Система поиска по Internet/intranet ресурсам с учетом русской морфологии

Григорьев А.С.
Ульяновск, Россия
ga@ulstu.ru

Валкин Л.М.
Ульяновск, Россия
vlm@ulstu.ru

1 ВВЕДЕНИЕ

«Продолжается информационный бум - растут количество и объемы серверов в WWW, увеличиваются мощности локальных сетей. Исключительно актуальна проблема поиска нужной информации. Всевозможные средства поиска разрабатываются крупнейшими компьютерными корпорациями... но без учета нашего "великого и могучего, правдивого и свободного", на 80% изменяемого русского языка». Такими словами начинается описание одного из лидеров на рынке русскоязычных поисковых систем – полнотекстовой системы индексации с учетом морфологии русского языка «Яndex» [3].

Действительно, высокая синтетичность русского языка, являющаяся предметом особой гордости его носителей, доставляет немало хлопот создателям поисковых систем. Это могло бы быть просто абстрактным рассуждением, если бы мы в повседневной практике не столкнулись с необходимостью реализации системы, объединяющей огромное количество разнородных и слабо связанных материалов, находящихся в разросшейся локальной сети нашего вуза. Как показала практика, авторы ресурсов, как правило, ограничены во времени и в энтузиазме, чтобы вести совместную деятельность по систематизации хранящихся и создающихся данных.

2 АНАЛИЗ СУЩЕСТВУЮЩИХ РЕШЕНИЙ

Очевидно, что одним из решений проблемы поиска информации является установка поискового сервера. Поскольку существующие на сегодняшний день поисковые системы имеют значительные различия по своей функциональности и эксплуатационным характеристикам, необходимо тщательно произвести их анализ. Как показало исследование, достигшие популярности продукты можно разделить на три основных класса:

1. Простые системы поиска по одному (реже – больше) серверу, часто без специальной обработки данных. Такие системы обладают очень ограниченной функциональностью, негибки в правилах поиска и сортировке и изначально имеют структуру, не позволяющую довести их до уровня корпоративного решения. Практически все такие системы написаны одиночками под свои локальные нужды и распространяются свободно (например «SimpleSearch» [9]).
2. Бесплатные системы, написанные большой группой программистов и предоставляющие такой уровень сервиса, который позволяет отнести их к профессиональным (к примеру «ht://Dig» [6], «UdmSearch» [7,8] и др.).
3. Профессиональные коммерческие решения, обладающие отлаженной технологией, проверенные временем и множеством заказчиков (например «Яndex» [3]).

Несколько наиболее распространенных систем поиска, тестирование которых нами проводилось, рассмотрим более подробно.

«SimpleSearch» (разработчик - Matt Wright) является довольно простым решением и вполне может подойти для одного сервера с небольшим объемом данных. В нашем случае, основным недостатком является невозможность индексации удаленных серверов (система работает только с файлами в локальных каталогах).

«UdmSearch» (разработчик - The UdmSearch Developers Team) использует для хранения данных реляционные СУБД (MySQL, PostgreSQL, miniSQL, Solid, iODBC и unixODBC). На наш взгляд это, во-первых, приводит к неоправданному усложнению системы, поскольку, в силу специфики поисковых систем, используется лишь малая часть от функциональности поддерживаемых СУБД. Во-вторых,

© Вторая Всероссийская научная конференция
ЭЛЕКТРОННЫЕ БИБЛИОТЕКИ:
ПЕРСПЕКТИВНЫЕ МЕТОДЫ И ТЕХНОЛОГИИ,
ЭЛЕКТРОННЫЕ КОЛЛЕКЦИИ
26-28 сентября 2000г., Протвино

увеличивает требования к аппаратному обеспечению системы.

«ht://Dig» (разработчик - The ht://Dig Group) можно отнести к классу профессиональных freeware систем. Однако в наших условиях она также имеет ряд недостатков. Отсутствует поддержка кодировок русского языка и русской морфологии. Получающиеся в процессе индексирования базы данных имеют довольно большой объем, что неприемлемо в нашем случае, поскольку объемы хранимой на серверах вуза (в том числе и внутренних) информации уже измеряются десятками гигабайт (зеркала сайтов, брошюры, информационно-методические ресурсы и др.). Отсутствует гибкий язык запросов (скобочные выражения, дополнительные модификаторы), наличие которого при поиске в большом объеме тематически (и лингвистически) сходных материалов рассматривается как обязательный критерий.

«Яndex» (разработчик - CompTek) одна из самых лучших профессиональных поисковых систем. Она свободна от большинства вышеперечисленных недостатков. Однако стоимость продукта для объема ресурсов, сопоставимого с объемами внутри нашего вуза составляет порядка $15000. Очевидно, что для вуза такая система практически недоступна.

В итоге, после изучения и тестирования различных поисковых систем было принято решение о создании собственного продукта, который бы наилучшим образом отвечал нашим требованиям:

1. Низкое требование к дисковой памяти при индексации. Базы данных индекса не должны превышать 25% от объема проиндексированных текстовых документов.
2. Поддержка русской и английской морфологии как эффективного средства улучшения релевантности результатов поиска.
3. Автоматическое определение кодировки документа. В условиях, когда «война кодировок» еще не окончена, нельзя полагаться на грамотную выдачу типа кодировки обслуживающим документ Web-сервером. В случае же FTP ресурсов, такая информация отсутствует вовсе. Все это определяет необходимость использовать элементы эвристического анализа для автоматического определения кодировки документа.
4. Система должна работать со стандартной переносимой базой данных во избежание проблем с привязкой к множеству существующих решений.
5. Система должна быть открытой, модульной и базироваться на исходном коде различных публичных подсистем, чтобы заранее убрать из рассмотрения мелкие ошибки и исключительные ситуации, которые были уже пройдены другими программистами.
6. Исходный код системы должен быть свободно доступен для использования, тестирования и внесения исправлений заинтересованными лицами.

3 МЕТОДЫ ОБРАБОТКИ И ХРАНЕНИЯ ИНФОРМАЦИИ В ПОИСКОВОЙ СИСТЕМЕ

Для поддержки морфологии русского и английского языков мы проработали и использовали следующий подход. Модуль индексации пытается выполнить приведение слова к корню. Хранение корней вместо исходных словоформ позволяет сократить общий размер баз данных в пять и более раз. Теряемая при этом гибкость для большинства применений несущественна. Многие системы имеют в своем составе словарь корней слов (90 тысяч слов в «Яndex») и генератор аффиксов, а объединение и генерацию словоформ осуществляют для каждого запроса. Это лишние затраты времени и прочих ресурсов, поэтому нам показалось более удобным создать базу со списком словоформ для русского и английского языка, а затем просто использовать ее. За основу были взяты стандартный английский словарь для системы ispell (для Unix) и русский словарь для ispell Александра Лебедева [18]. Генератор аффиксов для русского словаря был модифицирован в соответствии с рекомендациями Алексея Тутубалина [19] для достижения более приемлемых результатов при генерации словоформ. При объединении русского и английского словаря с соответственными генераторами аффиксов получились две базы данных формата BTREE (Berkeley DB). Одна для приведения слов к их корням, а вторая для расширения корней до возможных словоформ. Базы данных занимают, соответственно 28,3Мб и 15Мб. Они содержат достаточно полный список словоформ русского и английского языков (примерно 750 тысяч слов).

На производительность системы существенно влияют характеристики используемой СУБД. Мы решили использовать библиотеки лидера в мультиплатформенном программировании поддержки встроенных баз данных, продукта Berkeley DB от Sleepycat Software [10]. Это позволило создать не только переносимый продукт (компилируется как под большинством ветвей Unix, так и под операционными системами Microsoft Windows). Благодаря мощным встроенным средствам кэширования и сортировки, работа с базами данных проходит исключительно быстро. Стоит сказать, что работа широко распространенного агента передачи электронной почты sendmail [11], обрабатывающего три четверти всего почтового трафика в Интернет, основывается на использовании BerkeleyDB.

Для увеличения релевантности поиска при индексировании учитываются и сохраняются в БД следующие параметры:

- Условный признак каждого слова, характеризующий вес слова для каждого документа и еще некоторые параметры. Вес определяется в зависимости от частоты повторения слова в документе и от HTML тегов (<H1>-<H6>, <B>, <U>,<I>,<TITLE> и др.), в которые было заключено слово.
- Позиция (или позиции) слова в документе, что позволяет для выражений с несколькими словами учитывать расстояние между ними.
- Полученная от сервера дата создания/модификации и дата индексирования документа.
- Количество ссылок на каждый документ с других ресурсов.

Перед сохранением в БД тело документа приводится к текстовому формату, то есть очищается от HTML тегов и прочей ненужной информации. Убираются символы перевода строки, по возможности преобразовываются переносы слов. Все это дает нам на выходе «чистый» и компактный текст.

Хранение полного текста документа применяется поисковыми системами для того, чтобы при выводе документов выводить тот кусок, в котором встречается искомое слово или слова. По мнению авторов это увеличивает презентабельность выводимых результатов. Но, несмотря на очевидный выигрыш в презентабельности, такой подход оказывается слишком накладен с точки зрения дискового пространства под хранение документов. Более того, например, в случае с нашим вузом, большое количество серверов содержит «зеркала» каких-либо ресурсов, содержимое лазерных дисков, методические информационные ресурсы и прочие объемные данные. В отличие от Интернет, в интранет меньше считаются с размерами документов. Поэтому создавать в этих условиях сервер, который содержит 30-40% от внутривузовского объема данных только для презентабельности, мы считаем необоснованным. На наш взгляд, вывод первых нескольких предложений от документа (в которые попадают заголовки, определения, первые строчки текста) является хорошей альтернативой. В этой связи, с целью обеспечения возможности значительного сокращения размера генерируемых индексной частью БД, было решено сделать объем сохраняемой части документа переменной величиной. Она будет определяться настройками и может задаваться как в байтах, так и в процентах от всего текста документа. Так, например, чтобы размеры генерируемых БД не превышали 20% от объема индексируемых ресурсов, можно сохранять только первые 200 байт документа.

Также для экономии места на диске в системе реализуется опциональная возможность использования различных методов компрессии для сохраняемой информации, как общепринятых (библиотека ZLIB [12]) так и собственных, оптимизированных под конкретные нужды [14].

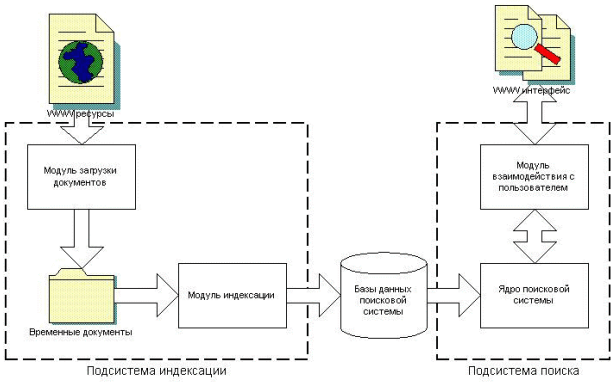
В любом языке присутствуют слова, практически не несущие смысловой нагрузки (союзы, местоимения и др.). Но поскольку они встречаются достаточно часто, то объем связанной с ними информации становится значительным. В то же время, в запросах к поисковым машинам эти слова не используются совсем или используются, но только для удобства человека (в запросах близких по форме к естественному языку). Например, запросы «программа которая форматирует диск» и «программа форматирующая диск» можно считать эквивалентными с точки зрения естественного языка, а так как мы решили хранить корни слов, то при исключении слова «которая» мы получаем эквивалентность и с точки зрения поисковой системы. Таким образом, мы сочли целесообразным игнорирование ряда слов, как русского, так и английского языков (далее будем называть их стоп-словами) и на стадии индексирования, и в процессе поиска. Из перечня таких слов составлен список стоп-слов. Такой подход позволит нам дополнительно сэкономить дисковое пространство и улучшить временные параметры поиска, практически не теряя на функциональности.

Анализатор входных выражений (запросов) строится с использованием средств генерации лексического анализатора yacc. Это позволяет при минимальных затратах времени и сил создать качественный анализатор сложных выражений, причем даже близких по форме к естественному языку [16]. Стоит заметить, что многие бесплатные профессиональные продукты не имеют развитых средств обработки выражений («ht://Dig» не способен обрабатывать выражения в круглых скобках, а «UdmSearch», имеет подобные средства только в дополнительном модуле и для этого необходимо использовать PHP3).

4 ОБЩАЯ СТРУКТУРА СИСТЕМЫ

Для возможности распределения нагрузки на программно-аппаратную платформу было принято решение о разделении системы на несколько модулей. Подсистема индексации представляет собой два параллельно работающих модуля. В задачу первого входит загрузка документов с WWW серверов, а в задачу второго - их индексация и сохранение результатов в БД. Подсистема поиска также разбита на две части, одна из которых представляет собой ядро системы и выполняет исключительно функции поиска, а другая отвечает за взаимодействие между ядром и пользователем. Модули подсистемы поиска

обмениваются информацией через сетевое соединение, что позволяет им функционировать на разных ЭВМ. Ядро системы постоянно загружено и находится в ждущем режиме, такой подход позволяет сократить время на обработку запросов за счет исключения из каждой операции поиска этапов инициализации. В целом структуру системы можно представить следующей схемой:



5 ПРИНЦИПЫ ФУНКЦИОНИРОВАНИЯ СИСТЕМЫ

5.1 Модуль загрузки документов

Это программа, осуществляющая обход дерева документов на сайте (сайтах), их загрузку и сохранение во временном каталоге для их последующего индексирования. Она представляет собой модифицированный web-spider, для реализации которого был использован wget-1.5.3 [13] (аналоги – ReGet, Teleport Pro, NetVampire).

Использование готового средства позволило решить нам следующие задачи:

- Освобождение от необходимости повторять стадии разработки и тестирования для данного типа функциональности, следовательно, экономия ресурсов и времени.
- Использование функциональности лидера на смежном рынке позволяет добиться превосходящих результатов и в своей области. Например, «ht://Dig», опираясь на свою, разработанную «с нуля» систему загрузки документов, не поддерживает индексирование FTP ресурсов, не содержит средств оптимизации нагрузки на сервер при индексировании и только начинает разрабатывать некоторые другие, уже встроенные во wget функции.

5.2 Модуль индексации

Данная программа производит обработку загруженных документов. Одна генерирует несколько БД [14]:

- БД слов – хранит некоторые параметры слов и соответствия между словами и идентификаторами документов.
- БД документов - хранит проиндексированные документы и еще ряд относящихся к ним параметров.
- БД идентификаторов документов – хранит соответствия между URL документа и его условным идентификатором.
- Две БД статусов – используются для определения статуса идентификаторов документов (занят, свободен и др.).
- БД ресурсов – хранит соответствия между именем сервера (формата «proto://host.domain») и всеми относящимися к нему идентификаторами документов.

Индексация ресурса (сайта) или его ветви проходит следующим образом:

- Инициализируются БД.
- Производится разбор текущего документа полученного от модуля загрузки на слова. Стоп-слова игнорируются. Для каждого слова вычисляется его вес.
- Документу присваивается уникальный 32-битный идентификатор.
- Сохраняется тело документа, его URL и прочая служебная информация.
- Повторяющиеся слова сливаются с учетом их весов и параметров. Полученный список слов сохраняется.
- Переходим к следующему документу (к шагу 2) или же завершаем индексирование, если документы закончились.

5.3 Модуль поиска (ядро системы)

В качестве лексем во входных параметрах модуль поиска воспринимает следующие ключевые слова и символы:

- «(» и «)» - Результат обработки содержимого скобок представить в виде самостоятельного объекта.
- «AND», «+», «&», «и» - Выделение общей части списков для слов.
- «OR», «|», «или» - Объединение списков.
- «NOT», «-», «не» - Исключение слов из списка.
- «URL=», «URL:» - Искать только в пределах сайта, адресуемого указанным URL и его подкаталогов.
- «”», «“» - Слова в кавычках должны находиться в пределах одного и того же предложения в документе.

Модуль поиска функционально представляет собой постоянно запущенный (или вызывающийся по

требованию из inetd) процесс, который обрабатывает заявки на поиск. К модулю поиска можно обратиться, создав TCP соединение с известным портом на поисковом сервере. Такая структура позволяет снять с ядра нагрузку по обеспечению интерактивного взаимодействия с пользователем, так как запросы и ответы от сервера производятся на специально разработанном простом языке запросов и ответов. Получив запрос, ядро производит поиск и сообщает запрашивавшему процессу результат поиска. Так как спецификация протокола запросов и ответов стандартизована [15], запросы могут производится как модулем взаимодействия с пользователем, так и произвольными программами, которым необходимо по тем или иным причинам искать информацию.

Основными требованиями к модулю являются его устойчивость к сбоям и скорость обработки запросов. Важным моментом в увеличении скорости обработки является совершенствование алгоритма объединения списков слов, полученных на этапе разбора запроса. Например, в случае объединения двух слов по «И», для каждого из которых найдено несколько тысяч документов, простейшие поисковые системы произведут неэффективное попарное сравнение, что повлечет загрузку системы на секунды. Ядро же поискового сервера SnapCat содержит развитый математический и эвристический аппарат для минимизации «стоимости запроса». Типичные запросы, содержащие объединения крупных списков документов, выполняются не дольше одной десятой секунды, из которых на собственно объединение приходится порядка нескольких миллисекунд (машина 2xP166).

5.4 Модуль взаимодействия с пользователем

Модуль взаимодействия с пользователем является процессом, запускающимся во время запроса пользователя по HTTP. Организованный в виде CGI-программы, модуль принимает от Web-клиента данные для поиска, производит сетевой запрос к поисковому ядру и отображает результаты, форматируя их в соответствии с шаблонами из конфигурационного файла. В задачи этого модуля входит только формирование корректного запроса к ядру системы и переработка результатов в удобный для человека вид.

Этот модуль может выполняться на какой угодно машине в сети, так как запрос происходит по указанному адресу сервера, на котором работает ядро.

6 ПОЛУЧЕННЫЕ РЕЗУЛЬТАТЫ

В общем случае, можно выделить следующий перечень основных характеристик, присущих поисковым системам:

- Возможности языка запросов.
- Скорость поиска.
- Скорость индексирования.
- Объем генерируемых БД.
- Возможность распределения нагрузки между несколькими программно-аппаратными платформами.

Как уже говорилось выше, благодаря использованию уасс нам удалось реализовать языка запросов, который по функциональности приближается к лучшим коммерческим аналогам.

Кроме 28,3+15 мегабайт словарей, если настроить систему на оптимизацию по размеру, то сгенерированные базы данных для нескольких сайтов общим объемом в 98 мегабайт (4832 документа) заняли порядка 15 мегабайт. Например у «ht://Dig:» при использовании тех же словарей, индексация всего лишь 1390 документов (18 мегабайт) создала базы данных общим объемом в 46 мегабайт, что на наш взгляд объясняется избыточностью хранимой информации и использованием не оптимальных методов хранения.

По скорости индексации, система практически не уступает лучшим коммерческим продуктам. На сайтах с преобладанием документов большого размера средняя скорость достигает 2 мегабайт в минуту. В среднем же от 1,2 до 1,5 мегабайт в минуту на машине 2xP166/128/SCSI-HDD. Для сравнения, заявленная скорость у индексатора «Яndex»: 1-2 мегабайта в минуту. «ht://Dig» не имеет встроенных в индексатор средств непосредственного создания базы (создание идет через промежуточный текстовый файл), а поэтому скорость, слагающаяся из индексации и объединения с базами существенно меньше.

Что касается скорости поиска. Тестирование проводилось на машине 2xP166/128Mb/SCSI-HDD. Запрос «профессора & кандидаты технических наук» (найдено 17 документов) в среднем выполнялся 0.167 секунд. Запрос «профессора | кандидаты технических наук» (найдено 570 документов) - 0.150 секунд. Те же запросы на в 5 раз меньшем объеме данных (1390 документов, 18 мегабайт) «ht://Dig» выполнял соответственно 1.56 и 1.16 секунд.

Благодаря разделению системы на несколько составляющих, четкому разграничению выполняемых функций и реализованной схеме взаимодействия стало возможным не только организовать параллельную работу нескольких модулей, что позволяет эффективно использовать многопроцессорные системы, но и распределить поисковую систему между несколькими ЭВМ. Например, подсистема индексирования работает на одной, ядро системы – на второй, модуль (а точнее любое их количество, в зависимости от числа обращений) взаимодействия с пользователями – на третьей.

Разумеется, система не лишена недостатков. К одним из основных следует отнести невозможность запросов, где некоторое количество символов заменяется на один

символ «*». Это объясняется выбранной схемой хранения слов. Однако стоит заметить, что необходимость введения поиска по подстрокам в значительной степени снимается из-за реализации поддержки морфологии русского и английского языков. В подавляющем большинстве случаев возможность поиска по подстрокам используется в качестве средства для выделения окончаний и сведения сложного запроса («средство | средства | средству | средством») к более простому («средств*»), что в нашей системе выполняется автоматически. Но, тем не менее, уже разрабатываются методы, которые позволят включить такую возможность без существенной переработки системы и потерь в производительности.

Следующим недостатком можно считать невозможность параллельной работы нескольких модулей индексирования (обращаем внимание, что именно несколько копий индексатора не могут работать с одной БД, но это не мешает запускать несколько модулей загрузки документов, которые могут работать параллельно с модулем индексации). Это связано с использованием внутреннего механизма кэширования, который был введен с целью увеличения быстродействия. Одним из вариантов, который уже прорабатывается и позволит устранить этот недостаток, является дальнейшее повышение модульности системы.

7 ЗАКЛЮЧЕНИЕ

Для недавно созданной системы сам факт ее функционирования уже можно считать успехом. Разработанные методы обработки и хранения информации показали свою состоятельность, что подтверждается продемонстрированными системой результатами.

Полученная поисковая система имеет очень небольшие требования к аппаратной части компьютера и программному окружению. Кроме этого, в отличие, например, от «UdmSearch», она самодостаточна и не нуждается во внешних SQL-серверах. Система имеет гибкие настройки, что позволяет в зависимости от условий регулировать размеры генерируемых БД и скорость поиска, выбирая оптимальное сочетание.

Механизмы индексации, релевантность поиска и возможности языка запросов постоянно совершенствуются, для чего ведутся дальнейшие исследования и изучаются продукты других разработчиков.

Список литературы

[1] Knuth, D.E., "Sorting and Searching", The Art of Computer Programming, Vol. 3, pp. 114-123, 145-149, 1968.

[2] Hoare, C.A.R., "Quicksort", The Computer Journal, 5:1, pp. 10-15, 1962.

[3] http://www.yandex.ru/ - Поисковая система "Яndex".

[4] http://www.aport.ru/ - Информационно-поисковая система "Апорт 2000".

[5] http://www.rambler.ru/ - Информационно-поисковая система "Рэмблер".

[6] http://www.htdig.org/ - Поисковая система "ht://Dig".

[7] http://mysearch.udm.net/ - Поисковая система "UdmSearch".

[8] http://search.udm.net/ - Поисковый сервис "Вся Удмуртия".

[9] http://www.scriptarchive.com/ - "SimpleSearch".

[10] http://www.sleepycat.com/ - Домашняя страница Sleepycat Software.

[11] http://www.sendmail.org/ - Сайт посвященный агенту Sendmail.

[12] ftp://ds.internic.net/rfc/rfc1950.txt, ftp://ds.internic.net/rfc/rfc1951.txt, ftp://ds.internic.net/rfc/rfc1952.txt – Формат данных используемый библиотекой zlib.

[13] http://sunsite.auc.dk/ftp/pub/infosystems/wget/, ftp://prep.ai.mit.edu/pub/gnu/wget-1.5.3.tar.gz, ftp://gnjilux.cc.fer.hr/pub/unix/util/wget/wget.tar.gz – Последняя версия Wget.

[14] http://admin.ulstu.ru/itr.cgi?ITR-0006 - Описание форматов БД поисковой системы SnapCat.

[15] http://admin.ulstu.ru/itr.cgi?ITR-0005 - Протокол удаленных запросов к поисковому серверу SnapCat.

[16] http://www.ulstu.ru/snapcat/help.html - Справка по форматам запросов к SnapCat.

[17] http://www.ulstu.ru/snapcat/search.html, http://www.ulstu.ru/snapcat/snapcat.html - Поисковый сервис SnapCat.

[18] ftp://mch5.chem.msu.su/pub/russian/ispell/ - словарь Александра Лебедева.

[19] http://www.lexa.ru:8101/lexa/russian.aff.patch - Патч к версии 0.99c9 словаря Лебедева.