Компьютерные инструменты в образовании, 2015
№ 4: 9–16
УДК: 621.3
http://ipo.spb.ru/journal

АЛГОРИТМ «ЛОЖНОГО ПУТИ» ДЛЯ СИСТЕМ ПОИСКА КОНТЕНТА

Жигулин Андрей Юрьевич

Аннотация

Разработка систем поиска основана на успехе поиска контента. Основной проблемой в такой разработке является высокая вероятность неправильного результата при поиске в случае нахождения ошибки в системе.

В статье рассматривается алгоритм, увеличивающий шанс успешного нахождения пользователем интересующего его контента в системе поиска, на основании пользовательского взаимодействия. Алгоритм описывает процесс получения данных системой, преобразования данных в ранжируемые сущности, а также определения отображаемых результатов. Также в статье приведены примеры основных алгоритмов, вошедших в основу алгоритма «ложного пути», их особенности и проблемы. Кроме того, в статье предоставлены дальнейшие пути улучшения алгоритма.

Ключевые слова: *алгоритм, «ложный путь», ранжируемые сущности, системы поиска контента.*

1. ВВЕДЕНИЕ

При внедрении системы поиска одной из главных проблем является большая вероятность ошибки в представлении добавляемого контента. Это может быть связано с человеческим фактором (непонимание контента при добавлении, опечатки, неправильная информация о контенте), а также с возможностями системы поиска (малое количество параметров контента при добавлении в систему, системные ошибки при добавлении в базу). Несмотря на большое количество различных алгоритмов поиска контента в системах поиска, данная проблема считается допустимой погрешностью при поиске, из–за чего часть контента найти становится практически невозможно.

Кроме того, практически все современные системы поиска контента выводят в отдельной области контент, каким–либо образом связанный с текущим рассматриваемым контентом (например сопутствующие товары при покупке). Такой поиск учитывает действия пользователя и выводит наиболее популярный контент, связанный с рассматриваемым. Тем не менее, ошибочно заполненный контент имеет малую вероятность быть результатом какого–либо действия пользователя, из–за чего становится невозможным и увеличение его популярности как сопутствующего контента.

Наконец, для поддержания актуальности контента необходима возможность легкой доступности такого контента, что в случае ошибочного заполнения не представляется возможным.

Таким образом, нерешенными остаются следующие задачи:

– учет данных ошибочно заполненного контента при поиске,
– отображение логически связанного с действиями пользователя ошибочно заполненного контента,
– актуализация ошибочно заполненного контента.

2. ОПИСАНИЕ РЕАЛИЗАЦИИ

В данной статье рассказывается об алгоритме, предназначенном для решения описанных задач путем представления накопленных системой поиска контента знаний в виде структурируемых, ранжируемых и быстро получаемых данных и дальнейшим получением сохраненных данных при нахождении совпадений в базе данных для увеличения шанса успешности поиска интересующего пользователей контента.

При рассмотрении систем поиска контента, прежде всего, необходимо выделить основные алгоритмы, использующиеся в различных известных системах поиска.

Для любой страницы в сети Интернет существует такая метрика, как индекс цитирования. Индекс цитирования (ИЦ) — показатель поисковой системы, вычисляемый на основе числа ссылок на данный ресурс с других ресурсов сети Интернет. В простейшей разновидности индекса цитирования учитывается только количество ссылок на ресурс. Тематический индекс цитирования (тИЦ) учитывает также тематику ссылающихся на ресурс сайтов, а взвешенный индекс цитирования — популярность ссылающихся сайтов (также в большинстве случаев вычисляемую на основе индекса цитирования).

Первоначально, до появления оптимизаторов сайтов, индекс цитирования реально отражал популярность соответствующего ресурса в интернете.

Первой крупной поисковой системой, начавшей активно использовать индекс цитирования, стала Google (алгоритм PageRank [1]).

PageRank — алгоритм вычисления веса страницы путем подсчета важности ссылок на нее. Данный алгоритм учитывает количество переходов на страницы и за несколько лет стал основой для аналогичных поисковых алгоритмов у многих других компаний [2]. В том числе, поисковая система Яндекс, запущенная в 1997 году, опиралась на аналогичный механизм, который получит название «ТИЦ», или «тематический индекс цитирования» [3].

Дальнейшие улучшения алгоритма PageRank проводились через внедрение технологии «PageRank sculpting» [4], добавляющей дополнительный параметр «nofollow» на определенные страницы для сохранения веса определенных страниц и дальнейшей передачи данного веса на целевые страницы. Тем не менее, данная технология по признанию компании перестала работать в 2009 году по причине равномерного распределения PageRank между исходящими ссылками после установки атрибута «nofollow».

Данный алгоритм работает также в расширении Google Toolbar [5], предоставляемом для браузеров Internet Explorer и Mozilla Firefox.

Несмотря на удобство реализации, алгоритм PageRank имел один крупный недостаток — при централизованном массовом обращении к определенным страницам (SPAM [6]) алгоритм терял свою актуальность.

Для предотвращения данной проблемы стал учитываться такой фактор, как время, проведенное пользователем на сайте. Таким образом, был разработан алгоритм BrowseRank [7, 8]. Суть алгоритма заключается также в построении графа, но, в отличие от алгоритма PageRank, его ребрами являются не ссылки, а непосредственно переходы по ссылкам.

На основе алгоритма BrowseRank был разработан данный алгоритм. Основное отличие данного алгоритма от алгоритма BrowseRank заключается в том, что ранжируемые сущности имеют свой идентификатор значимости, а также основной целью алгоритма является не анализ перехода по ссылкам, а анализ цепочек перехода по ссылкам, то есть поиск того пути в графе, который был бы затруднительным для пользователя вследствие каких–либо ошибок в системе. Именно получение такого пути для последующих запросов в системе позволит пользователю увеличить шанс нахождения того контента, для нахождения которого потребуется значительное количество времени. Результатом долгого поиска с большой долей вероятности может стать отказ от поиска в целом и переход на другую систему поиска, что, безусловно, не является желательным для компании, предоставляющей услуги системы поиска.

3. ОПИСАНИЕ АЛГОРИТМА

Когда пользователь использует систему поиска, система записывает все его действия (выбор фильтров, переходы между страницами контента, комментирование контента, выставление рейтингов) и ищет совпадения с данными, имеющимися в базе данных. Самые актуальные совпадения выводятся в специально отведенной области на странице системы поиска, предлагая пользователю тот контент, который мог бы его заинтересовать, но не отобразился в системе поиска вследствие ошибки представления. В случае выбора такого контента, пользователю предлагается оценить алгоритм, и, в зависимости от средней оценки и даты оценивания, путь приобретает больший либо меньший вес при следующем выборе данных для предоставления пользователю алгоритмом. Необходимость сбора данных на основании пользовательского взаимодействия описана в [9].

Формулировка алгоритма выглядит следующим образом:

1. Получение последнего действия пользователя (переход между страницами контента, выставление рейтинга контенту, комментирование контента и т. д.).
2. Преобразование действия пользователя в ранжируемую сущность.
3. Добавление действия пользователя в конец цепочки действий пользователя.
4. Добавление цепочки действий пользователя в базу знаний D (исключая добавление дубликатов, но учитывая актуальность данных при их совпадении). В базе знаний D хранятся элементы типа <key, value>, где key — это ранжируемая сущность, а value — объект, показывающий актуальность ее использования (количество использований, дата последнего использования, успешность, а также id объекта, на котором инструкция была признана успешной). Данный алгоритм будет содержать в базе ранжируемые сущности со сложными ключами, так как действия пользователя рассматриваются как продолжение цепочки из предыдущих действий. Поэтому, ключ имеет вид «$C_1..C_j$», где j — это количество распознанных действий в цепочке действий пользователя. При совершении следующего действия система преобразует цепочку так, чтобы добавленное действие находилось на определенной позиции (выбор сортировки действий остается на усмотрение разработчика, в том числе можно не сортировать действия перед записью в базу знаний вообще, но тогда количество ранжируемых сущностей в базе значительно возрастет, увеличив время поиска контента, подходящего для следующих цепочек действий). Изначально каждая ранжируемая сущность заносится в базу со значениями:

1 — количество использований;
0 — успешность;
текущая дата занесения в базу знаний – дата последнего использования;
Null — id объекта, так как пользователь еще не указал успешность действия.

В дальнейшем успешность помогает проверять актуальность ранжируемых сущностей и последующее удаление неактуальных сущностей. Такое детальное, но упрощенное в терминах хранения представление элементов позволяет экономить ресурсы и оставлять их для более важных задач (оптимизация данных и ресурсов рассмотрена в [10]).

5. Стадия «ложного выбора» — процесс проверки каждого элемента базы знаний D на актуальность для системы. Если в базе знаний D имеется набор ранжируемых сущностей, схожих по ключу с действиями пользователя, то система их получает и выводит в качестве списка результатов в отдельной области результатов. При правильной сортировке последовательности действий поиск происходит по совпадению подпоследовательностей цепочек, и, чем длиннее совпадение цепочек и больше значение успешности ранжируемой сущности, тем выше в списке будет результат. Способы проверки с оптимальным использованием ресурсов, а также различные алгоритмы выбора получаемых данных описаны в [11].
6. (Опциональное) уничтожение неактуальных цепочек.

Более наглядно алгоритм описан на рис. 1.

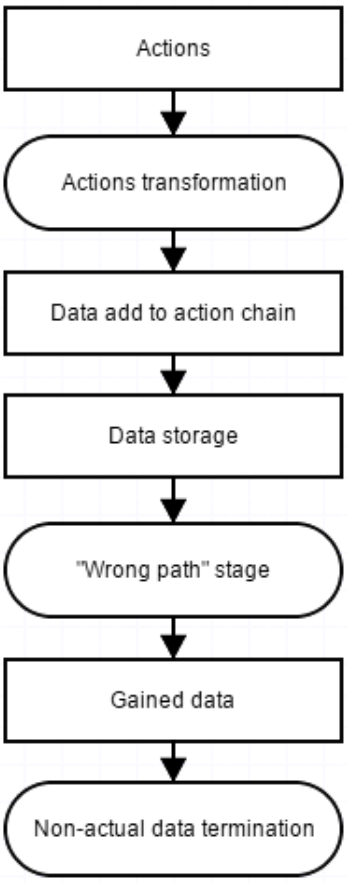

Рис. 1. Последовательность шагов алгоритма

Успешность корректируется пользователями в том случае, если пользователь перешел на страницу объекта, предложенного в области результатов, а затем подтвердил правильность предложенного результата (например совершил покупку с этой страницы и выставил во всплывающем окне подтверждение успешности выбора). Также успешность может проставляться автоматически при совершении покупки или скачивания контента.

Архитектура внедряемой системы разделена на несколько составляющих:

– интерфейс, определяющий основные возможные действия пользователя, а также ранжированные сущности, определяющие все возможные действия (организация данного интерфейса и сущностей, ему удовлетворяющих, требует 90% всего времени на интегрирование алгоритма ложного пути в систему и будет уникальна в каждой системе поиска);
– «Recorder», отвечающий за запись действий пользователя;
– «Receiver», отвечающий за актуализацию путей при успешном завершении алгоритма и удовлетворенности пользователя алгоритмом.

Взаимодействие описанных компонент продемонстрировано на рис. 2.

В ходе проведенных исследований были получены следующие результаты:

1. Сформулирован и построен алгоритм ложного пути для систем поиска контента.
2. Алгоритм внедрен в интернет–магазин и в систему поиска приложений с базой данных, насчитывающей более 9000 пользователей.
3. Внедрены метрики подсчета активности пользователей, успешности их покупок, логирование их действий, совершающие сбор информации в формате 24/7.

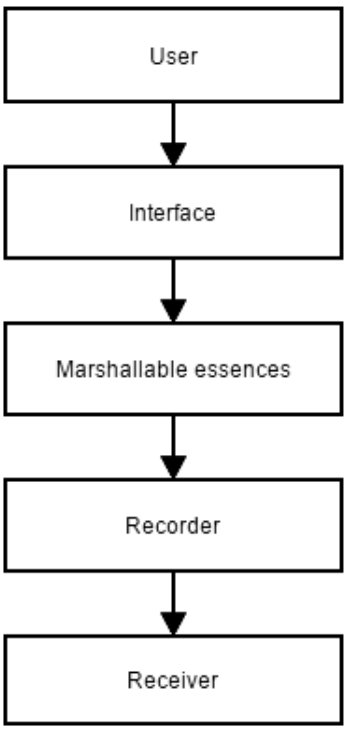

Рис. 2. Организация компонент системы

4. ЭКСПЕРИМЕНТАЛЬНАЯ ПРОВЕРКА

Для экспериментальной проверки алгоритм ложного пути был запущен для целевой аудитории, производящей установку приложений из системы поиска приложений и совершающей покупки в интернет–магазине. Пользователи были разделены по активности в соответствующих системах на 2 целевые группы с примерно одинаковой средней активностью за 6 месяцев. Группы были сформированы по ip–адресам. Таким образом, было сформировано 4 группы — 2 группы с примерно одинаковой активностью в интернет–магазине (для первой была включена система с алгоритмом ложного пути, для второй осталась прежняя система) и 2 группы с примерно одинаковой активностью в системе поиска приложений (с аналогичным разбиением).

Система поиска приложений включала в себя на момент интеграции с алгоритмом 286 приложений, интернет–магазин включал в себя 2341 уникальный товар. Кроме того, для дополнительной проверки в интернет–магазин было занесено 100 товаров с изначально неверными данными, усложняющими поиск. Для каждой группы в течение периода протяженностью 4 месяца производился подсчет фактических переходов на страницы систем поиска, логирование успешных покупок, время сессии.

В результате экспериментальной проверки в центре приложений количество «мертвых» страниц с низким количеством запросов уменьшилось в 5 раз, время достижения результата уменьшилось как в общем случае, так и в случае с самыми популярными приложениями. В интернет–магазине количество «мертвых» страниц уменьшилось вдвое, а количество запросов к страницам с изначально неверными данными возросло почти в 17 раз. Время достижения результата в интернет–магазине также уменьшилось как в общем случае, так и в случае с самыми популярными товарами. Более подробно результаты представлены в таблице 1.

Таблица 1. Результаты экспериментальной проверки алгоритма ложного пути в системе поиска приложений и в интернет–магазине

	До внедрения алгоритма	После внедрения алгоритма
Количество страниц приложений	286	286
Количество страниц приложений с числом запросов менее 2 за период	52	11
Среднее время от перехода в поиск до установки приложения	55 s	40 s
Среднее время от перехода в поиск до установки приложения для страниц с числом запросов более 100 за период	30 s	26 s
Количество страниц с товарами	2341	2341
Количество страниц с товарами с числом запросов менее 2 за период	603	300
Количество страниц с товарами с изначально неверными данными	100	100
Количество запросов к страницам с изначально неверными данными	14	236
Среднее время от перехода в поиск до помещения товара в корзину	174 s	153 s
Среднее время от перехода в поиск до помещения товара в корзину для страниц с числом запросов более 100 за период	90 s	71 s

По результатам экспериментальной проверки видно, что алгоритм ложного пути значительно увеличивает шанс нахождения контента с низким числом обращений и, что более важно, контента с изначально неверными данными. Кроме того, алгоритм поддерживает актуальность данного контента, из–за чего среднее время успешной сессии в системе поиска уменьшается. Наконец, алгоритм поддерживает актуальность и у популярных результатов поиска, что упрощает поиск популярного контента при неправильных условиях поиска и уменьшает время успешной сессии у популярного контента.

Таким образом, алгоритм решает поставленные задачи, не ухудшая работу обычного поиска.

5. ЗАКЛЮЧЕНИЕ

Являясь перспективным направлением разработки, системы поиска контента являются неотъемлемой частью большинства сайтов крупных компаний.

Все коммерческие компании непрерывно ведут разработки алгоритмов, способных обеспечить рост продаж. Часть компаний увеличивает сферу влияния своих систем, часть — предлагает сопутствующие товары при покупке.

Тем не менее, оба этих подхода не учитывают самый важный момент: для успешной покупки необходимо найти интересующий покупателя контент.

Данный алгоритм разработан для улучшения именно этой составляющей систем поиска.

Алгоритм является независимым и не нарушает описанные выше концепции предоставления сопутствующих товаров и расширения сферы влияния, а наоборот, дополняет и раскрывает их.

Логика данного алгоритма основана на записи действий пользователя, поэтому реализация алгоритма требует грамотного использования ресурсов при разработке архитектуры системы, требующей его внедрения.

Следующим шагом в улучшении данного алгоритма является создание алгоритма чистки неактуальных цепочек, который также должен учитывать существенное количество факторов, таких как частота использования цепочки в определенный промежуток времени или популярность целевой страницы.

Также перспективным развитием является создание защиты, способной отсеивать неподходящие цепочки, в случае если злоумышленник попытается симулировать успешное завершение цепочки и, тем самым, нарушить последовательность вывода подходящих результатов.

Список литературы

1. Method for node ranking in a linked database [электронный ресурс] / URL: http://www.google.com/patents/US6285999 (дата обращения 25.05.2015).
2. Beyond PageRank: Machine Learning for Static Ranking [электронный ресурс] / URL: http://www2006.org/programme/files/xhtml/3101/p3101-Richardson.html (дата обращения 25.05.2015).
3. Что такое ТИЦ [электронный ресурс] / URL: http://help.yandex.ru/catalogue/citation-index/tic-about.xml (дата обращения 25.05.2015).
4. PageRank sculpting [электронный ресурс] / URL: https://www.mattcutts.com/blog/pagerank-sculpting/ (дата обращения 25.05.2015).
5. Google toolbar [электронный ресурс] / URL: http://www.google.com/intl/ru/toolbar/ie/index.html (дата обращения 25.05.2015).

6. Web Spam Taxonomy [электронный ресурс] / URL: http://airweb.cse.lehigh.edu/2005/gyongyi.pdf (дата обращения 25.05.2015).
7. *Yuting Lu, Bin Gao, Tie-Yan Liu, Zhiming Ma, Shuyuan He, Hang Li, Ying Zhang.* BrowseRank: Letting Web Users Vote for Page Importance. China, 2008. P. 451—458.
8. *Boldi P., Santini M., Vigna S.* Pagerank as a function of the damping factor. USA, 2005. P. 557—566.
9. *Tao Wu, Hui He, Xiqian Gu, Ying Peng.* An intelligent network user behavior analysis system based on collaborative Markov model and distributed data processing. Canada, 2013. P. 221–228.
10. *Ericson C.* Real-Time Collision Detection. USA, 2004.
11. *Knuth D.E.* The Art of Computer Programming. Vols. 1–3. UK, 1998.

"WRONG PATH" ALGORITHM FOR CONTENT SEARCH SYSTEMS

Zhigulin A. Y.

Abstract

Search systems design is based on the successful rate of content find. The key problem in that kind of design is a good probability of failure in content find if any mistake in the system occurs.

The article discusses an algorithm that increases the chances of successful content find in content search systems by analyzing user behavior. The algorithm describes the process of data gain by the system, data transfer into marshallable essences, and corresponding results determination. The article also refers to most common algorithms that serve as the basement for the «wrong path» algorithm, their problems and key features. The article discusses possible improvements of the algorithm that should be taken into account in future.

Keywords: *algorithm, «wrong path», marshallable essences, content search systems.*

Жигулин Андрей Юрьевич,
аспирант СПбГУ,
dron.subzero@gmail.com