Chapter 127
An Efficient and Effective Index Structure for Query Evaluation in Search Engines

Yangjun Chen
University of Winnipeg, Canada

ABSTRACT

In this chapter, the authors discuss an efficient and effective index mechanism for search engines to support both conjunctive and disjunctive queries. The main idea behind it is to decompose an inverted list into a collection of disjoint sub-lists. The authors associate each word with an interval sequence, which is created by applying a kind of tree coding to a trie structure constructed over all the word sequences in a database. Then, attach each interval, instead of a word, with an inverted sub-list. In this way, both set intersection and union can be conducted by performing a series of simple interval containment checks. Experiments have been conducted, which shows that the new index is promising. Also, how to maintain indices, when inserting or deleting documents, is discussed in great detail.

INTRODUCTION

Indexing the Web for fast keyword search is among the most challenging applications for scalable data management. In the past several decades, different indexing methods have been developed to speed up text search, such as inverted files, signature files and signature trees for indexing texts (Anh and Moffat, 2005; Chen et al., 2004; Chen et al. 2006; Faloutsos, 1985; Faloutsos et al., 1988); and suffix trees and tries (Knuth, 1975) for string matching. Especially, different variants of inverted files have been used by the Web search engines to find pages satisfying a query (Arasu, 2001; Lemple et al., 2003).

A text database can be roughly viewed as a collection of documents and each document is stored as a list of words. Over the documents, there are two kinds of Boolean queries, that is, queries that can be constructed from query terms by conjunction (∧) or disjunction (∨). A document D is an answer to a conjunctive query $w_1 \wedge w_2 \wedge \ldots \wedge w_k$ if it contains every w_i for $1 \leq i \leq k$ while D is an answer to a

DOI: 10.4018/978-1-5225-7598-6.ch127

disjunctive query $w_1 \vee w_2 \vee \ldots \vee w_l$ if it contains any w_i for $1 \leq i \leq l$. Conjunction and disjunction can be nested to arbitrary depth, but can always be transformed to a conjunctive normal form:

$$(w_{11.} \ldots \vee\ w_{1l_1}\ .) \ldots \wedge (w_{k1.} \ldots \vee\ w_{kl_k}\ .)$$

In this chapter, we discuss a new method to evaluate both conjunctive and disjunctive queries by decomposing an inverted list into a collection of disjoint sub-lists. The decomposition is based on the construction of a trie structure *T* over documents and then associating each document word with an interval sequence generated by labeling *T* by using a kind of tree encoding.

With this method, we can improve the efficiency of traditional methods by an order of magnitude or more.

BACKGROUND

In order to efficiently evaluate such queries, indexes need to be established. It is well known that English texts typically contain many different variants of basic words, by using variant word endings such as 'ing', 'ed', 'ses', and 'ation'. All the variants of a word should be regarded as a match and therefore it is efficient for an index only include these basic words, or say, stems. Different algorithms have been developed to extract stems from documents. Among them, the algorithm proposed by Lovins (1968) is widely used.

By the signature file, a word is hashed to a bit string (called a signature) and all the words' signatures of a document are superimposed (bit-wise OR operation) into a document signature. When a query arrives, its signature will be created using the same hash function and the document signatures are scanned and many nonqualifying documents are discarded. The rest are either checked (so that the 'false drops' are removed) or they are returned to the user as they are. The main disadvantage of this method is the false drop (Kitagawa et al., 1997), which needs extra time to check. The signature file is greatly improved by the so-called signature tree (Chen et al. 2006), by which a set of signatures is organized into a binary tree structure and a sequential search of signatures is replaced with a search of binary trees. However, signature-based methods can be used only for evaluating conjunctive queries. For disjunctive queries, they are not efficient.

As pointed out by many researchers (Anh et al., 2005; Ao et al., 2011; Zobel et al., 2006), the inverted file is a more competitive indexing method than signature-based approaches. It is extensively used by different web search engines due to its efficiency and simplicity. Structurally, it contains two parts: a search structure or vocabulary, containing all the distinct words to be indexed, and a set of inverted lists with each constructed for a distinct word *w*, storing the identifiers of all those documents containing *w*. Queries are evaluated by fetching the inverted lists for the query terms, and then intersecting them for conjunctive queries, or merging them (by a set union operation) for disjunctive queries. According to (Zobel et al., 1998), the inverted file is superior to the signature file in almost every respect, including functionality, query time, and space overhead.

Since it was first proposed in mid-1960s, the inverted file has been adopted in information retrieval, database systems, distributed systems (Büttcher et al., 2005; Camel et al., 2001), and different search engines. Also, much effort has been spent on the improvement of its performance by using integer coding

Related Content