Список использованных источников

1. Горбенко П. К. Природа и свойства дырочных центров в кристаллах KCI / П. К. Горбенко, А. А. Ковтун // ФТТ. – 1971. – № 13. – С. 2766-1769.
2. Delbecq C., Hutchinson E., Schoemaker D., Yasaitis E., Yuster P. // Phys. Rev. – 1969. – № 187. – С. 1103.
3. Мелик-Гайказян И. Я. Радиационное образование бивакансий и катионных вакансий в CKt / И. Я. Мелик-Гайказян, Э. П. Куракина // Изв. АН СССР. Сер. физическая. – М., 1971. – Т. 35. – С. 1360-1363.

УДК 004.775

Е.В. Никитенко, канд. физ.-мат. наук
Р.В. Заровский, канд. техн. наук
М.А. Сдитанов, магистрант
Черниговский государственный технологический университет, г. Чернигов, Украина

АВТОМАТИЧЕСКАЯ СИСТЕМА ПОИСКА ИНФОРМАЦИИ В ТЕКСТАХ С ИСПОЛЬЗОВАНИЕМ АЛГОРИТМОВ НЕЧЕТКОЙ ЛОГИКИ

Разработана архитектура автоматической поисковой системы. Данная архитектура состоит из двух частей – сервис индексирования и сервис поиска. В сервис индексирования входят сервисы разбора документов и построения индексов и вспомогательные сервисы, обеспечивающие сохранение, загрузку и обработку данных для индекса. Сервис поиска разработан для быстрого доступа к поисковому индексу и получения релевантных данных.

***Ключевые слова**: поисковая система, веб-сервер.*

Розроблено архітектуру автоматичної пошукової системи. Ця архітектура складається з двох частин - сервіс індексування та сервіс пошуку. В сервіс індексування входить сервіси розбору документів і побудови індексів та допоміжні сервіси, що забезпечують збереження, завантаження й обробку даних для індексу. Сервіс пошуку розроблений для швидкого доступу до пошукового індексу й отримання релевантних даних.

***Ключові слова**: пошукова система, веб-сервер.*

The architecture of the automatic retrieval system was developed. The architecture consists of two parts – the indexing service and search service. Indexing service includes services for documents analysis and index construction and support services to ensure the preservation, loading and processing of the data for the index. Search service was developed for easy access to the search index and to obtain relevant data.

***Key words:** a retrieval system, web-server.*

Постановка проблемы. Необходимость в разработке такого рода системы вызвана наличием значительных объемов информации, сопутствующей процессам развития онлайн-ресурсов и Интернета в целом, и требованиями к быстрому доступу к нужной информации. В качестве основной причины создания поисковой системы можно выделить закрытость архитектуры и алгоритмов, узкая направленность поисковых алгоритмов в существующих поисковых системах. При этом можно выделить следующие недостатки поисковых систем:

- закрытая реализация поисковых алгоритмов;
- трудности с определением спам-текстов;
- трудность расширения и поддержки аналогов с открытым исходным кодом;
- неэффективная реализация поисковых алгоритмов в поисковых системах с открытым исходным кодом.

Поисковые алгоритмы разрабатываются уже несколько десятилетий, не раз изменялись требования к производительности поисковых алгоритмов и к поисковым системам в целом. В качестве основных требований к разрабатываемой системе были предъявлены следующие:

- модульность;
- масштабируемость;
- поэтапная реализация.

Анализ исследований и публикаций. Поисковая система представляет собой программно-аппаратный комплекс с веб-интерфейсом, обеспечивающий возможность поиска информации [1]. Программной частью поисковой системы является поисковая машина – комплекс программ, обеспечивающий функциональность поисковой системы и обычно являющийся коммерческой тайной компании-разработчика.

Поиск информации представляет собой процесс выявления в некотором множестве документов, которые посвящены указанной теме, удовлетворяют заранее определенному условию поиска или содержат необходимые факты, сведения, данные. Процесс поиска включает последовательность операций, направленных на сбор, обработку и предоставление необходимой информации заинтересованным лицам [2].

В общем случае поиск информации состоит из четырех этапов:

1. Определение (уточнение) информационной потребности и формулировка информационного запроса.

2. Определение совокупности возможных держателей информационных массивов (источников).

3. Извлечение информации из выявленных информационных массивов.

4. Ознакомление с полученной информацией и оценка результатов поиска.

Архитектура поисковой системы. Архитектура поисковой системы должна обеспечивать эффективную обработку данных, максимальную масштабируемость и возможность параллельной обработки данных.

В любой поисковой системе можно выделить три базовых части [3]:

1. Робот (crawler, spider) – сервис, который отвечает за сбор информации. Робот эмулирует работу пользователя, загружая страницы и сохраняя их в семантической базе данных.

2. Семантическая база данных. В базе данных хранится и сортируется собранная роботом информация.

3. Клиент. В этой части обрабатываются пользовательские запросы. В действительности клиент может быть разнесён по нескольким физически несвязанным компьютерам. Однако стоит отметить, что все эти компьютеры должны иметь доступ к базе данных.

Также можно выделить дополнительные подсистемы в архитектуре поисковой системы:

1. URL Server – список всех адресов.

2. Crawler – робот, который загружает страницы из списка адресов и передает в Store Server.

3. Store Server сохраняет страницы в Repository, чаще всего в виде текста документа. При этом вся дополнительная информация, такая как картинки, flash-анимация и прочее, не сохраняется.

4. Indexer разбирает сохраненные в Repository HTML-документы в последовательности слов и сохраняет их в базе данных.

5. Lexicon – список всех слов. Чаще всего слова хранятся в таблице с двумя полями "номер" и "слово". Таким образом достигается экономия места в базе данных, так как длинные слова заменяются достаточно коротким номером.

6. Anchors – выделенные компонентом Indexer ссылки (URL).

7. Links определяет ссылки одних сайтов на другие и передает это в PageRank.

Взаимодействие сервисов изображено на рисунке 1.

Рассмотрим варианты хранения записей в базе данных, т. е. параметры, по которому они отсортированы, и какая дополнительная информация хранится для каждой записи. В связи с этим появляются два понятия: прямой и обратный индексы.

В случае прямого индекса записи отсортированы по номеру документа. Для каждой записи хранится отсортированный по номеру список слов. Для каждого слова хранятся первые несколько (например, восемь) позиций вхождения слова в документ, количест-

во вхождений и формат вхождения. Под форматом вхождения подразумевается вхождение слова в текст ссылки, в описание к картинке, в заголовок и т. д. Такие слова будут иметь приоритет при поиске. Прямой индекс обновляется постоянно при работе робота. Для каждой страницы в базе данных хранится частота предполагаемого обновления, которая считается следующим образом: при очередном заходе робота на эту страницу в случае отсутствия обновлений частота увеличивается в два раза, а если страница за этот период времени менялась, то уменьшается. Также стоит отметить, что чаще всего робот индексирует не все слова из документа (например, только первую тысячу слов) и не все документы с одного сайта.

Обратный индекс используется клиентом при поиске. В этом случае записи отсортированы по словам. Для каждой записи хранится номер слова, список документов, в который входит это слово, и полная информация о вхождении. Отметим, что обратный индекс обновляется не так часто как прямой, а примерно раз в минуту.

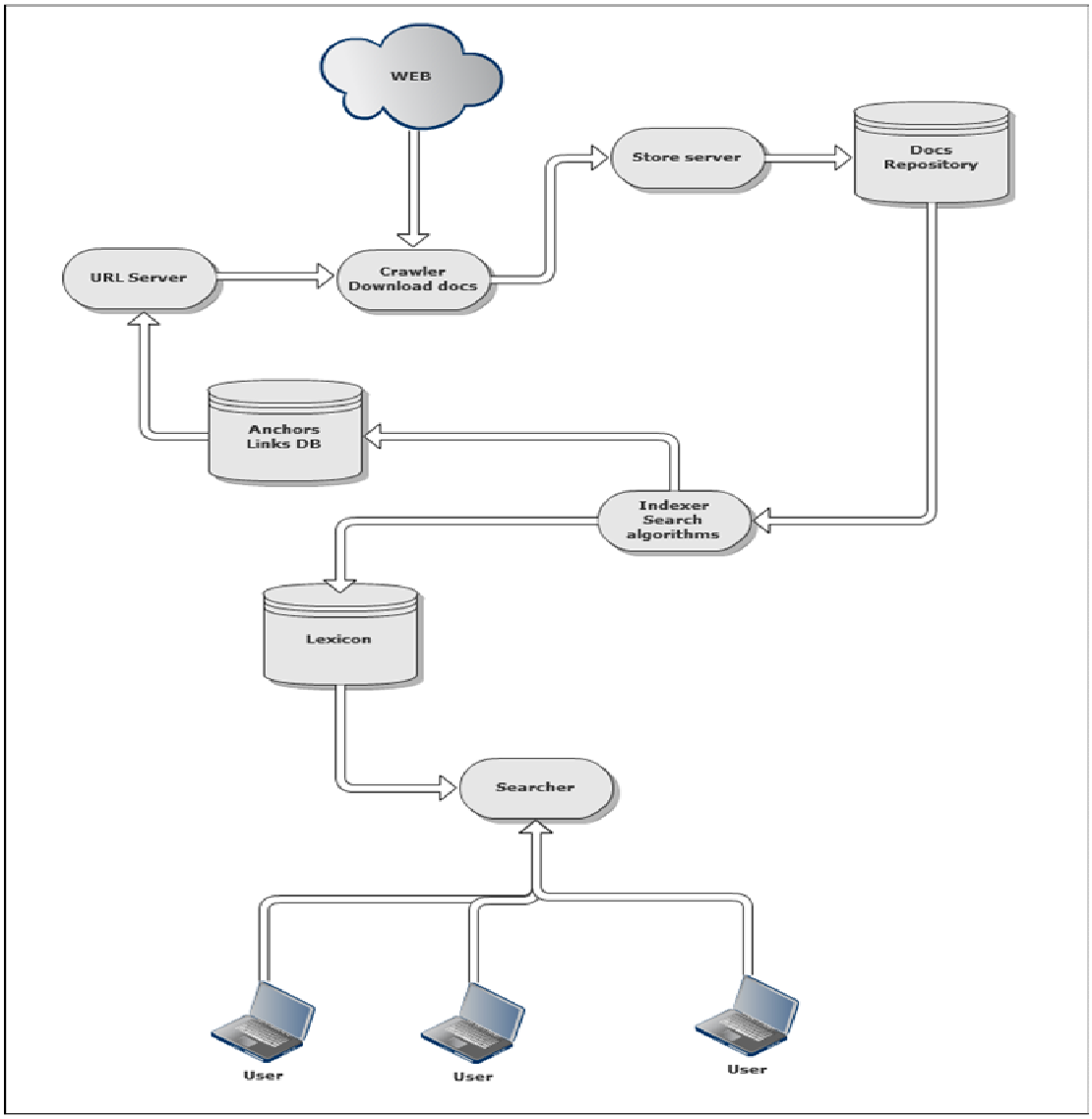

Рис. 1. Схема взаимодействия сервисов

Релевантность и позиция документа в поиске. Под релевантностью понимается то, насколько слово (совокупность слов) соответствует данному документу. Рассмотрим характеристики, которые могут влиять на позицию документа в списке ответов:

– Наличие слов в документе. Очевидно, что если слова в документе не встречаются, то данный документ не подходит под условия поиска.

– Частота вхождения слов. Чем чаще слова встречаются на странице, тем выше документ окажется в списке поиска.

– Форматирование слов. Если в документе слова встречаются в виде выделяющих тэгов, заголовков, описаний картинок, то такой документ будет иметь более высокий приоритет.

– В случае набора слова значение также имеют расстояние между этими словами в документе и их порядок.

– В некоторых поисковых системах (например, Яндекс) значение имеет морфологическое вхождение слов, т. е. падеж (род, лицо), в котором слово входит в документ.

– Регистрация в каталоге поисковой системы. Это очень важная характеристика, так как каталоги составляются вручную и в них уже заданы разделы и тематика страницы.

Работа клиента. Пользователю предоставляется возможность делать запросы как в естественном виде, так и с помощью SQL-подобного языка. Use-case диаграмма взаимодействия пользователя с системой достаточно тривиальна и изображена на рисунке 2.

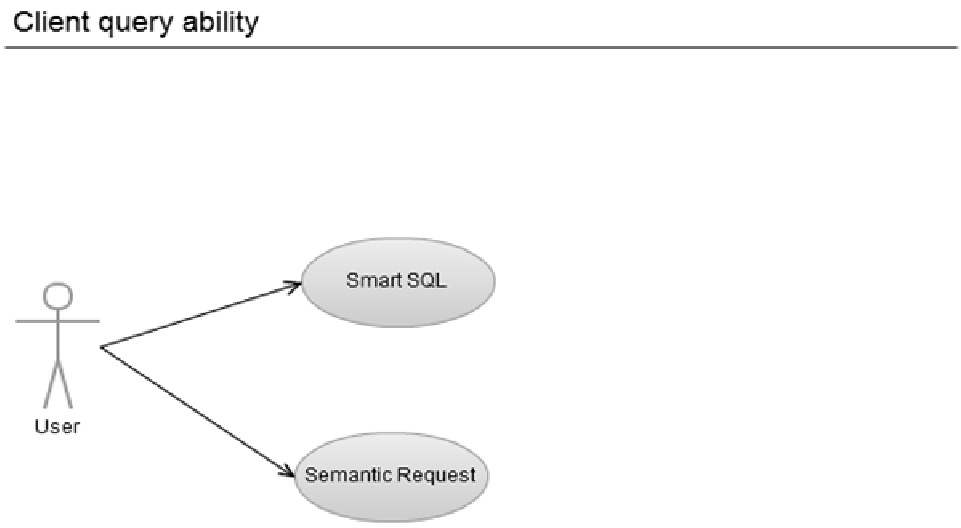

Рис. 2. Схема взаимодействия пользователя с поисковой системой

Сначала запрос разбивается на слова. Далее удаляются так называемые стоп-слова – слова, которые встречаются почти во всех документах (предлоги, союзы). На следующем шаге каждому слову сопоставляется его номер из "словаря". Для каждого слова из запроса находится в обратном индексе список документов, который содержит это слово. Из этих списков создаётся новый, содержащий те и только те документы, которые входили в списки для всех слов. Затем для каждого из документов вычисляется степень релевантности, и список сортируется по этому признаку. На этом шаге для всех документов создаются аннотации. Аннотацией может быть содержание тэга "description", контекст вхождения слов из запроса (наиболее близко стоящих или первое вхождение), первое предложение или заголовок документа.

Параллелизм в поисковой архитектуре. Крупные поисковые системы имеют базу размером в десятки миллионов документов и ежедневно обрабатывают миллионы пользовательских запросов, причём с каждым месяцем (с ростом количества пользователей Интернета) эти цифры ощутимо увеличиваются. В этих жёстких условиях главная задача поисковых систем – сохранение приемлемых для пользователей скорости и полноты выполнения запросов. Для запроса средней "тяжести", т. е. при поиске не

очень часто встречаемого слова, время отклика системы (без учёта времени передачи данных по каналу от поисковой системы к пользовательскому компьютеру) должно исчисляться десятыми долями секунды.

На сегодняшний день известны три основных подхода к решению этой проблемы:

1. Оптимизация базовых поисковых алгоритмов и архитектуры поиска.
2. Регулярное увеличение мощностей вычислительных ресурсов поисковой системы.
3. Использование архитектурной возможности масштабирования системы (если масштабируемость была заложена при проектировании системы).

Оптимизация поисковых алгоритмов и архитектуры поиска – это предмет постоянного внимания разработчиков. Увеличение мощностей – это регулярный переход на более мощные процессоры, добавление оперативной памяти, увеличение объёма жёстких дисков. Несмотря на то, что тактовая частота процессоров увеличивается каждый месяц, новая техника "не успевает" за ростом потребностей пользователей. К тому же постоянное обновление оборудования весьма недешево. Поэтому наряду с этим подходом используется масштабируемость архитектуры.

Архитектура поискового веб-сервера (Web Searcher). Рассмотрим параллельные решения на всех уровнях для архитектуры поисковой системы. На рисунке 3 изображена архитектура сервиса Web Searcher, который обслуживает запросы от пользователей системы.

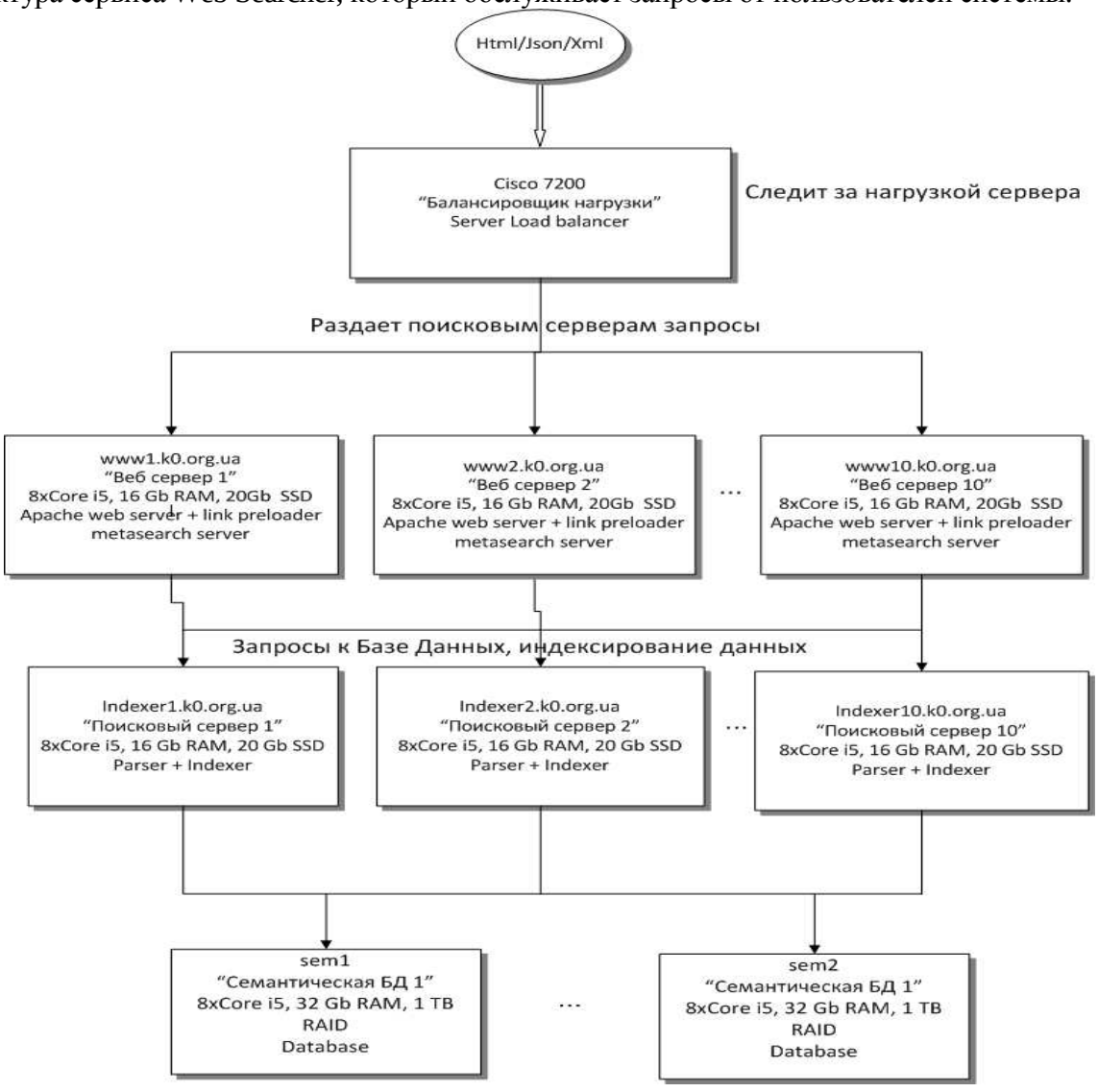

Рис. 3. Архитектура поискового веб-сервера

Маршрутизатор распределяет запросы от клиентов на поисковые системы. Поисковые системы обрабатывают запросы пользователей, обращаясь к базам данных индекса. Распределение нагрузки между фронтальными веб-серверами выполняют Cisco – сетевые устройства балансировки нагрузки. Каждый пользователь в зависимости от своего IP-адреса перенаправляется на один из трех наименее загруженных веб-серверов, используемых в поиске. Заметим, что функционально фронтальные веб-серверы совмещены с модулями слияния поисковых результатов от веба и остальных поисковых источников.

Разделение коллекции документов. В разрабатываемой поисковой системе используется "параллельный поиск" в нескольких поисковых источниках. Параллельный поиск – это одновременный поиск в специализированных базах (коллекциях), предлагаемых поисковой системой. Обычно источник – это отдельная база небольшого размера, отличная от "большой" базы документов. Подразумевается, что документы, индексируемые в такой базе, имеют некоторую регулярную структуру.

Если при поиске по обычной базе находятся и документы из базы параллельного поиска, точно соответствующие запросу, то одновременно (параллельно) с обычным результатами поиска выдаётся список из нескольких найденных документов.

База параллельных источников имеет существенно меньший размер, чем база веб-поиска. Обход и индексация документов в ней осуществляются отдельным роботом, поэтому обновление базы может происходить очень быстро (вплоть до ежеминутного).

Выделяются следующие четыре базы параллельного поиска:

1. По новостям (обновление каждые 10 минут, ежедневно около 3000 новостей).
2. По товарным предложениям интернет-магазинов (ежедневное обновление приблизительно 300 000 товаров).
3. По заголовкам статей энциклопедий (обновление раз в месяц, около 200 000 статей).
4. По каталогу ("ручному" описанию веб-ресурсов).

Особый интерес с точки зрения традиционных поисковых технологий представляет техника разделения большой базы документов, т. е. собственно базы веб-страниц. Сейчас она состоит из 1 млн документов и разделена на 30 частей.

Среди особенностей текущей реализации разделения веб-коллекции можно отметить следующее:

- есть центр контроля и распределения URL;
- отсутствует репликация коллекции по машинам;
- распределение документов по коллекциям случайно.

Для выполнения поискового запроса выделяется три фазы (рис. 4).

Первая фаза обработки запроса: выбор коллекции, трансформации запроса. Пользователь может явно указать, в какой коллекции следует искать. Если этого не сделано, то на основе лингвистического (эвристического) анализа запроса поисковая система может сделать допущение о приоритете специализированной коллекции или подходящей к характеру запроса рубрики каталога.

Вторая фаза обработки запроса: раздача запроса по коллекциям. Обычно используются все коллекции. Собирающий сервер раздаёт в коллекции модифицированные запросы, в которых для каждого термина сообщается глобальное значение его обратной частоты (IDF в терминах традиционного IR). Для этого на всех "собирающих" серверах хранится глобальная статистика терминов. Она изменяется медленно, поэтому обновляется относительно редко. Статистика подсчитывается по считающейся наиболее универсальной веб-коллекции. Таким образом, каждая поисковая машина ищет ответ на запрос с назначенными "сверху" глобальными частотами, и значения релевантности, вычисляемые в разных коллекциях, можно считать последовательными и вычисляемыми "в одной системе координат". Модификации запросов этим не ограничиваются, и

для специализированных коллекций (например, "энциклопедии") могут быть и другими, в том числе и очень специфическими.

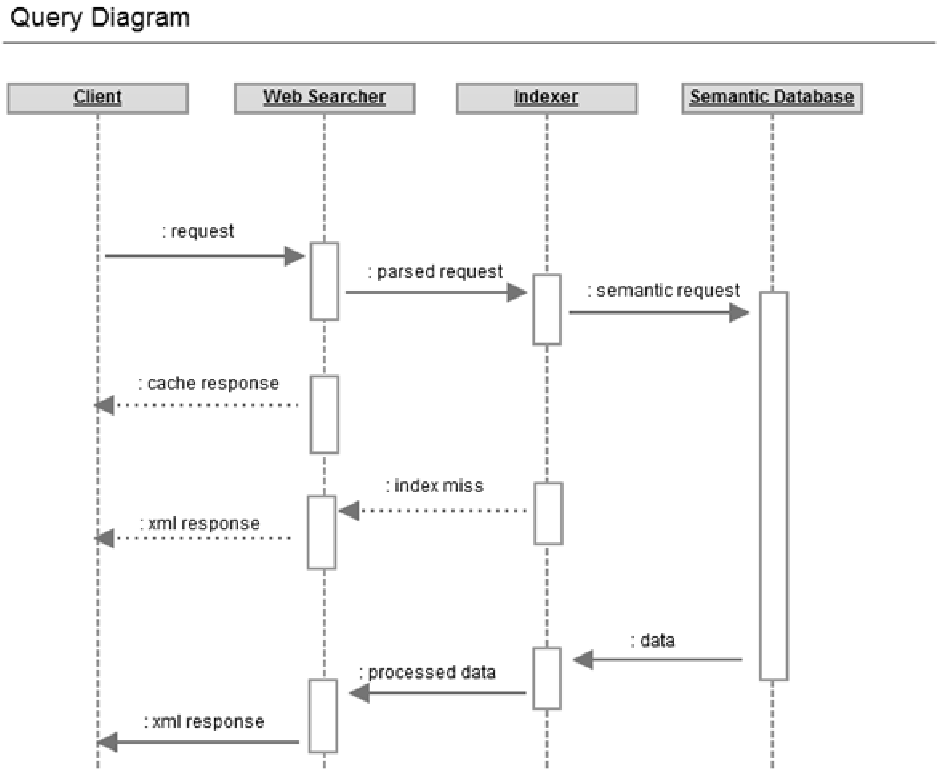

Рис. 4. Фазы обработки запроса

Третья фаза обработки запроса: исполнение и ранжирование запроса в коллекциях. Хотя в поисковой системе использовался стандартный полнотекстовый индекс, стоит упомянуть о процедуре вычисления неявных контекстных ограничений, применяемой в распределенной версии поиска. В этом случае на серверах "переднего края" производится синтаксический разбор запроса на основе ATN-грамматики, адаптированной к свободному порядку слов русского языка. В естественно-языковых фрагментах запросов выявляются несколько видов синтаксических связей (притяжение, перечисление, зависимости цели и места, счетные конструкции и др.) и устанавливаются эмпирически подобранные контекстные ограничения.

Глобальная для всех коллекций статистика слов используется как для "выравнивания" ранжирования между коллекциями, так и для корректировки контекстных ограничений в постсинтаксической фазе.

Выводы. Разработана архитектура автоматической поисковой системы. Данная архитектура состоит из двух частей – сервис индексирования и сервис поиска. В сервис индексирования входят сервисы разбора документов и построения индексов и вспомогательные сервисы, обеспечивающие сохранение, загрузку и обработку данных для индекса. Сервис поиска разработан для быстрого доступа к поисковому индексу и получения релевантных данных. Разработанная поисковая система по своим характеристикам не уступает открытому аналогу OpenSearch, а по скорости индексации разработанная система индексирует в 2,3 раза больше документов и обладает рядом качественных улучшений при ранжировании, такие как определение тематики, которые не присутствуют в OpenSearch.

Список использованных источников

1. Построение поисковых систем [Электронный ресурс]. – Режим доступа: http://habrahabr.ru/blogs/algorithm/114997.

2. Ландэ Д. В. Поиск знаний в Интернет. Профессиональная работа: пер. с англ. / Д. В. Ландэ. – М.: Диалектика, 2005. – 272 с.

3. Поисковые системы. Алгоритмы машинного обучения [Электронный ресурс]. – Режим доступа: http://www.scu.edu.au/programme/fullpapers/1938/com1938.htm.

УДК 004.934.2

Т.В. Шарий, канд. техн. наук
Донецкий национальный университет, г. Донецк, Украина

МОДЕЛЬ ПОСТОБРАБОТКИ РЕЧЕВЫХ СИГНАЛОВ FCAS

В статье рассматривается актуальный вопрос обработки параметров речевых сигналов в задаче распознавания речи. Указаны недостатки современных статистических моделей, с учетом которых предложена многоуровневая нечеткая когнитивная модель FCAS. Ядро FCAS представляет собой сеть фонетических процессоров, учитывающих вес речевого сегмента и работающих на признаковом, фонемном и словесном уровнях. Рассмотрена динамика и алгоритмы функционирования FCAS. Приведены результаты экспериментов с программной реализацией FCAS на тестовом словаре.

Ключевые слова: *речевой сигнал, когнитивное моделирование, фонема, элементарный фонетический процессор, FCAS.*

У статті розглядається актуальне питання обробки параметрів мовних сигналів у завданні розпізнавання мови. Вказано недоліки сучасних статистичних моделей, з урахуванням яких запропоновано багаторівневу нечітку когнітивну модель FCAS. Ядро FCAS являє собою мережу фонетичних процесорів, що враховують вагу мовного сегмента і діють на ознаковому, фонемному і словесному рівнях. Розглянуто динаміку й алгоритми функціонування FCAS. Наведено результати експериментів із програмною реалізацією FCAS на тестовому словникові.

Ключові слова: *мовний сигнал, когнітивне моделювання, фонема, елементарний фонетичний процесор, FCAS.*

The paper deals with an actual issue of speech signal parameters processing in context of speech recognition task. Starting with the shortcomings of state-of-the-art statistical models, pointed out in the paper, multilevel fuzzy cognitive model FCAS is proposed. The FCAS kernel is a network of elementary phonetic processors, processing the weights of speech segments and functioning at feature, phoneme and word levels. Model dynamics and algorithms are given. The results of experiments with FCAS program implementation on a test vocabulary are presented.

Key words: *speech signal, cognitive modeling, phoneme, elementary phonetic processor, FCAS.*

Постановка проблемы. Проблема построения эффективных командных систем голосового управления не перестает быть актуальной в сфере информационных технологий на протяжении нескольких десятилетий. На сегодняшний день разработано множество подходов, моделей и методов автоматического распознавания речи, но на практике они не отличаются необходимой точностью. Значительный прогресс был достигнут за последние два года благодаря технологиям GoogleVoice и Apple Siri [1], но успехи этих решений обусловлены применением облачных вычислений, а не разработкой принципиально новых и эффективных моделей и алгоритмов. Рынок русскоязычных программ представлен единичными разработками, демонстрирующими посредственные результаты даже в условиях отсутствия шума.

Анализ последних исследований и публикаций. Архитектура современных систем включает два основных модуля – модуль параметризации сигнала (front-end), производящий цифровую обработку речевого сигнала (РС) и формирующий последовательность векторов признаков (ВП) (некоторое компактное описание сигнала), и модуль постобработки сигнала (back-end) [2], выполняющий распознавание слов на основе полученных ВП и закона условных вероятностей Байеса. Такие системы сначала обучаются на многочасовых коллекциях речевых данных, и затем, на этапе распознавания, производят сопоставление входных образов с ранее вводившимися по обученным моделям.

В компьютерном распознавании речи часто делается акцент на учет особенностей слуховой и голосовой систем человека [2-8]. На этапе параметризации стандартом является применение метода MFCC (мел-частотных кепстральных коэффициентов) [2]. Данный метод позволяет получить компактное описание спектра сигнала с учетом свойственной человеку логарифмической частотной шкалы мел. Тем не менее, статис-