ISSN 2306-1561
Automation and Control in Technical Systems (ACTS)
2014, No 2, pp. 60-67.
DOI: 10.12731/2306-1561-2014-2-6

Search method of tree data structures

Sapego Yuliya Sergeevna

Russian Federation, Postgraduate Student, Department of «Automated Control Systems».

Moscow Automobile & Road construction State Technical University, 125319, Russian Federation, Moscow, Leningradsky prospekt, 64. Tel.: +7 (499) 151-64-12. http://www.madi.ru

kafedra@asu.madi.ru

Abstract. In modern data retrieval systems the main problem is how to store data properly and how to reach a high speed of processing of users researching requests. It is worth noting that the main point of information retrieval systems is not only research of exact conformity of requesting information, but also the point of researching of relevant information, where the level of relevance could be determined as the level of its semantic proximity to the researching request. Thus researching requests in such a kind of systems must be based on natural language in where initial information is formulated.

Keywords: tree data structure, suffix tree, trie (prefix tree), ternary search trees.

ISSN 2306-1561
Автоматизация и управление в технических системах (АУТС)
2014. – №2. – С. 60-67.
DOI: 10.12731/2306-1561-2014-2-6

УДК 004.8

Методы поиска по древовидным структурам данных

Сапего Юлия Владимировна
Российская Федерация, аспирант кафедры «Автоматизированные системы управления».
ФГБОУ ВПО «Московский автомобильно-дорожный государственный технический университет (МАДИ)», 125319, Российская Федерация, г. Москва, Ленинградский проспект, д.64, Тел.: +7 (499) 151-64-12, http://www.madi.ru
kafedra@asu.madi.ru

Аннотация. В современных системах выборки данных основной задачей является обеспечение надёжного хранения данных, а также высокой скорости выполнения поисковых запросов пользователей. Стоит отметить, что на информационно-поисковые системы возлагается не только задача поиска на точное соответствие запрашиваемой пользователем информации, а скорее задача по поиску релевантной информации, где степень релевантности можно определить как степень её смысловой близости к поисковому запросу, а это в свою очередь ведёт к тому что поисковые запросы в такого рода системах должны быть основаны на естественном языке, т.е на том же языке, в котором сформулирована исходная информация.

Ключевые слова: поиск по древовидным структурам данных, суффиксные деревья, префиксные деревья, тернарные деревья.

1. Введение

В последнее время возросла тенденция к обработке больших объемов текстовой информации, в первую очередь это связано с развитием сети Интернет, которое привело к появлению огромного количества текстовой информации, в виде обычных текстов, HTML- и XML-документов, сообщений электронной почты (по оценкам аналитиков объем доступной информации составляет около 4 200 терабайт).

В современных системах выборки данных основной задачей является обеспечение надёжного хранения данных, а также высокой скорости выполнения поисковых запросов пользователей [1, 6 – 15]. Стоит отметить, что на информационно-поисковые системы возлагается не только задача поиска на точное соответствие запрашиваемой

пользователем информации, а скорее задача по поиску релевантной информации, где степень релевантности можно определить как степень её смысловой близости к поисковому запросу, а это в свою очередь ведёт к тому что поисковые запросы в такого рода системах должны быть основаны на естественном языке, т.е на том же языке, в котором сформулирована исходная информация.

Время поиска в структурах, представленных в виде списков, по текстовым данным составляет $O(\log n)$, при этом списки ключевых терминов должны быть отсортированы, так как именно при удовлетворении такого условия возможен бинарный поиск за логарифмическо е время. Отсортированные списки имеют ещё один недостаток: их сложно модифицировать (удалять/вставлять ключевые термины) и количество операций затрачивается на это не меньше, чем $O(n)$.

Следует тметить, что существуют структуры данных, которые не обладают выше изложенными недостатками – это древовидные структуры данных. В данной работе будут проанализированы методы поиска полнотекстовой информации, представленной в древовидной структуре.

2. Виды древовидных структур

Ниже будут рассмотрены виды древовидных структур: префиксные деревья, а так же его модификации.

2.1. Префиксные деревья (trie)

Префиксное дерево (Trie, prefix tree, digital tree, radix tree) – это структура данных для реализации словаря (ассоциативного массива), ключами в котором являются строки. Особенностью данной структуры заключается в том, что ключи не хранятся в узлах такого дерева.

Практическое применение:

- предиктивный ввод текста (predictive text) – поиск возможных завершений слов;
- автозавершение (Autocomplete)в текстовых редакторах и IDE;
- проверка правописания (spellcheck);
- автоматическая расстановка переносов слов (hyphenation);
- Squid Caching Proxy Server.

На рисунке 1 приведен пример префиксного дерева для ключей: "A", "to", "tea", "ted", "ten", "i", "in", and "inn".

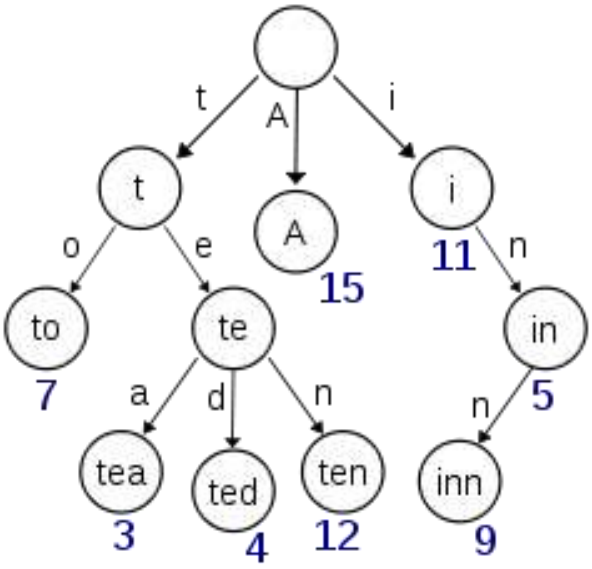

Рисунок 1 – Пример префиксного дерева

2.2. Суффиксные деревья (suffix tree)

Суффиксное дерево (suffix tree, PAT tree, position tree) – это префиксное дерево (trie), содержащее все суффиксы заданного текста (ключи) и их начальные позиции в тексте (values).

Суффиксным деревом T для строки $S = s_i \dots s_n$, называется структура данных, обладающая следующими свойствами[1]:

Представляет собой дерево с корневым узлом, имеющее ровно n листьев, пронумерованных от 1 до n;

Каждый внутренний некорневой узел имеет по меньшей мере два потомка;

Каждое ребро дерева, помечено непустой подстрокой строки S и не существует двух и более рёбер, исходящих из одного узла, помеченных строками начинающихся с одинаковых символов.

Для любого листа i дерева T, соединение всех меток рёбер на пути от корня до листа i образует подстроку $S^i = s_i \dots s_n$ строки S, т.е. суффикс строки S начинающийся с i-ой позиции.

Данный алгоритм был впервые введен Вайнеров (1973), который Дональд Кнут впоследствии охарактеризовал как «Алгоритм года 1973». В 1995 году был разработан первый алгоритм построения суффиксного дерева для заданной строки S за линейное время, сейчас известный как «Алгоритм Укконена». Наихудшее время выполнения в целом было $O\ (n \log n)$.

Множество разных задач можно решить с помощью суффиксных деревьев: начиная от задач поиска наибольшей общей подстроки, комплиментарных пар в последовательностях ДНК и заканчивая задачами сжатия информации, нечёткого поиска и кластеризации.

Практическое применение:

- поиск подстроки в строке $O(m)$, где m длина подстроки (но требуется $O(n)$ времени для построения суффиксного дерева для строки);
- поиск наибольшей повторяющейся подстроки;
- поиск наибольшей общей подстроки;
- поиск наибольшей подстроки-палиндрома;
- алгоритм LZW сжатия информации (они помогают найти повторяющиеся данные).

В качестве примера на рисунке 2 показано суффиксное дерево для текста BANANA. Суффиксы: «A$», «NA$», «ANA$», «NANA$», «ANANA$», «BANANA$».

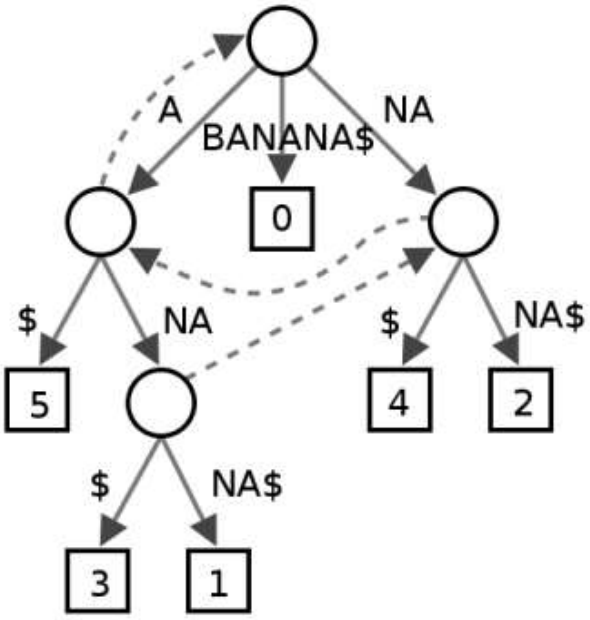

Рисунок 2 – Пример суффиксного дерева

2.3. Тернарные деревья

Тернарное дерево – один из видов префиксного дерева (trie), где узлы располагаются в виде бинарного дерева поиска.

Стоит отметить метод поиска по набору строк, базирующийся на тернарных деревьях поиска (ternary search trees). Впервые о тернарных деревьях было упомянуто ещё в 1964 году[3], однако только относительно недавно идеи лежащие в основе данной структуры данных получили своё практическое развитие[4].

Пусть для примера, необходимо осуществить поиск по следующим двенадцати двухбуквенным строкам: «as» «at» «be» «by» «he» «in» «is» «it» «of» «on» «or» «to». Тернарное дерево будет выглядеть следующим образом:

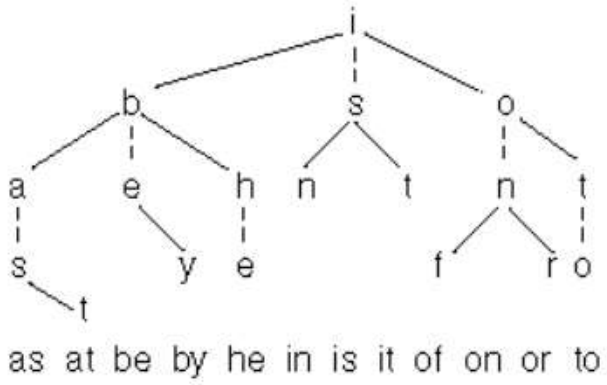

Рисунок 3 – Пример тернарного дерева

По сравнению с хеш-таблицами тернарные деревья позволяют очень быстро вернуть отрицательный результат поиска для шаблона, заведомо не содержащегося в исходных документах[1].

Стоит отметить, что тернарные деревья позволяют также эффективно осуществлять поиск по сходству, и уже много лет существует и работает система автоматического распознавания текста фирмы Bell Labs (OCR System), где для поиска терминов в словаре английского языка, объемом примерно в 34,000 символов впервые были использованы тернарные деревья поиска.

Тернарные деревья поиска можно использовать для решения многих проблем, в которых необходимо хранить большое количество строк и извлекать в произвольном порядке.

3. Сравнительный анализ

3.1. Время выполнения поиска

Поиск ключей в худшем случае будет выполняться за $O(m)$ в префиксных деревьях, т.к. время поиска не зависит от количества n элементов в словаре, а только от длины ключа (m). В бинарных деревьях сравнение ключей происходит за $O\ (log\, n)$, где n количество элементов в дереве, т.к. поиск зависит от глубины дерева. Таким образом, в худшем случае поиск в бинарном дереве занимает $O\ (m\ log\, n)$.

Префиксные деревья могу быть медленнее, например, чем хэш-таблицы в случае поиска данных, особенно если данные хранятся непосредственно на жестком диске или другом устройстве хранения время произвольного доступа является высокой по сравнению с основной памятью.

Суффиксные деревья обеспечивают поиск за $O(m + k)$ операций сравнения, где k − число всех вхождений искомой подстроки в исходной строке. Также требуется $O(n)$ линейного времени для построения суффиксного дерева для строки.

Время выполнения поиска в тернарных деревьях значительно меняется от входных данных, например, поиск будет осуществляться лучше, если будет даваться несколько

подобных строк, особенно когда эти строки имеют общий префикс. Кроме того, поиск в тернарных деревьях является более эффективным при хранении большого количества относительно коротких строк[4]. Поиск по ключам в тернаных деревьях происходит за $O\ (log\ n)$, худшее время выполнения $O(n)$.

3.2. Занимаемая память

Некоторые префиксные деревья могут требовать больше места, чем в хэш-таблице, как память может быть выделена для каждого символа в строке поиска, а не как один блок память для всех записей, как в большинстве хэш-таблиц.

Поскольку каждый узел суффиксного дерева имеет как минимум два потомка, и при этом каждый потомок порождает уникальный путь до одного и n суффиксов строки S, то это значит, что всего внутренних узлов будет не меньше чем n, и прибавив количество внешних узлов получается в итоге $2n$ узлов, из этого следует, что можно построить суффиксное дерево, занимающее $O(n)$ дискового пространства. Однако более подробные оценки требуемого места для суффиксного дерева в реальных системах, менее утешительны, так для текста длинны n, дерево в среднем занимает порядка $20n$ байт.

4. Заключение

В статье были рассмотрены три вида древовидных структур данных, указаны их основные особенности, приведены примеры практических применений. Проведен анализ по двум критериям: во времени выполнения поиска по ключам, а также объему занимаемой памяти.

Список информационных источников

[1] Адаманский А. Обзор методов и алгоритмов полнотекстового поиска. Новосибирский государственный университет. http://callisto.nsu.ru/documentation/searchreview.pdf.

[2] Jon Bentley, Bob Sedgewick. Ternary Search Trees. http://www.drdobbs.com/database/ternary-search-trees/184410528.

[3] Randomized Binary Searching with Tree Structures. Communications of the ACM, March, 1964.

[4] Bentley, Sedgewick. "Fast Algorithms for Sorting and Searching Strings". SODA, 1997.

[5] А. Б. Веретенников, Ю. С. Лукач. CLB-деревья: новый способ индексации. Известия УрГУ. Информационные технологии. 2006.

[6] Круглов А.М. Актуализация сведений о данных информационной системы средствами активного словаря-справочника данных / А.М. Круглов, А.В. Будихин, Д.А. Буров, А.В. Остроух // Научный вестник МГТУ ГА. Серия Аэромеханика и прочность. – 2007. - №119 (9). – С. 166-171.

[7] Николаев А.В. Принципы организации динамических интерфейсов доступа к данным с использованием словарей-справочников данных / А.В. Николаев, А.В.

Будихин, Д.А. Буров, А.В. Остроух // Научный вестник МГТУ ГА. Серия Аэромеханика и прочность. – 2007. - №119 (9). – С. 172-178.

[8] Николаев А.В. Использование словаря-справочника данных для реализации пользовательских средств обработки информации / А.В. Остроух, С.А. Будихин, А.П. Баринов, А.В. Николаев // Приборы и системы. Управление, контроль, диагностика. – М.: «Научтехлитиздат», 2008. – №3. – С. 13-15.

[9] Пшеничный Д.А. Анализ параметров и сравнение СУБД для реализации информационного обеспечения промышленного предприятия / Д.А. Пшеничный, А.В. Будихин, А.В. Остроух // Промышленные АСУ и контроллеры. - М.: «Научтехлитиздат», 2010. - №12. - С. 7-11.

[10] Помазанов А.В. Методика оптимизации баз данных / А.В. Помазанов, А.В. Остроух, А.И. Белоусова, А.О. Васильева // В мире научных открытий. Серия «Проблемы науки и образования». - 2012. - №12. - С.49-54.

[11] Белоусова А.И. Подход к формированию многоуровневой модели мультиагентной системы с использованием миваров / А.И. Белоусова, О.О. Варламов, М.Н. Краснянский, А.В. Остроух // Перспективы науки – Тамбов. «ТМБПринт», 2011. – № 5(20). – С. 57-61.

[12] A.V. Ostroukh, M.N. Krasnyanskiy, S.V. Karpushkin, A.D. Obukhov. Development of Automated Control System for University Research Projects // Middle East Journal of Scientific Research. 2014. Vol. 20 (12). pp. 1780-1784. DOI: 10.5829/idosi.mejsr.2014.20.12.21091.

[13] A. Ostroukh, A. Pomazanov. Realtime Development and Testing of Distributed Data Processing System for Industrial Company // Middle East Journal of Scientific Research. 2014. Vol. 20 (12). pp. 2184-2193. DOI: 10.5829/idosi.mejsr.2014.20.12.21106.

[14] Krasnyanskiy M.N., Karpushkin S.V., Obukhov A.D., Ostroukh A.V. Automated control system for university research projects // International Journal of Advanced Studies (iJAS). 2014. Vol. 4, Issue 1, pp. 22-26. DOI: 10.12731/2227-930X-2014-1-4.

[15] Mikhail Nikolaevich Krasnyanskiy, Andrey Vladimirovich Ostroukh, Sergey Viktorovich Karpushkin, Artyom Dmitrievich Obukhov, Nataliya Vyacheslavovna Molotkova and Irnina Vladimirovna Galygina. Electronic Document Management Systems Structure for University Research and Education // Journal of Engineering and Applied Sciences. 2014. Vol 9. Issue 5. pp. 182-189. DOI: 10.3923/jeasci.2014.182.189.