АЛГОРИТМЫ СЕМАНТИЧЕСКИХ ОТОБРАЖЕНИЙ НА ПОДМНОЖЕСТВО ЕСТЕСТВЕННОГО ЯЗЫКА, КАК ИНСТРУМЕНТ ИССЛЕДОВАНИЯ ПОИСКОВЫХ СИСТЕМ

В.В. Завертайлов

Появлению данного материала способствовал возросший за последние несколько лет спрос на крупные информационные ресурсы (например, сайты, мультимедийные приложения), в состав которых обязательно входит поисковая система. Существующие решения основаны на поиске среди словоформ, однако плохо (или вообще никак, например Mnogoserch, Google) обрабатывают такие моменты русского языка как чередование, выпадение гласных и согласных, сложные слова и прочие лингвистические тонкости. Наиболее удачные на сегодняшний день решения тоже далеки от идеала. Так, например, «великая тройка» - Яндекс, Рамблер и Апорт полностью игнорируют не только семантическую составляющую текста, но и синтаксис, не различают имена собственные даже в самых тривиальных ситуациях (для эксперимента можно набрать «козлов», «баранов»), хотя уже несколько лет существуют и апробированы алгоритмы, позволяющие в той или иной степени разрешать конфликтные ситуации (например, [3], [4]). Морфология – то, что более или менее удалось решить на сегодняшний день, тоже оставляет желать лучшего [2].

В данной работе осуществляется попытка построить некоторые алгоритмы, решающие проблемы поиска в текстах на естественных языках.

Рассмотрим поведение типичной поисковой системы.

Зачастую поисковые системы в своей работе используют механизм индексации текстов. Индексация означает предварительную подготовку текстов для поиска и применяется главным образом для его ускорения. Например, если поиск осуществляется в 10 мегабайтах документов, то можно подождать, пока поисковая машина «прочитает» их все, так как это не займет много времени. Однако, когда объемы текстовых статей велики, такой метод будет работать непозволительно долго. Для того, чтобы сделать возможным работу поисковых систем на больших объемах данных, текст обрабатывается заранее, а во внутренних структурах поисковых систем создаются индексы, которые представляют собой, как правило «координаты» слов тексте (чаще всего номер документа и номер слова в документе) и, возможно дополнительную информацию (используемую, как правило, для ранжирования результатов по релевантности). При поиске слово ищется в индексе, и по найденным координатам выдаются нужные документы. Если слов в запросе несколько, над их координатами производится операция пересечения. Несмотря на то, что принципы построения индексов во всех информационных поисковых системах похожи, от структуры индекса зависит большое количество качественных характеристик поисковой машины. Это обусловлено в первую очередь тем, что при индексировании происходит потеря информации. Например, если в качестве индексов поисковая система использует основу слов, то при поиске уже невозможно разобрать, какое слово из двух имелось в виду: «простаивающий» и «простейший». В структуре любой информационной поисковой системы (ИПС) всегда заложен компромисс между скоростью поиска и скоростью обновления индекса, связанный с допустимой степенью его фрагментации. Приоритеты выбираются в зависимости от области применения поисковой машины, и по этой причине многие ИПС имеют ограничения, заложенные в формат индекса.

Итак, попробуем разобраться, какая информация наиболее важна при создании индексов. Для этого проанализируем структуру среднестатистического текста, с которым мы встречаемся ежедневно, а так же посмотрим, чего именно обычно мы ждем от поисковых систем.

Разумно предположить, что поисковая система должна отдавать пользователю ссылки на документы, объем которых не велик, либо уметь локализовать интересующий пользователя блок текста. Это связано, во-первых, с тем, что в больших текстах содержится большее число семантических единиц, что может привести к ложному срабатыванию поисковой системы. С другой стороны большие тексты неудобны для пользователей.

Экспериментально был сделан вывод, что оптимальный объем статьи, которая будет восприниматься поисковой системой как единое целое, составляет от 2 до 8 тысяч символов (1-3 страницы).

Статья, в свою очередь может содержать параграфы. Можно выделить два основных подхода к построению статей. Условно назовем их «новостной» и «научный». Первый подход характеризуется тем, что в начале статьи кратко изложен весь смысл (самое интересное), а детали идут ниже по тексту. Второй подход – типовой для научных докладов, когда в начале статьи идет вводная часть, анализируются проблемы отрасли, затем ставится проблема, приводиться вариант решения, и, наконец, делаются выводы и перечисляются результаты. Таким образом, для поисковых систем наибольший интерес представляют первые и заключительные параграфы статьи, поскольку они наиболее емки в семантическом плане. Это же касается предложений в параграфах, значит индекс, учитывающий координаты слов в тексте должен хранить информацию о параграфе и предложении.

Следующая составляющая текстов, которая может учитываться в индексах – семантическая единица. Как правило, это слово или устойчивое выражение, или словосочетание. Представление семантических связей, безусловно, является самостоятельной и довольно сложной проблемой. В основе систем, учитывающих семантику, как правило, лежит база знаний, содержащая концептуальные, понятийные знания, изложенные в терминах предметной области. Как правило, база формируется в несколько этапов.

Первый этап - сбор информации и её накопление в базе данных.

Второй этап - структурирование знаний, извлечение из текста наиболее важных аспектов рассматриваемой проблемы. С этой целью производится сжатие (свёртка) текста посредством трёхстадийного процесса:

- определение основных доминирующих понятий;
- определение тем связанных частей текста;
- объединение их в структурированные объекты (фреймы).

Фреймы, в свою очередь, объединяются в кластеры, которые образуют более крупные блоки - графы. Граф текста служит средством представления содержания в виде семантической связи и позволяет разделить его на более или менее крупные смысловые структуры. В иерархическом дереве корневые узлы соответствуют наибольшей степени абстракции, а «листья» символизируют более частные понятия. При отборе понятий учитываются частотные признаки (коэффициенты активности), которые фиксируют каждое появление ассоциируемого понятия в анализируемом тексте. Чем выше коэффициент активности фрейма, слота или заполняющего его признака в базе знаний, тем большее значение имеет ассоциируемое понятие с точки зрения основной темы рассматриваемого текста. Таким образом, структуризация знаний представляет собой выделение смысловых компонентов текста и установления связей между ними.

Третий этап - накопление и систематизация знаний, объединение в классы и установление связей между ними. Основная проблема данного этапа - определение общих закономерностей поставленной задачи и разработка методик выявления пробелов и противоречий в базе знаний.

Структура индекса ИПС может быть представлена либо как указатель иерархических связей, обеспечивающих оперативный доступ ко всему объёму информации, либо в виде когнитивной карты, реализуемой методом многооконного представления информации. По существу, иерархический принцип - это группировка по вертикали, а альтернативный принцип - это горизонтальная группировка текстов. [5]

Безусловно, семантический уровень является важной составляющей при реализации ИПС, однако сложность подхода вынуждает искать компромисс между качеством и скоростью, упрощая или создавая альтернативные математические модели семантического уровня. Далее мы предложим алгоритм, основанный на понятии «словарный граф», и рассмотрим его применение в контексте задачи выявления узких мест в существующих поисковых системах. Для этого нам нужно дополнить индекс синтаксической и лексической составляющей.

Будем представлять индекс синтаксического уровня упрощенно. Для этого каждое предложение разобьем на семантический контекст, понимая под этим обособленную часть предложения, например, деепричастный оборот, или сложноподчиненное предложение. При этом мы не будем строить иерархию семантических контекстов (функциональную форму), а лишь сделаем для каждо-

го слова пометку, в каком контексте оно было встречено и в какой позиции.

Далее, нам понадобится лексический уровень, поскольку он наиболее полно реализован в популярных поисковых системах (Яндекс, Рамблер).

Существует два основных метода морфологического анализа: декларативный и процедурный.

При декларативном методе анализа имеется полный словарь всех возможных форм слова и соответствующая им морфологическая информация. Таким образом, морфологический анализ заключается в поиске нужного слова в словаре и извлечения соответствующей морфологической информации. Существенными недостатками этого метода являются большой объем словаря и невозможность его автоматического заполнения.

В процедурных методах анализа слова разбиваются на основу и окончания с суффиксами. В русском языке большинство слов можно отнести к определенному флективному типу. Флективный тип это множество слов с одинаковыми правилами словообразования. Таким образом, в словаре хранится только основа и ссылка на соответствующий флективный тип, по которому определяются морфологические характеристики этого слова. Этот метод позволяет значительно сократить объем словаря, и дает возможность автоматически заполнить его.

Однако такой подход не учитывает многих особенностей русского языка.

Во-первых, чередования. Автором работы был предложен комплекс мер для выявления чередований. Используя словарь Зализняка [6] построены картежи вида [основа, кластер основ], где кластер основ – совокупность видоизмененных основ для какой-либо парадигмы. Например, слова “идти” и “шел” попали в один кластер. Кроме того, как правило, слова с чередованием типа гар-гор/зарзор/лаг-лож так же попали в один кластер. Таким образом, частично была решена проблема чередований и выпадений гласных и согласных.

Дополнительно к этому предложен набор правил (таблица 1), позволяющих выявить чередования. В большинстве случаев наличие чередования и участвующая в нём гласная определяется двумя последними согласными основы. Пусть последний согласный основы X, а предпоследний – Y.

Таблица 1 – Набор правил чередований для гласных

1	Если X=Y (удвоенная согласная на конце), то чередования нет: *тонн*
2	Если X – ***ть***, то если основа – это основа существительного 1-го склонения, то есть чередование с гласной ***о***. Иначе чередования нет.
3	Если X – ***ц***, то беглая гласная – ***е***, а Y смягчается: *ов**е**ц, двер**е**ц, кол**е**ц, полотен**е**ц.* Для слова *за**я**ц* имеется орфографическая особенность – буква ***я*** вместо ожидаемой ***е***. Чередования нет, если Y – ***щ*** или Y предшествует согласная: *беглец, близнец, жрец*.
4	Если X – ***й***, то под ударением пишется ***е***: *сем**е́**й, стат**е́**й*, а без ударения – ***и***: *го́ст**и**й, певу́н**и**й*. Исключения: *су́д**е**й* (также *суд**е́**й*), *ру́ж**е**й*, *у́л**е**й, чи́р**е**й*.
5	Если X – ***нь***, то гласная – ***е***. При этом Y смягчается, а X в формах род.п. мн.ч. сущ. 1 и 2 склонений и отвердевает для м. р. ед.ч. краткой формы прилагательных и причастий: *песен, вишен, корень, камень, древен* (начальная форма – *древний*), *господень*, *песенный.* Исключения отвердевания X: *барышень, боярышень, деревень*. Исключения чередования – редкие, новые слова, имена собственные: *ревень – из ревеня, Сковпень – Сковпеня*.
6	Если Y – ***ль***, то гласной нет: *пальм, фольг, ольх*. Исключение: *валет* (беглость вариативна).
7	Если Y – мягкий (кроме ***ль***, включая ***й***), шипящий или ***ц***, то гласная – *æ*: 1.под ударением пишется ***ё*** (после шипящих – ***о***): *серёг, княжон, кишок* *2.* без ударения – ***е***: *судеб, кошек, книжек* В форме Мужской род ед.ч. краткой формы прилагательных и причастий (*достои**н**)* имеется орфографическая особенность – буква ***и*** вместо ожидаемой ***е***.
8	Если X – ***ль***, то гласная ***е***: *ка́п**е**ль, кораб**е**льный*, кроме форм Им.п. ед.ч. существительных 2 склонения, в которых чередования нет: *корабль, журавль*, а формы с ***е*** считаются просторечными или шуточными: *рубель, журавель*. Исключения: *стебель, комель, кегель* (также *кегль*, И. мн. *кегли*).
9	В корнях слов *яйцо, один* гласная – ***и***: *яиц*, *один*, *яичный, единый*.

Если морфологическому анализатору не удалось распознать основу, поисковая машина пытается применить вышеописанные правила ко всем вариантам псевдооснов. Для этого производится поиск «подозрительных» букв, поиск подходящего правила и выполняется замена, после чего полученная основа проверяется на наличие в словаре основ и делается вывод о правильности разбора. Для проверки правил используются списки флективных типов (для выявления морфологических данных), словарь ударений, а также вспомогательный словарь для анализа выпадений.

Анализ чередования согласных выполняется на стыке корня и суффикса. Учитываются следующие случаи: *ч/к (пе**ч**ёшь – пе**к**у), ж/г (бере**ж**ёшь – бере**г**у*), *д/ж (хо**д**ишь – хо**ж**у), с/ш (но**с**ишь – но**ш**у), з/ж (во**з**ишь – во**ж**у), т/ч (ле**т**ишь – ле**ч**у), б/бл (лю**б**ишь – лю**бл**ю), в/вл (лю**в**ишь – лю**бл**ю), ф/фл (гра**ф**ишь – гра**фл**ю), м/мл (кор**м**ишь – кор**мл**ю), п/пл.*

В настоящий момент до конца не ясно, в какой мере чередования и выпадения должны влиять на релевантность.

Еще одним камнем преткновения для поисковых машин являются составные слова. Предлагаемый метод довольно прост и учитывает 4 случая сложных слов:

1. Стыковочная гласная “о” или “е”. Системы выдвигает гипотезу о том, что слово состоит из двух основ и выполняет проверку (рекурсивный вызов) на корректность обеих основ. Учитывается, что от первой основы не следует отделять постфиксы (хотя, существует очень низкая вероятность встретить суффикс), а от второй - префиксы. При этом, в случае успешного разбора обоих частей слова, морфологическая информация берется из последней. Кроме того, система смотрит, не относится ли стыковочная к какому либо из слов (т.е. проверяется три гипотезы).
2. Числительные – предполагается реализация специального алгоритма для интерпретации составных числительных.
3. Используются приставки, имеющие собственное семантическое значение: «супер», «ультра», «экстра». В системе хранится словарь таких приставок.
4. Дефисное описание. Такие слова разбиваются на два еще на этапе выделения слов из текста.

Таким образом, можно предложить структуру индекса, переставленную в таблице 2, который бы учитывал структуру текста наиболее полно.

Таблица 2 – Структура индекса

Морфологическая составляющая	**Контекстная составляющая**
Основа	Документ (статья)
Окончание	Абзац
Множество суффиксов	Предложение
	Семантический контекст
	Слово
	Позиция в слове (для составных слов)

Рассмотрим одно из применений такого рода индексов. Введем понятие словарного графа. В вершинах графа находятся словоформы и основы слов. Дуги взвешены. Каждая дуга может принадлежать к одному из 5 типов.

1. Основа. Эта связь существует между словоформой и ее предполагаемой основой. Вес дуги зависит обратно пропорционально от количества отсеченных при разборе морфем и количества вариантов разбора.
2. Морфологическая форма. Эта связь выставляется между словами с одинаковой основой. Вес дуги прямо пропорционально зависит от коэффициента достоверности принадлежности слов к одному и тому же флективному типу и обратно пропорционально от количества флективных типов, к которому отнесена основа.
3. Семантическая форма. Эта связь устанавливается между словоформами, которые отличаются приставками. В результате ряда экспериментов было выяснено, что приставка очень сильно влияет на семантику слова (например “пришел” и “ушел”). Таким образом, во многих системах основа хранится в словаре вместе со своей приставкой. Однако такая связь полезна при генерации текстов. В настоящий момент хороший алгоритм построения таких связей автору не известен.
4. Синоним. Данная связь устанавливается между словами, имеющими схожий смысл.
5. Дистрибуция. Эта связь показывает, могут ли данные слова сочетаться в текстах, следуя друг за другом. Направление дуги – от первого слова ко второму. Вес дуги показывает, насколько такое сочетание достоверно. Рассчитывается, как количество словосочетаний, встреченных в достаточно большом объеме текста, в которых первое слово соот-

ветствует начальной вершине графа, а второе – конечной.

Заметим, что это не далеко не исчерпывающий список типов дуг, и в зависимости от типа задачи его можно расширить (например, построить дуги по словарю антонимов, омонимы и т. д.). Как видно из графа определения, он учитывает, в основном, лексический, семантический, и косвенно синтаксический уровень естественного языка (дуга, типа 5). Более того, дуги 1-4 типа актуальны для естественного языка постоянно (то есть, если их коэффициенты были получены в результате автоматического обучения, значения коэффициентов в идеале не должны зависеть от выборки). Назовем граф, в котором построены дуги первых четырех типов - чистым. Дуги пятого типа необходимы, чтобы учесть структуру текста. Можно считать, что веса таких дуг есть функция от пары слов и индексов, структура которых была показана в таблице 2.

Дуги первого и второго типа для естественных флективных языков можно построить, используя алгоритм Белоногова. На первом этапе требуется выявить из больного объема текстов всевозможные словоформы. При этом необходимо набрать статистику, сколько раз встречалась каждая словоформа с тем или иным суффиксом и окончанием. Для этого были вручную получены списки морфем русского языка.

Каждое выделенное слово поступает на дальнейшую обработку. Нужно всевозможными способами отделить от слова суффиксы и окончания. При этом разумно выделять не более трех суффиксов. Морфема-кандидат (т. е. отделенная часть слова, про которую еще не известно, является ли она морфемой русского языка) проверяется на соответствие в базах суффиксов или окончаний. Полученная словоформа заносится во временный словарь.

На следующем этапе работы системы ставится задача выделения из общего списка словоформ – основ. Алгоритм основан на том, что существует сильная корреляционная связь между грамматическими характеристиками слов и буквенным составом их окончаний и суффиксов. Например, слова "ложка" и "поварешка" – существительные женского рода, единственного числа. Можно заметить, что любая попытка получить новую словоформу первого слова, используя допустимые суффиксы, приставки и окончания, может быть без проблем повторена и со вторым, а морфологические признаки образованных слов будут совпадать.

Таким образом, можно ввести понятие флективного типа - множества слов, к которым могут быть применены одинаковые словообразовательные функции, чтобы полученные в результате слова имели сходную морфологическую информацию.

В результате анализа структуры русского языка, для имен существительных было получено более 60 флективных типов. Каждый из них задавался списком концевых морфем, допустимых для данного флективного типа при склонении слова.

Когда в систему поступает очередная основа, требуется определить, с какой степенью достоверности она относится к каждому флективному типу и если эта вероятность превышает 90 %, то основа считается правильной, а между выделенной основой и полученными из текста, соответствующими ей словоформами, ставится связь первого типа. Между всеми словами, отнесенными к одной основе, ставятся связи второго типа. При построении таких отношений следует учитывать методы анализа чередований и выпадений, которые были предложены выше.

Связи второго типа можно усилить, используя словарь Зализняка. Словарь представляет собой набор картежей слов. Слова, которые отнесены к одному картежу, являются словоформами друг друга. Таким образом, связь между словами из одного картежа должна получить больший вес.

Действуя аналогично для приставок, можно попытаться построить дуги 3-го типа, однако построенные автоматически связи не очень достоверны, поскольку нет очень четкой зависимости между приставками и флективным типом. Попытка сопоставить приставки флективным типам методом кластеризации позволяет выявить лишь часть приставок, которые крайне редко встречаются с каким-либо флективным типом.

Связь 4-го типа строится по словарю синонимов непосредственно [6].

Имея эти данные, покажем, как обучить граф (подобрать коэффициенты дуг 5-го типа), а так же как на основании обученного графа и индексов строить высоко релевантные тексты для конкретных поисковых запросов, которые могут быть прочитаны человеком. Такая задача часто встречается в Internet, когда необходимо поднять рейтинг статьи в поисковых системах, однако на сегодняшний день автору не известны способы, позволяющие автоматизировать этот процесс.

Путь C = <W1 W2 … Wn> - целевой поисковый запрос, где Wi – конкретные слова.

Задавая этот запрос интересующим нас поисковым системам, мы получим набор ранжированных статей (обучающую выборку), M = <P1, P2, … Pk>, где Pj – набор статей от конкретной поисковой системы: Pj = <S1, S2…Sm>, где Sk – конкретная статья.

Обработаем статьи последовательно и построим индексы:

1. Последовательно просматриваем текст, отслеживая контекстную составляющую.
2. Для каждого слова находим множество предполагаемых основ, используя словарный граф и связи первого типа.
3. По найденной основе, информации об отсеченных от словоформы морфемах и контекстной составляющей строим и сохраняем индекс.

Предложенный далее ряд формул позволяет рассчитать коэффициент релевантности статьи по поисковому запросу.

$$r_i = \sum_{j=0}^{n} \begin{pmatrix} S1 - \min\limits_{k=0}^{m_j}(K_{flect}(i,j,k)) \\ - \min\limits_{k=0}^{m_j}(K_{ml}(j,k)) \end{pmatrix} + K_{context}$$

$$S1 = \sum_{k=0}^{m_j} \begin{pmatrix} K_g(j,k) + \\ (K_{col}(i,j,k) * K_{meta}(i,k)) \\ - K_{flect}(i,j,k) - K_{len}(i,j,k) \\ + K_{pos}(i,k) \end{pmatrix}$$

$$K_g(j,k) = (1 - sign(k)) \times (C_{group} - j * C_{fin})$$

$$sign(k) = \begin{cases} 0, k = 0 \\ 1, k > 0 \end{cases}$$

$$K_{col}(i,j,k) = isFlect(S_j, O_{j,k}) \times Col_{i,k} \times C_{col}$$

$$Kmeta = C_{meta}(1 + COL_{i,k,t})$$

$$K_{flect}(i,j,k) = colFlect(O_{j,k}) \times C_{flect} \times Col_{i,k}$$

$$K_{len}(i,j,k) = \left\| S_j \right\| - \left\| O_{j,k} \right\| \times Col_{i,k} \times C_{lenCol}$$

$$K_{ml}(j,k) = \left\| S_j \right\| - \left\| O_{j,k} \right\| \times C_{len}$$

$$K_{context} = C_{context} * \frac{\sum_{t=2}^{n} \begin{pmatrix} A_{\min}(t) * C_{abzac} + \\ P_{\min}(t) * C_{pred} \\ + W_{\min}(t) * C_{word} \end{pmatrix}}{(n - t + 1)}$$

$$K_{pos}(i,k) = \frac{C_{wpos} * \sum_{t=0}^{o_k}(APos(k,t))}{(Col(k,i) + 1)}$$

i – индекс текущей статьи;

r_i – коэффициент релевантности для i-й статьи;

j – индекс текущей словоформы из поискового запроса;

k – индекс очередной основы, относящейся к k-й словоформе;

K_g – бонус за каждое новое слово из запроса, найденное в текущей статье;

K_{col} – бонус за количество встреченных k-х основ в i-й статье;

K_{flect} – штраф за количество флективных типов у k-й основы;

K_{len} – штраф за краткость основы по сравнению с исходной словоформой, учитывающий количество k-х основ в i-й статьи;

K_{ml} – штраф за краткость основы по сравнению с исходной словоформой.

Kcontext – бонус за близость слов из запроса в тексте статьи

Kmeta – бонус за то, что к ключевых словах содержатся основы из запроса, найденные в статье.

Kpos – бонус за положение основы внутри статьи;

S_j – словоформа из исходного запроса;

$O_{j,k}$ – сопоставленная S_j словоформе основа;

isFlect(S_j, O_j, k) – вернет 1, если словоформа S_j сопоставима с каким-либо флективным типом, соответствующим основе $O_{j,k}$;

$Col_{i,k}$ – количество k-х основ, встреченных в i-й статье; Coli,k,t – количество k-х основ, встреченных в i-й статье, относящихся к t-му ключевому слову;

colFlect(O_j, k) – количество флективных типов, соответствующих основе O_j, k;

$Col_{i,k}$ – количество k-х основ, встреченных в i-й статье;

S_j – j-я словоформа из запроса;

$O_{j,k}$ – соответствующая j-й словоформе основа;

$Col_{i,k}$ – количество k-х основ, встреченных в i-й статье;

Норма от словоформы определяется как количество символов в словоформе;

n – количество слов в поисковом запросе;

Amin(t) – Минимальное число абзацев, которое необходимо прочитать, чтобы встретить t слов из запроса;

Pmin(t) - Минимальное число предложений, которое необходимо прочитать, чтобы встретить t слов из запроса;

Wmin(t) - Минимальное число слов, которое необходимо прочитать, чтобы встретить t слов из запроса;

Ok – Абзацы, в которых встречается k-я словоформа;

APos(i, k, t) – смещение k-й словоформы в t абзаце;

Col(k, i) – Количество к-х словоформ в i-й статье;

C_{lenCol}, C_{len}, C_{flect}, C_{meta}, C_{col}, C_{group}, C_{fin} – некоторые коэффициенты.

Релевантность каждой статьи обратно пропорциональна позиции статьи выданном ИПС списке статей. Пусть первая статья в списке имеет коэффициент релевантности 1, вторая – 1/2, третья – 1/3 и так далее. Таким образом, зная этот коэффициент, выборку статей и поисковый запрос, можно рассматривать ri как функцию от C_{lenCol}, C_{len}, C_{flect}, C_{meta}, C_{col}, C_{group}, C_{fin}. Решив задачу оптимизации каким либо алгоритмом, найдем эти коэффициенты.

Теперь решим задачу построения высоко релевантных, читабельных для человека, текстов.

Задача построения читабельных текстов решается с использованием пятого типа связей в словарном графе. Покажем идею на примере. Пусть нам даны пара предложений:

“Стая птиц высоко летит над землей” и “Стадо коров высоко забрались в горы”. В словарном графе, помимо прочих, будут построены следующие связи:

птиц -> высоко
коров -> высоко
высоко -> летит
высоко -> забрались.

Если обходить граф от слова “стадо” и в вершине “высоко” пойти по дуге “высоко” -> “летит”, то обход даст следующую фразу: “Стадо коров высоко летит над землей”. Учитывая то, что существует связь четвертого типа (синоним) стадо-> табун, а так же знание флективных признаков обоих слов, то можно получить следующее: “Табун коров высоко летит над землей”. Это предложение человек может прочитать без проблем, но его семантическая ценность – весьма сомнительна. Проходя по дугам пятого типа можно без проблем генерировать читаемый текст. Покажем, как сделать это текст высоко релевантным.

Представим текст в виде последовательности ключевых и не ключевых слов:

<s1><s2>…<k1><sn>…<ki>…<sm>,

где s1 – некоторые слова, ki – слова из ключевого запроса. Поскольку si не участвуют в формуле расчета релевантности, они могут быть любыми. Используя формулу расчета релевантности и индексы, отберем из всей массы статей предложения, содержащие ключевые слова и отсортируем их по релевантности. Путь получено P=<p1, p2…pn> предложений. Оставив ключевые слова в этих предложениях на своих позициях и используя дугу пятого типа, заменим остальные слова по описанному выше принципу. При этом, чтобы снизить эффект влияния других слов на поиск, будем идти по дугам со слабой связью. В процессе построения таких предложений возникает задача: пусть в предложении есть несколько ключевых слов, например 2. Как, используя словарный граф, построить цепочку между ними нужной длины? По построению предложения, такая цепочка обязательно существует. Учитывая то, что близость слов из запроса в тексте положительно влияет на релевантность, будем строить цепочки кратчайшей длины, используя алгоритм Дейкстры. Для полученных таким образом предложений повторно рассчитываем релевантность и сортируем предложения по убыванию.

Будем последовательно менять предложения в исходных абзацах на те, которые мы получили на предыдущем этапе. Таким мы увеличиваем релевантность абзацев, не меняя их структуры. Предложения, которые не содержат ключевые слова, сгенерируем по словарному графу, подставляя первое слово в предложении как исходную точку маршрута в графе. Определим релевантность каждого полученного абзаца и отсортируем их в порядке убывания.

Взяв за основу самую релевантную статью из исходных, распределим абзацы. В начале работы алгоритма все абзацы считаем незафиксированными. Далее, будем брать самый релевантный незафиксированный абзац в исходной статье и менять его на самый релевантный абзац из тех, которые были подготовлены на предыдущем этапе. После этого замененный абзац фиксируется и удаляется из списка подготовленных абзацев. Как вариант, на этом шаге для распределения абзацев можно брать концентрацию ключевых слов.

По построению, полученный текст будет обладать релевантностью, большей, или равной релевантности самой хорошей статьи.

К недостаткам подхода можно отнести то, что он не учитывает ранг страницы, рассчитанный на неязыковых отношениях между документами (например, коэффициент Page Rang). Учет этого коэффициента можно внести в предложенную формулу расчета релевантности, однако как это повлияет на общее качество алгоритма пока неизвестно.

Данный подход применим для всех существующих ИПС, а его практическое применение позволяет решить задачу, которая часто встречается среди специалистов по поисковой оптимизации.

СПИСОК ЛИТЕРАТУРЫ

1. http://scon155.phys.msu.ru/~swan/ Орфографический словарь русского языка для ispell.
2. http://www.internet.ru/index.php?itemid=484 Битва титанов: полемика между РБК и Яндексом.
3. http://www.delphikingdom.com/asp/viewitem.asp?catalogid=412 склонение фамилий, имен и отчеств по падежам Библиотека функций.
4. http://www.rco.ru/article.asp?ob_no=30 Russian Context Optimizer: путь к возможностям Oracle interMedia в русскоязычных базах данных.
5. http://www.ifap.ru/library/index.htm Материалы конференции «EVA 2003 Москва».
6. http://09.org.ru/downloads/?id=2 словарь синонимов русского языка.