МЕТОД ОРГАНИЗАЦИИ ИНДЕКСА КОЛЛЕКЦИИ XML ДОКУМЕНТОВ

Веретенников А.Б.[1]
e-mail: alexander@veretennikov.ru

Автором ведется разработка системы для полнотекстового поиска в большом массиве текстовых документов [1, 2]. Одним из важных типов текстовых документов является XML документ. Под поисковым запросом будем понимать набор слов или фразу. Обычно, поисковые системы выдают в качестве результата поиска набор документов, которые содержат искомые слова, а также информацию о том, в каком месте документа находятся эти слова.

При поиске в XML документе этого недостаточно. XML документ это набор элементов, каждый из которых является отдельной сущностью, поэтому имеет смысл в результаты поиска включить информацию о том, в каком элементе содержатся искомые слова. Для этого можно взять XPath данного элемента, как в [3].

Пример результата поиска: $1.xml$, $/root/section[12]/p[23]$. Т. е. искомые слова находятся в документе $1.xml$, где внутри тега *root*, в 12-м по порядку теге *section*, есть теги p, и в 23-м из них и содержаться искомые слова. Кроме того, допустим, p имеет атрибут id, который обозначим как ключевой. Имеет смысл получить XPath в виде $/root/section[12]/p[id = 4356]$, где 4356 – значение атрибута id у 23-го тега p.

Часто для поиска в XML документах используются запросы, основанные на XPath или XQuery [4, 5]. При этом используются B-деревья или аналогичные индексы, но для построения таких индексов для более менее большой коллекции (например, 10-20 Гб) требуется весьма значительное время.

Полнотекстовый индекс, основанный на инвертированных файлах или CLB-индекс [1, 2] строится гораздо быстрее. Разработанный автором метод позволяет получить в результатах поиска XPath искомого элемента. Скорость создания индекса и поиска практически не снижается, по сравнению с обработкой обычных текстов. Если ищем с учетом некоторого XPath, то из результатов поиска можно исключить те элементы, которые не соответствуют этому XPath.

[1] www.veretennikov.org

Для организации индекса используем следующую модель данных:

1) Файл с информацией о документах – содержит описание проиндексированных документов. Это имя документа, ID документа и т. д. для каждого документа.

2) Индекс – содержит для каждого слова список его вхождений в обработанных документах, т. е. например, записи вида (ID, Position), где ID – ID документа, Position – позиция слова в документе.

3) Репозитарий – содержит текст проиндексированных документов, для быстрого извлечения фрагментов текста при включении их в результаты поиска.

В [6] для решения подобной задачи предлагается по сути в качестве одного документа взять каждый элемент исходного XML документа. Однако при данном подходе будут значительно увеличены 1 и 2 компоненты системы, за счет увеличения количества документов.

Вместо этого автор предлагает использовать репозитарий. Для быстрого извлечения фрагмента текста репозитарий разделен на блоки фиксированного размера (обычно 4 Кб). Когда пользователю выводятся результаты поиска, мы читаем из репозитария блок, в котором содержатся искомые слова. В начале этого блока сохранен текущий XPath, соответствующий началу блока в XML документе. Далее читаем текст в блоке, пока не дойдем до местоположения искомых слов. При чтении текста, при обнаружении XML тегов осуществляется корректировка XPath. В результате получаем XPath, соответствующий искомым словам.

Можно хранить XML документ как он есть. Но, т. к., теги могут повторяться, как в одном так и в нескольких документах, и более того, в наборе XML документов может использоваться одно и то же пространство имен, структуру тегов можно сохранить иначе. В отдельном файле храним информацию о пространствах имен. В результате каждое пространство имен имеет свой ID, каждый тег в пространстве имен имеет свой ID (уникальный в данном пространстве имен).

Вместо исходных конструкций вида <namespace:tag> в репозитарии сохраняем пару чисел $namespace\_id$, tag_id. Где tag_id – ID тега, $namespace\_id$ – ID пространства имен. $namespace\_id$ может быть как глобальным ID, так и локальным ID, взятым относительно текущего документа. При этом, скорее всего данные ID-ы будут

небольшими числами, и поместятся в 1 байт каждый, т. е., размеры хранимых данных уменьшаются. Используется специальный метод кодирования, чтобы можно было отличить ID-ы тегов от остальных символов текста. Также сохраняется информация о конце тега, начале, окончании атрибута.

Автором были обработаны 50 Гб XML документов (120 тыс. файлов). Скорость создания индекса меньше чем для простых текстовых файлов, но незначительно, и по сути обусловлена накладными расходами использования XML парсера. Скорость поиска по сравнению с поиском в обычных текстовых файлах практически не изменилась.

Литература

[1] *Веретенников А.Б.* Эффективная индексация текстовых документов с использованием CLB-деревьев. // Системы управления и информационные технологии, 2009, 1.1(35). - С. 134–139.

[2] *Веретенников А.Б.* Программный комплекс и эффективные методы организации и индексации больших массивов текстов. // Диссертация на соискание ученой степени кандидата физико-математических наук. Екатеринбург, 2009.

[3] *David Mertz* XML Matters: Indexing XML documents // http://www.ibm.com/developerworks/xml/library/x-matters10.html.

[4] *Wei Wang, Hongzhi Wang, Hongjun Lu, Haifeng Jiang, Xuemin Lin, Jianzhong Li* Efficient Processing of XML Path Queries Using the Disk-based F&B Index // Proceedings of the 31st VLDB Conference, Trondheim, Norway, 2005.

[5] *Haifeng Jiang, Hongjun Lu, Wei Wang* XR-Tree: Indexing XML Data for Efcient Structural Joins // Proc. Int'l Conf. Data Eng. (ICDE 02), IEEE CS Press, 2002, pp. 253–263.

[6] *Jaap Kamps, Maarten Marx, Maarten de Rijke, Borkur Sigurbjornsson* XML Retrieval: What to Retrieve? // SIGIR'03, July 28–August 1, 2003, Toronto, Canada.