УДК 004.021:004.738.52

АЛГОРИТМ ЭКСТРАКЦИИ ЗНАЧИМОЙ ИНФОРМАЦИИ ИЗ СТРАНИЦ WEB-САЙТОВ

А.П. ШОРКИН

Белорусский государственный университет информатики и радиоэлектроники
П.Бровки, 6, Минск, 220013, Беларусь

Поступила в редакцию 22 октября 2012

Описаны разработанные алгоритмические подходы к выделению из web-страниц отдельных семантических частей: содержание и служебно-навигационная часть. Подходы базируются на механизме определения единых элементов на разных страницах одного сайта. Главной задачей исследования является улучшение качества информационного поиска посредством исключения из поискового массива web-страниц служебно-навигационной части. Реализованный эксперимент по анализу качества информационного поиска в web на основе тестовых данных с использованием реализованного алгоритма привел к определенному улучшению средней оценки точности поиска. В статье приведен детализированный анализ результатов информационного поиска с использованием описанного алгоритма.

Ключевые слова: алгоритм экстракции, информация, web-сайт.

Введение

На сегодняшний день большая часть сайтов Интернет имеют достаточно сложные стилевые формы и разрабатываются в средах динамической генерации контента html-страниц [1]. Стилевое оформление web-сайтов подразумевает общий дизайн всех страниц, механизмы навигации по страницам сайта, фирменный знак (лого), контакты и прочую информацию, включая рекламные блоки (баннеры).

Некоторая область web-страницы, отвечающая за оформление (назовем ее служебно-навигационной частью, или фреймингом), в большинстве случаев существенно отличается от семантической части страницы (содержание). Этот факт необходимо учесть в проектировании и разработке механизмов автоматизированной сборки и процессинга индексации текстов с целью информационного поиска.

Задача исследования – реализация методов автоматического разделения web-страниц на служебно-навигационную и семантическо-содержательную части. Гипотеза представляет собой предположение полезности исключения и уменьшения веса служебной части html-страниц из индекса поисковой системы с целью реализации задачи повышения качества информационного поиска.

В следующем разделе описаны существующие методики анализа структуры оформления страниц сайтов. Далее приведен предложенный в работе алгоритм, проанализирована методика оценки качества работы алгоритма.

Существующие подходы

Задача анализа и разбора шаблонов автоматически создаваемых web-страниц часто привлекает внимание разработчиков и исследователей в последнее время. Определение структурной части информации может быть использовано для разных целей: процессинга (обработки) документов [1], разнообразных задач информационного поиска, мониторинга вариативных изменений и организации кэша html-страниц [2].

Работа [3] объясняет как сложившаяся практика стилевого оформления web-сайтов средствами динамического создания сайтов нарушает основные предположения лежащие в основе большинства сегодняшних поисковых машин в сети Интернет. Среди таких базовых предположений следующие [4]:

1) принцип релевантности (качественного соответствия) web-ссылок (Relevant Linkage Principle) – гипертекстовые ссылки, созданные авторами материала на странице, ссылаются на «авторитетные» и качественные документы и ресурсы по аналогичной тематике;

2) сайты, организовавшие обмен ссылками друг на друга, являются тематически схожими (Topical Unity Principle);

3) текст, окружающий html-ссылку, соответствует тематике материала web-сайта, на который проставлена ссылка (Lexical Affinity Principle).

В отличие от [4], мы анализируем, в первую очередь, семантическое наполнение html-страниц и классический механизм поиска страниц документов, не учитывая служебную структуры web-ссылок. Тогда представляются возможными выявление и оценка качества процесса информационного поиска с использованием свободно доступных тестовых данных (например, «Тестовые данные информационно-поисковой системы Yandex.ru», свободно распространяемой компанией Яндекс).

Используемые методики анализа шаблона html-страниц делятся на

1) базирующиеся на разборе dom-дерева отдельной html-страницы [1, 2];

2) базирующиеся на определении регулярных (т. е. повторяющихся) участков web-страницы с одного сайта [4].

Частым способом построения шаблона web-страниц является организация служебно-навигационной части по обоим краям web-страницы (контентно-содержательная часть при этом располагается в центре). На уровне программного html-кода страницы конструируются в виде таблицы (возможно, с невидимыми пользователю границами) или нескольких вложенных таблиц <table> (тэг html), при этом отдельные элементы web-страницы разнесены в различные ячейки.

Методики, базирующиеся на разборе и анализе dom-дерева, реализуются на основе анализа шаблонизированной структуры web-страниц и выдвигают предположение об экстракции контентной части web-страницы на основе базовых эвристических предположений о некоторых «регулярных» способах оформления документов. Данные методики как правило комбинируются с методами, основанными на выделении регулярных участков и фрагментов html-страниц одного web-сайта [1].

Методики, базирующиеся на выделении повторяющихся участков страниц одного сайта реализуются на основании предположения повторяемости у страниц одного сайта элементов стилевого оформления и одновременно различия контентно-содержательной части.

Стоит отметить также, что в работе [4] регулярные фрагменты анализируются для различных web-сайтов для детерминирования кластерных сообществ web-ресурсов, связанных между собой.

Предлагаемый алгоритм

В работе использован подход, базирующийся на определении повторяющихся областей в множестве страниц одного сайта. Предложен алгоритм для выполнения описанной задачи.

На вход программному алгоритму поступает некоторая директория с html-файлами, соответствующими страницам одного анализируемого сайта. Алгоритм выполняет обработку данных файлов и выполняет определение и маркировку в них участков программного кода, которые считаются служебно-навигационными частями. С различными настройками алгоритмом производится или удаление служебно-навигационных элементов из html-файлов, или маркировка (выделение) этих частей специальными html-тегами.

Структура алгоритма разделяется на несколько этапов:

1) деление html-файлов на базовые, неделимые при дальнейшем сравнении символьно-кодовые последовательности– фрагментеры;

2) выполнение поиска файловых подмножеств (кластеров), с повторяющимся набором цепочек фрагментеров (кластеры определяют некоторое подмножество страниц с единой слу-

жебно-навигационной частью). Набор цепочек фрагментеров представляется одной или несколькими цепочками (подряд расположенных последовательностей) фрагментеров. Желательно, чтобы выполнялись следующие условия: множество цепочек было как можно больше (длиннее) и находилось в большом количестве html-файлов;

3) удаление (или маркировка) определенной последовательности цепочек фрагментеров из всех файлов данного кластера;

4) в случае не покрытия кластером всего множества файлов, необходимо заново выполнить шаги 2–4 для оставшегося множества документов;

5) в случае наличия остатка нетронутых файлов (из html-кода которых не было удалено или промаркировано ни одного фрагментера), то в этих файлах выполняется новый поиск и удаление (маркировка) служебно-навигационных частей – цепочек фрагментеров – из прочих кластеров.

Шаг 1. *Фрагментеризация html-файла.* Файл делится на базовые неделимые при сравнении и маркировке оформления кодово-символьных последовательностей – фрагментеры. Фрагментер определим как

1) конкретный тег спецификации html;

2) текст, находящийся между тегами.

Пробелы, находящиеся между html-тегом и текстом необходимо игнорировать.

Шаг 2. *Нахождение кластера с единой служебно-навигационной частью.* Каждый экземпляр файлового массива реализуется множеством *цепочек фрагментеров*. Цепочка фрагментеров – это непрерывная последовательность фрагментеров, идущих подряд, с длиной min_fragmenters_chain, где min_fragmenters_chain – параметр, определяющий минимальную длину обрабатываемого (вырезаемого и маркируемого) куска файла в фрагментерах (в реализации был использован min_fragmenters_chain=6). Для всех цепочек фрагментеров с длиной min_fragmenters_chain производится расчет значения хэш-функции с использованием алгоритма CRC32.

Таким образом, производится отображение каждого файла в определенное множество чисел – хэш-значений цепочек фрагментеров с длинами min_fragmenters_chain. Для всех цепочек фрагментеров длины min_fragmenters_chain производится запоминание размера (длины) оригинального текста в байтах.

Построение кластера выполняется следующим образом.

1. *Нахождение в кластере начальной пары документов.* Производится перебор всех пар html-документов с веб-сайта, после чего для каждой найденной пары выполняется определение длины в байтах совпадающих цепочек фрагментеров.

В случае слишком большого совпадения между файлами (задаваемый параметр для файлов – более 70% от всей длины), эти файлы определяются как дубли и не могут быть использованы с целью создания первой пары в обрабатываемом кластере.

Для пары файлов с максимальным совпадением производится добавление в кластер. Алгоритмом производится пересечение этих файлов (т.е. набор хэш-кодов цепочек фрагментеров), определение служебно-навигационной части кластера, и запоминание для последующей обработки. Выполняется определение порога min_nav_length (настраиваемый параметр) как равное 80% от длины в байтах служебно-навигационной части кластера. В последующих вычислениях при выполнении построения конкретного кластера длина служебно-навигационной части подлежит уменьшению, но не более порогового значения min_nav_length.

2. *Выполнение поиска документов для дальнейшего добавления в кластер.* Реализуется поиск документа, максимально пересекающегося со служебно-навигационной частью кластера. Файл добавляется в конкретный кластер в случае выполнения условия: длина в байтах совпадения со служебно-навигационной частью кластера не меньше значения min_nav_length.

Служебно-навигационная часть кластера делается эквивалентной пересечению с вновь добавленным кластером. Производится повторение данного шага до тех пор, пока существуют подходящие необработанные файлы для добавления в данный кластер.

Шаг 3. *Извлечение служебно-навигационных элементов из множества документов кластера.* В случае присутствия в построенном кластере не менее 4 документов, данный кластер определяется как корректно построенный, после чего из всех документов кластера удаляется служебно-навигационная часть кластера (т.е. единая часть всех html-файлов данного кластера).

Шаг 4. *Выполнение поиска других кластеров.* В случае, если после построения кластера в массиве необработанных файлов присутствует не менее 4 html-документов, повторяются шаги 2–4 для всех оставшихся документов. Если построение кластера выполнить не удается, производится понижение порогового значения min_nav_length с шагом 20 % (до 40 %) и повторяются шаги 2–4 с новым пороговым значением min_nav_length.

Оценка реализации алгоритма

Для оценки эффективности работы реализованного алгоритма производился последовательный анализ поисковых результатов алгоритма в разрезе каждого кластера и каждого документа. Были проанализированы массивы документов с использованием html-интерпретатора, прошедшие обработку алгоритмом, с целью определения корректности отнесения вырезанного/маркированного фрагмента к оформлению страницы, а не к контентной части.

Оценка реализованного алгоритма производилась на коллекции сайтов, предоставленных компанией Яндекс (http://yandex.ru). Для тестирования различных алгоритмов информационного поиска. Указанная коллекция состоит из более чем 35 тысяч html-страниц, закачанных с разнообразных сайтов. Коллекция включает большое количество вариаций стилевого и дизайнерского оформления web-сайтов, так как данные сайты разрабатывались различными коллективами и индивидуальными специалистами с использованием разнообразных технологий web-разработки, языков программирования и СУБД.

Оценка производилась путем последовательного визуального анализа обработанных (вырезанных/маркированных) элементов html-страниц в стандартном web-браузере с использованием подсветки обработанных частей станиц. Элементы html-страницы, отнесенные алгоритмом к «повторяющемуся оформлению», маркировались цветом.

Было последовательно проанализировано 60 случайно выбранных из коллекции web-сайтов и до 5 результатов обработки алгоритмом маркировки служебно-навигационной части. Для всех html-документов проанализированного сайта была произведена оценка реализации алгоритма от «2» (служебно-навигационная часть не промаркирована, или алгоритм неверно пометил содержательную часть документа) до «5» (служебно-навигационная часть определена корректно). Баллы «3» и «4» определялись в промежуточных вариантах в случае обнаружения незначительных неточностей. Были зафиксированы следующие оценки: оценка 2 для 4-х проанализированных сайтов; оценка 3 – для 3-х, оценка 4 – для 10-ти, оценка 5 – для 43-х проанализированных сайтов. Средний балл составляет 4,53 балла.

Для сравенния результатов алгоритма с уже существующим подходом, была проведена разработка программной модели метода, базирующиегося на разборе dom-дерева отдельной html-страницы по рекомендациям [1]. Результат выполнения поиска при помощи данного алгоритма по той же выборке сайтов составляет 4,42 балла, что показывает более высокую результативность разработанного автором метода на данной коллекции документов

Заключение

Представлен алгоритм нахождения в структуре web-страниц служебно-навигационной части и отделения ее от контентно-содержательной части. Оценка результатов выполнения алгоритма показала, что алгоритм в целом достигает поставленные цели и показывает более высокие результаты по сравнению с имеющимся алгоритмом. В ходе анализа выявлены вероятность содержания в оформлении сайта данных, важных для результатов поиска, а также случаи, когда невозможны выявления и маркировки нестандартных элементов стилевого оформление, базируясь лишь на анализе повторяющихся частей html-страниц. Применение алгоритмов, подобных реализованному, может показать эффективность для специализированных задач информационного поиска в случаях, когда изначально детерминирована вероятность снижения качества поиска вследствие зашумления служебно-навигационными элементами оформления html-страниц конкретного сайта. К примеру, данными задачами могут быть: регулярный мониторинг определенного набора web-сайтов; выполнение информационного поиска на сайтах стандартной структуры: корпоративных сайтах, форумах, web-библиотеках.

ALGORITHM FOR MINING OF CORE WEBSITES PARTS FOR INFORMATIONAL SEARCH EFFICIENCY

A.P. SHORKIN

Abstract

Algorithm for automatic dividing of web page into 2 parts: service-navigational and contend parts is described. The method is based on the mining of repeatable elements in html-pages from same website. Main theory is that the quality of information search can be improved by tagging / deleting navigational elements of html pages. Developed method successfully mine service and content parts from html-pages. On the other hand, deleting of service part does not guarantee perfect improvement of web information search quality.

Список литературы

1. *Ландэ Д.В.* Основы интеграции информационных потоков. М., 2006.
2. *Barfourosh A., Nezhad H., Anderson M. et. al*. Information Retrieval on the World Wide Web and Active Logic: A Survey and Problem Definition. Michigan, 2002
3. *Sebastiani F.* // ACM Computing Surveys. 2002. Vol. 1. P. 1–47.
4. *Liao C., Alpha S., Dixon P.* // Proceedings of Australian Data Mining Conference. Canberra, 2003.