Опыт участия ИС «Кодекс» в РОМИП 2003

© Губин Максим

ИК «Кодекс»

max@gubin.spb.ru

Аннотация

Статья описывает опыт участия компании «Кодекс» в РОМИП 2003. Представляемая нами система «Кодекс» предназначена для создания информационно-справочных систем и систем документооборота. В статье кратко описывается архитектура системы и методы, которые использовались и используются при внутренней оценке качества информационного поиска. Рассказывается, как выполнялись задания семинара и трудности, с которыми столкнулись разработчики системы.

1 . Цели участия и организация семинара.

1.1. Цели участия в семинаре

В настоящее время на российском рынке можно выделить несколько видов информационных систем, которые используют технологии информационного поиска и автоматической классификации:

1. Системы поиска в Интернет/Интранет. Это такие общедоступные сервисы, как Яндекс, Рамблер, Апорт, Пунто и т.п., а также ряд предлагаемых этими и другими разработчиками систем для реализации заказных решений.
2. Информационные системы, представляющие пользователю документарные базы данных. Наиболее известными среди них являются правовые системы, такие как Гарант, Кодекс, Консультант+, Референт, а так же информационные системы технической документации (Кодекс-Стройэксперт, СтройКонсультант).

3. Системы документооборота. Такие системы, как правило, в качестве компонента содержат систему хранения документов с возможностью поиска и классификации. К ним относятся Documentum, Excalibur, Евфрат, Кодекс-Документооборот и т.д.
4. Специализированные информационные системы – архивы различных периодических изданий, библиотечные системы.

Наша организация много лет занимается разработкой и внедрением систем, которые можно отнести к п. 2, 3, 4, и мы давно интересуемся исследованиями в области информационного поиска. Отсутствие сложившихся в России в этой области общепринятых стандартов приводило к тому, что все разработки опирались только на собственный опыт или на информацию из зарубежных источников.

Когда возникла идея организации семинара, посвященного данному вопросу, мы были готовы его поддержать и принять в нем участие. Нам казалось, что участие в этом мероприятии позволит достигнуть следующих целей:

1. Познакомиться с другими группами, ведущими разработки в данной области. Это позволило бы обмениваться идеями и ускорить общее движение вперед.
2. Использовать наши разработки в необычных для нас задачах, таких как Web-поиск, которым мы никогда не занимались, а так же применить новые для нас методы оценки.
3. Исследовать на нестандартном наборе данных разрабатываемые нами перспективные алгоритмы.

Хотя с самого начала было понятно, что участие в инициативе потребует определенных ресурсов, мы сильно недооценили, требуемые усилия и необходимое время. Особенно это проявилось на заключительных этапах и не позволило полностью реализовать намеченный план. Более подробно о достигнутых результатах - в конце данной статьи.

1.2. Организация семинара

Хочется выразить благодарность всем организаторам данной инициативы. Безусловно, положительными моментами являются следующие:

1. Ориентация на решения, выработанные зарубежными семинарами. Это позволило лучше понять задания, что именно надо сделать и не изобретать «велосипед».
2. Использование в программном обеспечении и для представления данных открытых стандартов. Такой формат, как

XML значительно упростил адаптацию и доработку программного обеспечения.

Все недостатки проведения семинара, на самом деле, являются следствием новизны и сложности организации. Так, например, результаты были представлены достаточно поздно для того, чтобы их можно было успеть детально проанализировать к моменту написания данной статьи.

2 . Описание системы

2.1. Общее назначение системы

ИС «Кодекс» разрабатывается более 10 лет. Первоначально она разрабатывалась только как информационно-правовая система. В дальнейшем, при возникновении новых задач, система стала активно развиваться как средство для разработки систем хранения документов различного назначения. В настоящее время на базе ядра системы разработано и внедрено несколько сотен информационных систем, которые хранят от десятков до миллионов документов в одной базе данных.

2.2. Архитектура системы

Основой системы является база данных собственной разработки для хранения текстов. База данных находится в одном или нескольких файлах, при этом данные хранятся в сжатом виде с использованием собственного архиватора или библиотеки ZLIB[1].

Система поддерживает хранение документов в следующих форматах: внутренний формат «Кодекс», RTF, PDF и XML.

Для ускорения поиска документы индексируются. Слова хранятся в индексе во всех словоформах, морфологический анализ производится уже на этапе поиска. Индекс имеет структуру B-дерева, информация в нем сжимается с помощью двухпроходного алгоритма:

1. Пост-листы сжимаются с помощью кодирования дельт с переменным количеством байт [2].
2. Страницы дерева сжимаются LZSS алгоритмом.

Применение такой последовательности дает возможность достичь отношения индекс/данные равного 0.1-0.3. Столь большое внимание к компактности индекса, даже в ущерб производительности, объясняется тем, что база данных вместе с индексами поставляется пользователям и каждый сэкономленный мегабайт важен. Подобная архитектура текстового индекса (правда, с более примитивной схемой сжатия) описана, в статье [3]. Интересно, что хотя наша разра-

ботка производилась раньше, чем описанная в статье, и совершенно независимо от нее, результаты и технические решения получились очень похожими.

Над уровнем базы данных находится оболочка, исполняющая сценарии на языке JavaScript (ECMA-262). Это позволяет достаточно легко адаптировать систему к различным требованиям и разрабатывать нестандартные интерфейсы. Практически все программное обеспечение, которое обеспечивало обработку данных РОМИП, было написано на этом уровне.

2.3. Подсистема информационного поиска.

Подсистема информационного поиска является составной частью подсистемы базы данных. При этом имеется механизм, который позволяет настраивать алгоритмы поиска под конкретные задачи из сценариев. При информационном поиске используется следующая информация о документе:

1. Термины, содержащиеся в документе. Для каждого термина указывается его нормализованный вес и признаки присутствия данного термина в заголовках.
2. Атрибуты документа. Это специфические для приложения характеристики документа, например, автор, принявший орган, номер, дата принятия и т.п. По нашим данным более 30% запросов пользователей нашей системы содержат значения атрибутов. Специфическая для наших приложений обработка атрибутов документов не использовалась в задачах семинара.
3. Гипертекстовые связи между документами.

Информационный поиск производится в несколько этапов. На первом этапе производится разбор запросов и формирование терминов-кандидатов на поиск. При этом производится морфологический анализ и формирование словоформ, присваиваются веса. На втором этапе производится отбор документов, содержащих термины запроса и их взвешивание. Выбираются документы, содержащие все слова запроса. Если таких документов нет, или их менее определенного числа (которое может настраиваться), применяются специальный алгоритм отбрасывания терминов. При взвешивании используется схема подобная классической TF*ITF [4], в несколько измененном виде. На следующем этапе учитываются связи между документами, путем прибавления части веса документа документам, на которые он ссылается. Атрибуты рассматриваются как специальный вид терминов, с достаточно сложной схемой их выделения из запроса и взвешивания.

2.4. Внутренние методы оценки информационного поиска

Любая разработка требует периодической проверки достигнутого результата. У нас имелись коллекции документов, запросы, которые были получены из протоколов работы пользователей, и эксперты - юристы. В первую очередь, разрабатываемые методики должны были оценить качество поиска с точки зрения пользователя. Кроме того, первоначально юристы относились к данной функции достаточно скептически и не очень хорошо представляли, что следует от нее ожидать. Первый вариант методики внутренней оценки начал использоваться в 95 году. Применялась следующая методика оценки качества:

1. Эксперты из массива пользовательских запросов отбирали некоторое количество.
2. Данные запросы обрабатывались системой.
3. Результаты обработки каждого запроса оценивались экспертом по пятибалльной шкале. Буквально, эксперту ставилось следующее задание «Представьте, что вместо системы результат поиска формирует ваш коллега-практикант. Оцените его работу».
4. Пункты 2 и 3 повторялись для каждой версии алгоритма, и для каждой версии вычислялось среднее арифметическое всех данных оценок.

Данный метод достаточно хорошо себя зарекомендовал. Судя по отзывам пользователей, качество работы системы заметно возросло. Особенно заметное улучшение качества дали учет в функции взвешивания наличия термина в заголовках документов и учет гипертекстовых связей между документами. Однако недостатком была большая трудоемкость оценки. Кроме того, исключительно по психологическим причинам, эксперты при повторяющихся опытах начинали снижать качество обработки, систематически занижая или увеличивая оценку в зависимости от настроения. Данную систематическую ошибку мы пытались компенсировать «замешиванием» повторяющихся заданий, то есть пользователю вместе с новыми представлялись и уже оцененные результаты. Такой прием часто используется в usability экспериментах. Однако это опять же увеличивало трудоемкость оценки работы системы. Необходимость автоматизации процесса оценки была очевидна. Используемая сейчас у нас методика внутренней оценки качества системы описана в статье [5].

3. Участие в РОМИП

До участия в семинаре мы никогда не индексировали Web-коллекции. Хотя в ряде наших проектов индексировались HTML-данные, но при этом использовался примитивный разборщик файлов, разработанный с применением встроенного интерпретируемого языка сценариев системы, и потому очень медленный. При этом не анализировались кодировки, и не выделялось форматирование и гиперссылки. Первые же тесты показали, что при таком подходе время обработки тестового набора данных неприемлемо большое. Было принято решение разработать специальный разборщик HTML-документов. Предполагалось, что в Интернете можно найти подходящие исходные коды, которые при минимальной адаптации могли бы быть использованы. Было проанализировано порядка десяти подобных модулей, но все они оказались не подходящими для использования в нашей системе: либо это были универсальные библиотеки, которые предъявляли достаточно высокие требования к отсутствию ошибок в HTML, либо они требовали наличие документа полностью в памяти, либо же интерфейс не подходил для встраивания в систему. В конце концов, был разработан свой модуль на языке C. Он представляет собой HTML-парсер с интерфейсом подобным SAX [6]. При загрузке тестовой коллекции оказалось, что документы содержат очень большое количество стилевых таблиц и сценарных вставок. Данные элементы просто игнорировались. Разборка гипертекстовых ссылок принесла новые сюрпризы: относительная адресация значительно усложнила алгоритм разбора, кроме того, она не могла быть решена в рамках парсера текста – для получения идентификатора документа надо было выполнить обращение к базе данных. Перед тем, как изменять архитектуру системы, вводя в парсер текста callback, мы решили проверить, будет ли это действительно важно для данной коллекции. Было проверено относительно небольшое (20) количество страниц, случайно выбранных из коллекции, и мы пришли к выводу, что все связи либо указывают на документы за пределами коллекции, либо являются чисто навигационными связями внутри одного сайта. Поэтому было принято решение не усложнять задачу и в первый раз выполнить задание с упрощенным алгоритмом, не учитывающим связи между документами.

Загрузка коллекции осуществлялась на компьютере с процессором PIII-650 и 192 Мб памяти, винчестер 30Гб под управлением Windows XP. Чистое время загрузки не известно, т.к. в процессе загрузки система несколько раз останавливалась для проведения проверок как идет процесс, и по техническим причинам (вышел из

строя вентилятор охлаждения). Чистое рабочее время загрузки коллекции составило 3 суток. Общий размер базы данных, которая содержала сжатые исходные страницы, индексы и сценарии обработки, составил 2.39 Гб.

Обработка задачи ad hoc проводилась на той же самой машине с помощью специально написанных сценариев. Определенную трудность составляло то, что исходные данные не содержали никаких тестовых примеров, которые позволили бы проверить наличие грубых ошибок реализации.

При первом прогоне выполнения задания оказалось, что после обработки некоторого числа запросов быстродействие системы резко падало, процессор загружался на 100%. Анализ показал, что причина крылась в алгоритме использования страниц памяти. Система обычно эксплуатировалась в режиме, когда запросы приходили в смеси – запросы на поиск, запросы на выборку документа. При этом страницы индекса снимались из памяти, как только не хватало места для страниц с текстом документов, и обычно их было не очень много. При тестовом прогоне в памяти оказалось слишком много страниц индекса и реализованный алгоритм вытеснения, хорошо работавший на небольшом количестве страниц, оказался неэффективным. Функция определения приоритета снятия страницы, которая учитывала положение страницы в дереве и частоту ее использования, требовала слишком много процессорного времени. Пришлось переделать механизм управления памятью, адаптировав его для нового вида нагрузки. В результате, прогон всех 15511 запросов занял около суток. Система сразу формировала файл результатов в нужном формате. Кроме автоматизированного прогона, около 10 запросов было выполнено вручную, чтобы проверить правильность функционирования системы. Данная проверка показала, что минимальные требования к правильности функционирования выполняются – система выбирает документы, содержащие слова запроса.

Первоначально планировалось выполнить два прогона, на втором применив новый алгоритм поиска, учитывающий взаимное положение слов в документах. Однако, оценив время, которое потребуется на адаптацию новой версии системы и выполнения данного эксперимента, мы пришли к выводу, что к заявленным срокам не успеваем и решили провести их позже, когда будут известны результаты, выполнив поиск только для тех запросов, которые оценены асессорами.

4. Результаты РОМИП

Мы ожидали результаты с определенным волнением. Во-первых, как уже говорилось, коллекция была во многом для нас не привычной. Во-вторых, как описывалось в предыдущем разделе, в алгоритмах поиска и индексации было введено большое количество упрощений. Поэтому, ожидалось, что качество поиска системы должно было заметно ухудшиться. По нашим оценкам отключение учета гиперссылок и флага наличия слова в заголовках приводило к ухудшению качества поиска (в смысле уменьшения параметров точность и полнота) на 30-50% для стандартных для нашей системы данных.

Методики оценки, использованные в РОМИП, очень академичны и имеют достаточно проработанный и сложный математический аппарат. Как эти данные соотносятся с более привычными для нас инженерными полуэмпирическим методами, еще предстоит выяснить. К сожалению, из-за ограниченных ресурсов мы выполнили только один прогон, нам не удалось выполнить и провести анализ качества наших новых алгоритмов. Поэтому при анализе результатов мы могли только оценить, насколько хорошо наша реализация информационного поиска в данных условиях выглядит на фоне других участников.

Если анализировать классические параметры точность и полнота, и по ним попытаться сравнить системы, считая количество участников, показавших лучшие и худшие значения, то наша система окажется крепким «середнячком». Это говорит о том, что, по крайней мере, грубых ошибок реализации, несмотря на спешку, сделано не было. Странно то, что исходя из обычного соотношения точность/полнота, должно было получаться, что системы, показавшие лучшую полноту, должны были проиграть в точности и наоборот. Однако результаты показывают, что подобная зависимость проявляется только для одного случая, во всех остальных мы либо выигрываем по обоим показателям или проигрываем.

При проведении семинара все оценки рассчитывались для двух вариантов оценки релевантности документа. При варианте «strong» документ считался релевантным, если все эксперты его так оценили. При варианте «weak», документ оценивался релевантным, если оценивался так хотя бы одним экспертом. Худшие значения полноты и точности для оценки релевантности асессорами «strong» достаточно очевидны, т.к. в этом случае количество документов, отобранных как релевантных, меньше. Несколько расстраивает то, что разница по точности столь большая – практически в 3 раза. Это говорит о том, что большая часть отобранных по запросу документов оказа-

лись «спорными» с точки зрения человека и при реальной эксплуатации результаты могут показаться сильно «зашумленных».

Если сравнивать полученные значения с результатами, которые получаются для систем, участвующих в конференции TREC показывают, что у нас есть еще к чему стремиться. Хотя прямое сравнение из-за больших различий в методике, коллекции и методах работы асессоров не очень корректно, но для аналогичной задачи поиска по Web коллекции [7], наши результаты варианта «weak» попадают в средние строки таблицы.

Стандартный 11-точечный график точность-полнота организаторами был рассчитан по двум методикам. Первая, оригинальная RI-RES, представлена на рис.1. Так как на момент написания этой статьи описание этого метода не было доступно, анализ этого графика произвести невозможно.

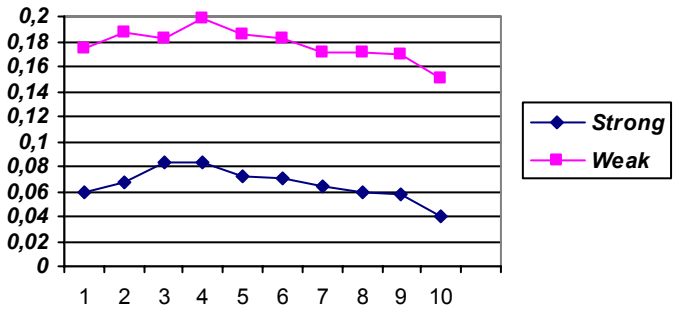

Рисунок 1. График точность/полнота рассчитанная по методике RI-RES

Второй график, по методике TREC представлен на рисунке 2. Использование стандартной методике расчета позволяет сравнить его с графиками, которые построены для систем, участвующих в аналогичных заданиях TREC.

Достаточно резкое снижение точности с ростом полноты, особенно для «weak» варианта, являются следствием того, что при реальной эксплуатации системы, ошибка выдачи большего количества результатов не столь важна, если функция взвешивания хорошо разместила в начале выборки важные для пользователя документы. При этом некоторое занижение точности даже полезно, т.к. позволяет пользователю косвенно оценить полноту базы системы и конкретность своего запроса. Поэтому, при разработке алгоритмов по-

иска, точности при больших выборках, как правило, не уделялось большого внимания.

Материал семинара еще требует дополнительной обработки, которую мы планируем провести в период до следующего семинара.

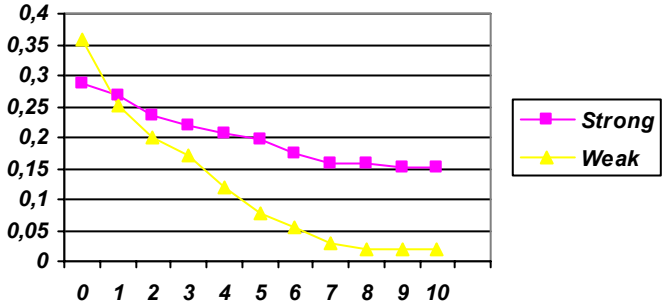

Рисунок 2. График точность/полнота, расчитанный по методике TREC

5 . Что планируется сделать на следующем семинаре.

Как всегда, первый блин, отчасти, вышел комом. Нехватка времени, отсутствие опыта не дали возможности достаточно адаптировать систему. Имея сейчас тестовую коллекцию и результаты первого семинара, мы сможем значительно лучше подготовиться к следующему. Для участия в дорожке ad hoc по Web-коллекции требуется доработка индексатора, также необходимо реализовать учет гипертекстовых ссылок между документами и форматирования текста. На следующем семинаре мы планируем выполнить несколько прогонов, которые покажут, оказывает ли учет этих параметров такое же влияние на качество поиска, как и для более привычных для нас коллекций.

Мы также надеемся, что появится специализированная дорожка по «контролируемым» коллекциям документов.

Конечно, мы понимаем, что другие участники не будут стоять на месте, и мы сможем сравнить, насколько быстро мы развиваемся по сравнению с другими организациями, занимающимися разработками в данной области.

Благодарности

Я выражаю свою благодарность всему коллективу разработчиков «Кодекса», которые бескорыстно помогали осуществить эту работу.

Я очень благодарен своим домашним, которые терпели воющий вентилятор ноутбука в течение не одной ночи.

Большие благодарности всем организаторам РОМИП и особенно Игорю Некрестьянову, без усилий которого однозначно ничего бы не получилось.

Литература

[1] Zlib Home Site http://www.gzip.org/zlib

[2] F. Scholer, H.E. Williams, J. Yiannis, J. Zobel Compression of Inverted Indexes For Fast Query Evaluation (2002)

[3] S. Melnik et al. Building a distributed full-text index for the web. Technical report, Stanford Digital Library Project, July 2000. Available at wwwdiglib. stanford.edu/cgi-bin/get/SIDL-WP-2000-0140. http://citeseer.nj.nec.com/melnik00building.html

[4] S.E. Robertson, K. Sparck Jones, Simple, Proven Approaches to Text Retrieval, http://www.uky.edu/~gbenoit/637/SparckJones1.html

[5] М. Губин Исследование качества информационного поиска с использованием пар слов. RCDL'2003

[6] SAX http://www.saxproject.org

[7] David Hawking and Nick Craswell Overview of the TREC-2001 Web Track.