Список использованных источников:

1. Титенко С.В. Автоматизация построения тестовых заданий в системах дистанционного обучения на основе понятийно-тезисной модели. [Электронный ресурс]. Режим доступа: https://cyberleninka.ru/article/v/avtomatizatsii-postroeniya-testovyh-zadaniy-v-sistemah-distantsionnogo-obucheniya-na-osnove-ponyatiyno-tezisnoy-modeli.

2. Зорин Ю.А. Автоматизация построения многовариантных тестовых заданий на основе деревьев И/ИЛИ. Автореферат. Томск-2015. [Электронный ресурс]. Режим доступа: http://tekhnosfera.com/view/602708/a#?page=1.

3. Черных Т.А. Автоматизация процесса формирования экзаменационных билетов с использованием LATEX [Электронный ресурс] / Черных Т. А., Полищук Ю. В. // Университетский комплекс как региональный центр образования, науки и культуры: материалы Всерос. науч.-метод. конф., 29-31 янв. 2014 г., Оренбург / М-во образования и науки Рос. Федерации, Федер. гос. бюджет. образоват. учреждение высш. проф. образования "Оренбург. гос. ун-т". - Оренбург,2014. - . - С. 2681-2685. Режим доступа: http://elib.osu.ru/bitstream/123456789/175/1/2681-2685.pdf.

4. Шендерович, В. А. Использование штрих-кодов для автоматизации работы с тестовыми заданиями / В. А. Шендерович // Компьютерные системы и сети: материалы 54-й научной конференции аспирантов, магистрантов и студентов, Минск, 23 – 27 апреля 2018 г. / Белорусский государственный университет информатики и радиоэлектроники. – Минск, 2018. – С. 124 – 125.

ВЫЯВЛЕНИЕ КЛЮЧЕВЫХ ПРИЗНАКОВ ПРИ ИНДЕКСИРОВАНИИ ТЕКСТОВЫХ ДОКУМЕНТОВ

Шичков Д.В.

Белорусский государственный университет информатики и радиоэлектроники
г. Минск, Республика Беларусь

Куликов С.С. – к.т.н.,доцент

Предлагается метод выделения ключевых признаков при построении поисковых образов текстовых документов. Метод основан на использовании интеллектуальных алгоритмов статистического анализа документов, в которых учитываются характеристики не только самих текстов, но и знания о предметной области в виде тематических корпусов текстов и сформированных на их основе лингвистических словарей. Метод может быть использован при индексировании текстовых документов из различных источников (Интернет, локальная сеть, жесткие диски отдельных пользователей).

Индексирование текстовых документов обеспечивает не только быстрый, но и качественный поиск. В процессе индексирования происходит присваивание документу структуры, называемой поисковым образом, которая описывается в терминах информационно-поискового языка. Различают два вида индексирования [1]: координатное и посткоординатное (или классификационное индексирование). При первом подходе поисковый образ документа описывается дескрипторами (ключевыми словами, семантическими признаками или понятиями определенных онтологий) в некоторой системе координат [1, 2], а при втором – документ относится к одному или нескольким заранее сформированным классам, индекс которых выступает в роли поискового образа соответствующего текста [1]. В большинстве существующих информационно-поисковых систем используется координатное индексирование, выполняемое полностью в автоматическом режиме [3].

Можно определить следующие этапы при формировании поисковых образов документов [4]:

- сбор файлов, предназначенных для индексирования;
- выделение из собранной коллекции документов текстового содержимого с максимально возможной степенью сохранения исходной разметки и форматирования;
- лингвистическая обработка текста;
- непосредственная процедура индексирования с сохранением результатов в базе данных информационно-поисковой системы.

В зависимости от типа информационного источника, из которого происходит формирование списка документов, предназначенных для индексирования, используются различные стратегии и технологии. Например, если источником выступает локальная сеть или жесткий диск компьютера пользователя, то сбор файлов можно выполнить простым сканированием интересующих объектов (общедоступных папок на компьютерах локальной сети, жестких дисках).

Основными проблемами при анализе ресурсов сети Интернет являются: стремительный рост объемов информации и ее быстрое изменение, динамическая генерация ресурсов путем выполнения инструкций на удаленном. С учетом этого, современными системами всего обработано порядка 60% доступных ресурсов Интернета без учета информации, которая хранится в базах данных и недоступна для индексирования («скрытый» веб). Для решения этих проблем создаются метапоисковые системы, которые осуществляют поиск путем обращения к индексным структурам других поисковых систем [5].

Трудность выделения текстового содержимого из загруженных файлов связана в первую очередь с тем, что документы имеют различные форматы представления (например, html, doc, pdf и т.д.). Таким образом, для перевода документа в формат plain text («чистый» текст) необходимо произвести процедуру конвертации. Вторая проблема, возникающая на данном этапе, – определение кодировки документов. Она в особенности характерна для текстов на русском и белорусском языках.

Список использованных источников:

1. Индексирование // Большая научная библиотека [Электронный ресурс]. – Режим доступа: http://bse.sci-lib.com/article054017.html. – Дата доступа: 05.03.2019.
2. Новый систематизированный толковый словарь // Государственная публичная научно-техническая библиотека России [Электронный ресурс]. – 1995. – Режим доступа: http://www.gpntb.ru/win/book/. – Дата доступа: 05.03.2019.
3. Майковский, В.В. Обзор подходов и методов индексирования в информационно-поисковых системах / В.В. Майковский // Сб. тр. Всерос. науч. школы-семинара молодых ученых, аспирантов и студентов «Интеллектуализация информационного поиска, скантехнологии и электронные библиотеки». – Таганрог: Изд-во ТТИ ЮФУ, 2010. – С. 77–78.
4. Manning, C. Introduction to Information Retrieval / C. Manning, P. Ragha-van, H. Schütze. – 1 edition. – Cambridge University Press, 2008. – 496 p.
5. Захаров, В.П. Информационные системы (документальный поиск): учеб. пособие / В.П. Захаров. – СПб.: Изд-во СПбГУ, 2002. – 188 с.

КОМПИЛЯТОР ЯЗЫКА ОБЕРОН, РЕАЛИЗУЮЩИЙ ПАРАДИГМУ РАСШИРЯЕМОГО ПРОГРАММИРОВАНИЯ

Шулицкий Д.С.

Белорусский государственный университет информатики и радиоэлектроники
г. Минск, Республика Беларусь

Сурков К.А. – ст. преподаватель

В настоящее время перед программистами зачастую встаёт задача расширения функционала существующего кода. Зачастую эта задача решается при помощи механизмов наследования и полиморфизма в ООП. Однако это не позволяет модифицировать уже существующие объекты и переопределять статические функции.

Самый простой способ модификации существующей программы – изменение её исходного кода. Однако, это не всегда допустимо. Например, может быть недопустима модификация библиотечного модуля. Модификация существующего кода приводит к необходимости повторного тестирования модулей.

В объектно-ориентированном программировании названные выше проблемы решаются при помощи наследования и полиморфизма. Однако это не позволяет модифицировать данные в существующих объектах. Для модификации кода необходимо создавать объекты классов, что зачастую не удобно.

Эти проблемы можно решить при помощи расширяемого программирования. Расширяемое программирование состоит из расширения кода и расширения данных.

Расширение кода означает, что новый модуль изменяет работу существующих модулей и реализуется при помощи переопределения существующих процедур, которое похоже на переопределение методов в ООП. Существует несколько способов реализовать переопределение кода:

1) замена адреса вызываемой процедуры во всех точках её вызова;
2) вызов процедуры через процедурную переменную (вместо прямого вызова процедуры);
3) генерация компилятором процедуры-переходника, которая выполнит вызов через процедурную переменную;
4) замена кода процедуры по месту.

Расширение данных означает добавление новых полей в существующую запись. Для расширения данных можно вручную ассоциировать оригинальные объекты с объектами-дополнениями, но следствием этого являются следующие проблемы:

- высокая трудоёмкость, так как необходимо поддерживать таблицу соответствий;
- снижение надёжности, так как необходимо уничтожать объекты-дополнения при уничтожении оригинальных объектов, а ошибки при реализации могут привести к утечкам памяти даже при наличии сборщика мусора;
- падение производительности при поиске объекта-дополнения для объекта-оригинала.
- необходимость синхронизации доступа к таблице соответствия в многопоточных приложениях.