был представлен список слов (графа «*Слова»* в табл. 1) и поставлена задача «упорядочить список слов с тем, чтобы наиболее близкие по значению слова были в начале списка». Такое ранжирование позволяет соотнести каждому слову в списке – некоторое целое число – его ранг, порядковый номер в списке. Чем меньше ранг у слова, тем оно ближе по значению к исходному.

Если респондент затруднялся указать для некоторых слов их положение относительно других, то таким словам присваивался максимальный ранг в списке (например, см. столбец «*Э4»* в табл. 1).

Таблица 1

Упорядочение респондентами списка слов семантически близких слову *Истина*

Эксперт N / ***Слово***	***1***	***2***	***3***	***4***	***5***	***6***	***7***	***8***	***9***	***10***	***11***	***12***	***13***	***14***	***15***	***16***	***Среднее***
Авторитет	3	7	8	9	8	7	7	8	5	9	6	6	9	5	6	7	6,88
Бог	9	6	1	9	1	8	8	1	1	9	6	2	9	5	6	8	5,56
Вера	7	8	3	9	9	3	9	9	5	9	6	2	9	6	6	9	6,81
Действительность	5	1	6	3	5	5	2	7	1	5	1	5	3	2	2	4	3,56
Догмат	8	9	9	9	7	9	6	5	2	9	2	2	9	5	6	5	6,38
Знание	6	5	4	9	6	1	4	4	2	9	2	3	4	3	6	6	4,63
Правда	2	3	2	9	2	2	5	2	1	1	6	1	1	2	1	1	2,56
Реальность	4	4	7	2	3	4	1	6	1	3	1	6	2	2	2	3	3,19
Факт	1	2	5	1	4	6	3	3	1	3	1	6	5	2	2	2	2,94

Наличие такой численной оценки положения слов в списке, позволяет их усреднить (графа «*Среднее*»), то есть упорядочить на основе оценок экспертов. Таким образом, упорядоченный список семантически близких слов для слова *Истина* будет такой: *Правда, Факт, Реальность, Действительность, Знание, Бог, Догмат, Вера, Авторитет*.

Результат упорядочения списков (тем же способом) для слов *Жаргон*, *Самолёт*, *Сюжет* представлен в таблице 4.3.

Приложение 5. Википедия

Три базовых принципа формирования Википедии:

1. *NPOV* (Neutral Point Of View) – представление содержания статей с нейтральной точки зрения. На спорные и конфликтные темы представляются все значимые точки зрения без навязывания своего мнения. Прения по некоторому вопросу обсуждаются, оцениваются, но авторы статей в них не вовлекаются. Необходимо холодное, честное, аналитическое описание вопроса.

2. *Verifiability* – возможность проверить материал, наличие ссылок на достоверные источники, на опубликованные материалы.

3. *No original research* – Википедия – это не первоисточник информации. Цитирование источников позволяет проверить надёжность информации.

Отношения в Википедии

MediaWiki (оболочка Wikipedia) предоставляет механизм категорий, позволяющий классифицировать статьи и другие страницы в проектах Wikimedia. Категории выбираются и присваиваются статьям в ходе совместной работы пользователей (так называемый процесс collaborative tagging, другое название – folksonomy[1]. Страница категории представляет из себя (i) заголовок – название категории, (ii) краткое описание назначения

1 «Folksonomy – это создаваемая совместными усилиями, открытая для расширения система меток, позволяющая пользователям Интернет категоризовать информационные ресурсы, такие как Интернет страницы, онлайн фотографии, интернет ссылки. Использование меток (другое название – теги, аналог ключевых слов), свободно выбираемых пользователем, позволяет улучшить работу поисковых систем, поскольку для категоризации применяется знакомый, понятный, используемый пользователем лексикон» (http://en.wikipedia.org/wiki/Folksonomy). Подход folksonomy преследует цели: создание личной коллекции [ссылок] и общественной коллекции [108] (цит. по [184]). В работе [160] представлена формальная модель folksonomy, как набора троек (пользователь, тег, ресурс) или трёхдольного графа. Особенность Википедии в том, что некоторая категория присваивается статье раз и навсегда для всех пользователей [184]. Примерами folksonomy систем могут служит del.icio.us (открытое хранилище закладок), в которой пользователи придумывают и присваивают теги интернет страницам, и flickr – где пользователи-фотографы присваивают теги фотографиям [127]. Таким образом, теги нужны для того, чтобы пользователи могли быстро *повторно* найти некоторую информацию, помеченную тегами.

категории (аналог глоссы в тезаурусе)[1], (iii) список названий статей, имеющих данную категорию. Закономерно поэтому, что систему категорий называют тезаурусом с иерархическими отношениями между категориями [184]. Иерархические отношения между категориями возможны, поскольку категории могут быть присвоены не только статьям, но и самим категориям в Википедии. Категории используются в поисковых алгоритмах [176], [182].

В энциклопедии Википедия (и вообще в ресурсах, построенных на основе системы MediaWiki) представлены отношения эквивалентности, иерархии и ассоциативные отношения (табл. 1), связывающие друг с другом статьи и категории.

Таблица 1

Виды отношений в Википедии (адаптированно из [184])

Отношение	**Обозначение**	**В терминах MediaWiki**
Эквивалентность (синонимия)	USE USE FOR	Перенаправление (redirect)
Иерархия	Broader Term Narrow Term	Категории данной категории Подкатегории данной категории
Ассоциативность	Related Term	Ссылки между категориями

В Википедии активно используется механизм перенаправлений, или иначе отношение эквивалентности[2]. Механизм перенаправлений позволяет решить такие проблемы, как заглавные/сточные буквы в заголовке статьи, разные варианты написания заглавного слова, аббревиатуры, синонимы[3], разговорные выражения, научная терминология [135].

Пример иерархической цепочки категорий (Broader Term) для категории «Шифр» – «Криптография» – «Защита информации» –

1 Правила Википедии рекомендуют, чтобы у категорий не было аннотаций, то есть названия категорий должны быть ясными и не требующими пояснений. Возразим, что в редких случаях (особенно, когда нет основной статьи, описывающей понятия категории) аннотации нужны, например, когда категории имеют научное название, см. например, http://ru.wiktionary.org/wiki/Категория:Эпистемические_глаголы

2 В русской Википедии есть такие примеры эквивалентности, как: перенаправление со статьи «Бр**о**узер» на статью «Бр**а**узер», «Астронавт» – «Космонавт», «Космодром Байконур» – «Байконур», «Космос (астрономия)» – «Вселенная», «Linux» – «GNU/Linux».

3 Обратную проблему – проблему многозначности в Википедии решают с помощью специальных страниц, содержащих перечисление значений для данного слова со ссылка на соответствующие страницы.

«Информатика» и т.д. Пример иерархической цепочки подкатегорий (Narrow Term) для категории «Криптография» – «Аутентификация» – «Биометрия».

Гиперссылки – это основной способ навигации в Веб, они же связывают как страницы, так и категории в Википедии. Ассоциативные отношения между категориями определяются наличием обычных ссылок между страницами-категориями [184]. В работе [103] различают понятия *related terms* (семантически связанные, близкие по значению слова) и *similar terms* – семантически сходные, сходные по значению слова (в основном синонимы). Таким образом, понятие *Semantic relatedness* шире, чем *Semantic similarity*, так как сюда включаются (кроме синонимии) ещё отношения: меронимии, антонимии и др. [103].

К какому виду отношений отнести ссылки между статьями энциклопедии? Поскольку между собой могут быть связаны самые разные статьи[1], а наличие связи определяется общностью контекста статей, постольку указать жёстко какой-то один тип отношений, связывающий статьи, было бы не верно. Следующим шагом в развитии Википедии, как технологии семантической паутины[2], является: (i) указание *типа ссылок* между статьями, (ii) указание *типа данных* внутри статей. Результаты разработки такого семантического расширения, встраиваемое в ВП (Semantic Wikipedia), представлены в работе [183] и доступны в интернет[3].

---

1 В работе [135] указывают на проблему гиперссылок, на то, что часто в энциклопедической статье указаны ссылки на статьи, которые незначительно, слабо связаны с исходной статьёй. Это проблема для поисковых систем, которые выполняют поиск на основе анализа гиперссылок. Учёные из Новой Зеландии [135] предлагают рассматривать только взаимные ссылки. Я бы предложил следующее: рассматривать и не взаимные, но взаимным давать приоритет. Какой приоритет? Это уже будет зависеть от алгоритма. Например в HITS – включать в корневой набор только вершины, взаимносвязанные с исходной вершиной, в базовый – включать вершины с не только взаимными ссылками.

2 «Semantic Web – это расширения Веб, позволяющие выполнять автоматическую (и ручную) обработку данных вычислительным системам (и человеку). Расширения обеспечат: (i) представление данных в машинно-читаемой форме, описывающих содержание веб-страниц, (ii) возможность пользователям указывать отношения (например, семантические) между различными типами данных». http://www.w3.org/2001/sw/SW-FAQ

3 См. программу (http://ontoworld.org) и описание (http://ru.wikipedia.org/wiki/Семантическая_вики).

Замечания о категориях и ссылках Википедии

Названия страниц в формате вики состоят из двух частей: пространство имён (необязательная часть) и собственно название. Например, статья «Шифр» имеет страницу обсуждения с заголовком [[Обсуждение:Шифр]][1], в которой пространству имён соответствует «Обсуждение:».

Страница – это любой документ энциклопедии, который имеет заголовок. И статьи, и категории являются страницами.

Статьи – это страницы в пустом пространстве имён. Это основные страницы энциклопедии. Например, статья Шифр имеет заголовок [[Шифр]]

Категории – это страницы в пространстве имён «Категория:». Они служат для группирования сходных по тематике страниц.

Присвоить странице категорию можно, добавив странице тег категории со ссылкой на страницу категорию. Например, редактируемый текст статьи [[Рукопись Войнича]][2] содержит:

```
[[Категория:Шифры]]
[[Категория:Википедия:Избранные статьи]]
[[Категория:Древние книги]]
[[Категория:Нерасшифрованная письменность]].
```

Пользователь, в свою очередь, видит внизу страницы перечисление категорий (со ссылками на страницы категории):

Шифры, Википедия:Избранные статьи, Древние книги, Нерасшифрованная письменность

Страницы категории генерируются автоматически, они содержат ссылки на все страницы, содержащие упоминание о данной категории.

Сообщество Википедии рекомендует придерживаться следующих правил[3]:

- *Каждая страница должна быть внесена хотя бы в одну категорию. Можно внести и в несколько, однако иногда бывает разумнее внести страницу в категорию более высокого уровня.*
- *Каждая категория, кроме одной категории верхнего уровня, должна быть внесена хотя бы в одну категорию более высокого уровня.*

---

1 Двойные скобки в формате вики являются аналогом гиперссылки в Веб.

2 См. http://ru.wikipedia.org/wiki/Рукопись_Войнича

3 См. http://ru.wikipedia.org/wiki/Википедия:Категории

- *В одну категорию включают похожие статьи. В одну категорию включают похожие подкатегории.*
- *У категорий нет аннотаций.*

В работе [135] отмечают, что одним из основных источников информации в программных проектах, связанных с Википедией, являются категории. Но даже этот источник данных не является однозначно определённым, поскольку категории могут представлять не только родовидовые отношения (гиперонимия, is-a), но также отношения часть-целое (меронимия), наличие свойств (has-property) [173].

Категории не образуют дерево[1], скорее – это направленный граф без циклов [150] (хотя авторы Википедии прилагают усилия, чтобы таксономия категорий была бы деревом, по крайней мере, это отражено в рекомендациях к написанию статей Википедии[2]). Могут существовать многие схемы категоризации одновременно. Сообщество Википедии рекомендует избегать циклы, а в случае их обнаружения – избавляться от них.

Группировать статьи в энциклопедии можно с помощью категорий, списков, и навигационные шаблоны (article series box, navigational templates). Навигационные шаблоны указывают статьи в хронологической или иной последовательности. У каждого способа есть свои достоинства и недостатки[3]. Например, списки могут содержать ссылки на ещё не существующие страницы, а категории – нет[4]. Другой недостаток категорий в том, что удаление страницы из категории нельзя обнаружить, если только не было установлено слежение (watch) на все страницы категории[5]. Возможность автоматически связать (auto-linking) категорию со страницей (достаточно указать тег категории на странице) – главное достоинство категорий.

Главная разница между категориями вики и тегами социальных сетей[6] заключается, вероятно, в том, что пользователи создают теги сами, то есть теги отражают лексикон пользователя. А категории вики создаются

1 См. http://en.wikipedia.org/wiki/Wikipedia:Categorization

2 См. http://ru.wikipedia.org/wiki/Википедия:Правила_отнесения_в_категории

3 См. http://en.wikipedia.org/wiki/Wikipedia:Categories,_lists,_and_series_boxes

4 См. http://en.wikipedia.org/wiki/Wikipedia_talk:Categorization/Archive_1#Lists_v._categories

5 См. http://meta.wikimedia.org/wiki/Category_talk:Demo

6 См. замечание о folksonomy на стр. 183.

пользователями сообща. Тем не менее, теги – это ещё один источник информации, который может использоваться для кластеризации текстов. В работе [111] показано, что: (i) теги мало подходят для кластеризации текстов блогов[1]; (ii) часто встречающиеся теги (ЧВТ) подходят для описания кластеров как метаданные концепта кластера. Оценка пропорции ЧВТ в кластере позволяет оценить и удалить слабые[2] кластеры, проверить тематическую целостность набора текстов блогов.

Общий недостаток категорий и списков вики – это дублирование информации, то есть параллельное существование списков (например, List of astronomers) и категорий (Astronomers). К сожалению, дублирование информации не полное: не все статьи из списка могут быть помечены соответствующей категорией (например, не у всех страниц астрономов со страницы http://en.wikipedia.org/wiki/List_of_astronomers указана категория *Astronomers*).

На декабрь 2005 года 78% статей было соотнесено каким-либо категориям, всего зафиксировано 87 тыс. категорий [150], на январь 2006 года 94% статей соотнесено каким-либо категориям, всего зафиксировано 91.5 тыс. категорий [173].

1 См. Блогосфера российского интернета. Информационный бюллетень Яндекс. Осень 2006 года. http://company.yandex.ru/articles/yandex_on_blogosphere_autumn_2006.pdf

2 Под силой кластера понимается число тегов, часто употребляемых разными пользователями. Таким образом, авторы работы [111] выделяют сильные и слабые кластеры.