Journal of Advances in Computer Research
Quarterly ISSN: 2008-6148
Sari Branch, Islamic Azad University, Sari, I.R.Iran
(Vol. 3, No. 3, August 2012), Pages: 1-7
www.jacr.iausari.ac.ir

Improved Skips for Faster Postings List Intersection

Faraein Aeini[1]✉, Fariborz Mahmoudi[2], Naeim UsefiFard[3]
(1) Ph.D. Student, Department of Computer Engineering, Sari Branch, Islamic Azad University, Sari, Iran
(2) Department of Computer Engineering, Qazvin Branch, Islamic Azad University, Qazvin, Iran
(3) Lecturer, Shiraz University, Shiraz, Iran
faraein.aeini@qiau.ac.ir; mahmoudi@qiav.ac.ir; n.usefifard@gmail.com

Received: 2012/06/11; Accepted: 2012/06/21

Abstract
Information retrieval can be achieved through computerized processes by generating a list of relevant responses to a query. The document processor, matching function and query analyzer are the main components of an information retrieval system. Document retrieval system is fundamentally based on: Boolean, vector-space, probabilistic, and language models. In this paper, a new methodology for matching function of Boolean retrieval systems is proposed and tried to extend postings list data structures and increase the efficiency of using postings lists and skips. The final effect of these considerations is in decreasing the search time.

Keywords: *Information retrieval, Boolean retrieval, Inverted index, skip pointer*

1. Introduction

The amount of data in the Internet and World-Wide Web is increasing unexpectedly nowadays. The weak structure in web information source, heterogeneity, and data deluge in especially in World-Wide Web put forth the difficulties to access the data via browsing and searching. Therefore, we face different data formats, which raise the maintenance cost in web mining application (like, Comparison shopping appliances).

The Information Extraction (IE) burdens a huge endeavour to translate the input pages to structured data automatically. This structured data, which is produced in IE make it possible for the web mining application and searching tool to be ready for post-processing. While in information retrieval (IR), recognizing the relevant documents through a collection of documents is the main concern [1][2].

Formally, an IE task is defined by its input and its extraction target. The input can be unstructured documents like free text that are written in natural language or the semi-structured documents that are pervasive on the Web, such as tables or itemized and enumerated lists. The extraction target of an IE task can be a relation of k-tuple (where k is the number of attributes in a record) or it can be a complex object with hierarchically organized data. For some IE tasks, an attribute may have zero (missing) or multiple instantiations in a record. The difficulty of an IE task can be further complicated when various permutations of attributes or typographical errors occur in the input documents.

There are different ways to extract tokens from documents in document processing stage. After that, identify relevant documents to query is the most important issue.

In this paper, an efficient method for reducing the number of comparisons to determine the relevant documents with query is proposed.

An important contraption in this paper is extensions to postings data structures and ways to increase the efficiency of using postings lists.

The final effect of these considerations is in decreasing the search time and increasing the system accuracy.

The paper continue, first we discuss about related works. In third section we describe our proposed method. In the fourth section there is a discussion about the practical results of proposed method and its comparison with previous method and the final resulting is mentioned in the fifth section.

2. Related Work

There are a lot of works about Boolean information retrieval. For example, investigated document ranking methods in thesaurus-based Boolean retrieval systems, and proposed a new thesaurus-based ranking algorithm called the Extended Relevance (E-Relevance) algorithm [3].

Daniel Z. Zanger [4] proposed a theory asserted roughly that whenever two or more documents are similarly ranked at any two points along the *p*-continuum with respect to this model for either an AND or an OR query containing exactly two terms, then they were similarly ranked at all points in between.

Shmuel T. Klein [5], proposed heuristics for choosing a good order are suggested for applications include negated terms in Boolean queries.

Due to increase in web-based applications, the need for growth information retrieval system that accommodate user's needs become critical. A wide range of commercial information retrieval systems are based on standard Boolean model and at less scale vector models. Although the imperfections of these models are now part of text-book knowledge, the expansion of new models still have to overcome the possibility and testing challenge.

The simplest form of document retrieval is for a computer to do this sort of linear scan through documents. This process is commonly referred to as *grepping*through text. But it is not proper way to process large document collections quickly [6].

The method to avoid linearly scanning the texts for each query is to *index* the documents. We now cannot construct a term-document matrix in a naive way. A much better representation is to save only the things that do occur, that is, the 1 position, which called *inverted index* and has become the standard term in information retrieval. We keep a *dictionary* of terms. Then for each term, we have a list that records which documents the term occurs in. Each item in the list – which records that a term appeared in a document– is formally called a *posting*. We walk through the two postings lists simultaneously, in time linear in the total number of postings entries. If the list lengths are m and n, the intersection takes $O(m + n)$ operations.

Skip pointers are impressively shortcuts that allow us to avoid processing parts of the postings list that will not figure in the search results [6].

3. The Proposed Method

For extracting the data from web, takes data from user, search it and get well selected pages. For example, the crawler starts from given pages and linked those pages [7].

Diagram of the proposed method for Boolean information retrieval is illustrated in Figure 1 that consists of three basic parts namely token extraction, inverted index construction and inverted index comparison at the end. Each part is described in continue of this section.

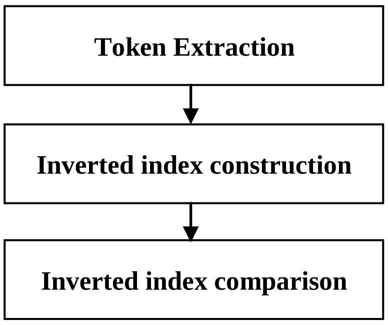

Figure1. The block diagram of our proposed system

Token Extraction

Token extraction is outside the scope of this article. We used methodology proposed by Ji et al. in [8] to extract exclusive content of web pages. In their work, to extract meaningful information including records and data schema from the kind of pages, a method based on Tag tree template is proposed. The approach is based on monitoring the variable fields in Tag tree. Hence, the extracted text information can be meaningful with rich structure information. The records in pages and the schema of the records can be extracted from the exclusive content by finding repeating patterns and using some heuristic rules [8].

Inverted Index Construction

One of the most important innovations of the proposed method is improving posting list data structure at indexing time.

In previous works, successive integers are dedicated to each new document in first confrontation in the duration of index construction. There are the lists of normalized tokens for each document as indexing input. The structure of input lists contains a pair of term and docID in inverted index. [6].

A fixed length array would be useless as some words occur in many documents, and others in very few. For an in-memory postings list, two good alternatives are singly linked lists or variable length arrays. Singly linked lists allow cheap insertion of documents into postings lists (following updates, such as when recrawling the web for updated documents), and naturally extend to more advanced indexing strategies such as skip lists, which require additional pointers. Variable length arrays win in space requirements by avoiding the overhead for pointers and in time requirements because their use of contiguous memory increases speed on modern processors with memory caches.

Owing to the fact that the inverted index update only at specified time intervals, use of linked list and variable length array is proposed in this paper. First, postings lists implement by link list, and save this in disk. At each indexing time, first postings lists by link list data structure update and then map into an array by appropriate length. So, not only we can benefit from cheap update or insertion of new documents into postings lists, but also this makes it possible to be more compact and faster to traverse.

Extra pointers can in practice be encoded into the lists as offsets.

Inverted Index Comparison

Our proposed algorithm for faster postings list intersection via Improved Skips, called IMSkips, shown in figure.2.

The place of skips is one of the effective issues in the number of comparison. It means having more skips bring shorter skip spans which cause a large number of comparisons and huge amount of storage space while fewer skips lead to few pointer comparisons and long skip spins (fewer opportunities to skip).[6]

When subtraction between docIDs of two consecutive posting is large value, long skip spans is preferred. In contrast, if this value is small, shorter skip spans would be better choice.

Furthermore, in general approach of using skip, the postings intersection can use a skip pointer when the end point is still less than the item on the other list. If not, postings between this skip and next skip should be process consecutively, and there is not any jump available for the postings in this interval.

A new innovation, which has been found to work well in practice, is proposed to improve upon.

To solve above-mentioned problem, we proposed to use long evenly-spaced skip pointers, and define a threshold for each posting list based on the average of difference between their docIDs in consecutive postings.

Our proposed algorithm to process the postings has two main situations.

1. If difference between docID(p1) and docID(p2) values be less than the threshold value, the consecutive processing is proposed by our algorithm, even if it is given the opportunity to compare with the skip.
2. Otherwise, the algorithm proposed to check first skip after docID (p1) to process, even if docID (p1) is located between two consecutive skip. This checking is possible by low cost, because of implementing the posting list by array.

This improvement in our proposed algorithm can be proven empirically, and we can usually process postings list intersection in sub linear time.

By checking the skip of docID(p1) and docID(p2), two situation could be occurred:

2.1 In the first situation, skip's docID(p1) is smaller than docID(p2), and there is possibility for skip. The skip continues until the skip is possible.

2.2 In the second situation, skip of docID(p1) is larger than docID(p2), so there is no advantage in skip. Our algorithm is proposed to use binary search in this situation on docIDs by indices ((p1)+1 , skip(p1)-1), rather than consecutive comparison of posting lists. Therefore, the number of comparisons decreases from k to log (k). Where, k is the span of the skip.

Finally, it should be noted that the presence of skip only helps for AND queries, not for OR queries.

Consider Figure 3 as an example. The average difference between the docIDs in consecutive postings in upper list is 7, that is [(3-1) + (5-3)+….+(83-74)]/17, So threshold value is 7. Suppose we've stepped through the lists in the figure until we have matched 3 on each list and moved it to the results list.

```
    Intersect ImSkips (p1,p2)
      Answer←<>
      WHILE p1 <> NULL AND p2 <> NULL   DO
        IF docID(p1) = docID(p2) THEN
    ADD(Answer, docID(p1));
    P2 = p2 + 1;
    IF (docID(p1)-docID(p2)<=MidP)
              P1 = p1 + 1;
    ELSE
     IF docID(skip(p1)) <=docID(p2)
        WHILE docID(Skip(p1))<=docID(p2) DO
                   P1 = SKIP (p1);
          ELSE
              Min = p1+1;
        Max = SKIP(p1) -1;
                 WHILE (MIN<=Max AND Found=False) DO
                   Mid = (Min +Max)/2;
                   IF docID(Mid) >docID(p2)
                             Max = Mid – 1;
                   ELSE IF docID(Mid) <docID(p2)
                             Min = Mid+1;
                   ELSE
                   Begin
                             Found = True;
                             P1 = Min;
                   END
        ELSE IF docID(p1) <docID(p2) THEN
    IF (docID(p1)-docID(p2)<=MidP)
              P1 = p1 + 1;
    ELSE
     IF docID(skip(p1)) <=docID(p2)
        WHILE docID(Skip(p1))<=docID(p2) DO
                   P1 = SKIP (p1);
          ELSE
              Min = p1+1;
        Max = SKIP(p1) -1;
                 WHILE (MIN<=Max AND Found=False) DO
                   Mid = (Min +Max)/2;
                   IF docID(Mid) >docID(p2)
                             Max = Mid – 1;
                   ELSE IF docID(Mid) <docID(p2)
                             Min = Mid+1;
                   ELSE
                   Begin
                             Found = True;
                             P1 = Min;
          END
        ELSE
P2 = p2 + 1;
```

Figure2. Postings lists intersection with proposed algorithm (IMSkips)

In classic skip method, for docID(p2)=32 we should step through the list until next skip. But in our proposed method, because difference between 3 and 32 is more than 7, The Algorithm compare the value of docID(p2) with skip of docID(p1) (i.e. 25). So easily decrease the number of comparisons.

P1:

1	3	5	6	9	12	15	17	25	32	48	56	57	63	69	74	83

P2:

3	32	74

Figure3. Postings list with skip pointers. The red numbers show skips.

Another situation occurs when we looking for next posting (i.e. 74) in lower list. In classic skip method, we should step through the upper list until reach to docID (p1) = 74, or larger value. But according to our proposed method, by using binary search the same result are achieved with fewer comparisons. This improvement is predominant in longer posting list.

4. Experimental Results

This section evaluates and compares the performance of the proposed method against previous method, which is implemented on a same dataset.

In our experiments, we use dotIR dataset that (available in http://ece.ut.ac.ir/DBRG/webir/download.html) contains one million Persian web pages. 500 pages were selected randomly among them to construct inverted index. Moreover, 50 queries are used that have already prepared for this dataset in their website.

To evaluate and comparing methods, we used from the number of comparisons between postings entries. As can be seen in Table.I, the proposed method is always better results in compare with the previous method.

Our experiments are repeated with two skip span. In first experiment, we select $\sqrt{P}$ evenly-spaced skip places for a postings list of length p, and in second we use from $\frac{3}{2}\sqrt{P}$. As seen from the results, the proposed method works better with longer skip span.

Comparison between classic skip pointers and proposed method is shown that the proposed method is superior, especially with dealing with diffuse distribution docID in postings lists, for example like p2 in figure.3.

The experiments demonstrate that our method efficiently reduce the searching time among distinguished related pages for intersection queries.

Table1. Experimental results

Methods / Skip Span	Classic skip pointer (The number of Comparisons)	Proposed Method (IMSkips) (The number of Comparisons)
$\sqrt{P}$	776	592
$3/2\sqrt{P}$	784	464

5. Conclusion and future work

In this paper, the parsed web pages are shown in Tag trees, consequently the template generates for each site. The generated templates were made according to cost-based tree measurement. The records in pages and the schema of the records generally extracted with the help of finding repeating patterns and using some heuristic rules from the exclusive content.

We proposed improved skips in order to achieve the same intersection query result with relatively fewer numbers of comparisons. In this regard, the array structure is used in implementation of postings list. This implementation makes it possible to use binary search instead of linear search. Also, we put forward using heuristic rule to compare postings between two skips, instead of postings linear comparison.

The future works can be work on optimizing the method of finding effective tokens and constructing inverted index with the help of the shorter postings lists.

6. References

[1] Chia-Hui Chang, Mohammed Kayed, Moheb Ramzy Girgis, Khaled Shaalan, A Survey of Web Information Extraction Systems, IEEE, 2009.

[2] Kshitija Pol, et al, A Survey on Web Content Mining and extraction of Structured and Semi structured data, IEEE, 2008.

[3] Joon Ho Lee, Myoung Ho Kim, Yoon Joon Lee, Ranking documents in thesaurus-based boolean retrieval systems, Information Processing & Management, 2002, Pages 79–91.

[4] Daniel Z. Zanger, Booz, Allen & Hamilton, Greensboro Dr, Interpolation of the extended Boolean retrieval model, Information Processing & Management Volume 38, Issue 6, 2002, Pages 743–748.

[5] Shmuel T. Klein, On the use of negation in Boolean IR queries, Information Processing & Management Volume 45, Issue 2, 2009, Pages 298–311.

[6] C D. Manning, P Raghavan, H Schütze, An Introduction to Information Retrieval, Cambridge University Press Cambridge, England 2009.

[7] Martin Ester, Hans-Peter Kriegel, Mattis Schubert , "Accurate and efficient Crawling for relevant Websites".

[8] X Ji, J Zeng, et.al, Tag tree template for Web information and schema extraction, Expert Systems with applications 37 (2010) 8492–8498.