Fast Set Intersection and Two-Patterns Matching

Hagai Cohen and Ely Porat*

Department of Computer Science, Bar-Ilan University, 52900 Ramat-Gan, Israel
{cohenh5,porately}@cs.biu.ac.il

Abstract. In this paper we present a new problem, the *fast set intersection* problem, which is to preprocess a collection of sets in order to efficiently report the intersection of any two sets in the collection. In addition we suggest new solutions for the *two-dimensional substring indexing* problem and the *document listing* problem for two patterns by reduction to the *fast set intersection* problem.

1 Introduction and Related Work

The intersection of large sets is a common problem in the context of retrieval algorithms, search engines, evaluation of relational queries and more. Relational databases use indices to decrease query time, but when a query involves two different indices, each one returning a different set of results, we have to intersect these two sets to get the final answer. The running time of this task depends on the size of each set, which can be large and make the query evaluation take longer even if the number of results is small. In information retrieval there is a great use of inverted index as a major indexing structure for mapping a word to the set of documents that contain that word. Given a word, it is easy to get from the inverted index the set of all the documents that contain that word. Nevertheless, if we would like to search for two words to get all documents that contain both, the inverted index doesn't help us that much. We have to calculate the occurrences set for each word and intersect these two sets. The problem of intersecting sets finds its motivation also in web search engines where the dataset is very large.

Various algorithms to improve the problem of intersecting sets have been introduced in the literature. Demaine et al. [1] proposed a method for computing the intersection of k sorted sets using an adaptive algorithm. Baeza-Yates [2] proposed an algorithm to improve the multiple searching problem which is related directly to computing the intersection of two sets. Barbay et al. [3] showed that using interpolation search improves the performance of adaptive intersection algorithms. They introduced an intersection algorithm for two sorted sequences that is fast on average. In addition Philip et al. [4] presented a solution for

* This work was supported by BSF and ISF.

A. López-Ortiz (Ed.): LATIN 2010, LNCS 6034, pp. 234–242, 2010.

computing expressions on given sets involving unions and intersections. A special case of their result is the intersectin of m sets containing N elements in total, which they solve in expected time $O(N(\log \omega)^2/\omega + m \cdot output)$ for word size ω where *output* is the number of elements in the intersection.

In this paper we present a new problem, the *fast set intersection* problem. This problem is to preprocess a databases of size N consisting of a collection of m sets to answer queries in which we are given two set indices $i, j \leq m$, and wish to find their intersection. This problem has lots of applications where there is a need to intersect two sets in a lot of different fields like Information Retrieval, Web Searching, Document Indexing, Databases etc. An optimal solution for this problem will bring better solutions to various applications.

We solve this problem using minimal space and still decrease the query time by using a preprocessing part. Our solution is the first non-trivial algorithm for this problem. We give a solution that requires linear space with worst case query time bounded by $O(\sqrt{N output} + output)$ where *output* is the intersection size.

In addition, we present a solution for the *two-dimensional substring indexing* problem, introduced by Muthukrishnan et al. [5]. In this problem we preprocess a database D of size N. So when given a string pair (σ_1, σ_2), we wish to return all the database string pairs $\alpha_i \in D$ such that σ_1 is a substring of $\alpha_{i,1}$ and σ_2 is a substring of $\alpha_{i,2}$. Muthukrishnan et al. suggested a tunable solution for this problem which uses $O(N^{2-y})$ space for a positive fraction y and query time of $O(N^y + output)$ where *output* is the number of such string pairs. We present a solution for this problem, based on solving the *fast set intersection* problem, that uses $O(N \log N)$ space with $O((\sqrt{N \log N output} + output) \log^2 N)$ query time.

In the *document listing* problem which was presented by Muthukrishnan [6], we are given a collection of size N of text documents which may be preprocessed so when given a pattern p we want to return the set of all the documents that contain that pattern. Muthukrishnan suggested an optimal solution for this problem which requires $O(N)$ space with $O(|p| + output)$ query time where *output* is the number of documents that contain the pattern. However, there is no optimal solution when given a query consists of two patterns p, q to return the set of all the documents that contain them both. The only known solution for this problem is of Muthukrishnan [6] which suggested a solution that uses $O(N\sqrt{N})$ space which supports queries in time $O(|p| + |q| + \sqrt{N} + output)$. We present a solution for the *document listing* problem when the query consists of two patterns. Our solution uses $O(N \log N)$ space with $O(|p| + |q| + (\sqrt{N \log N output} + output) \log^2 N)$ query time.

The paper is structured as follows: In Sect. 2 we describe the *fast set intersection* problem. In Sect. 3 we describe our solution for this problem. In Sect. 4 we present similar problems with their solutions. In Sect. 5 we present our solution for the *two-dimensional substring indexing* problem and the *document listing* problem for two patterns. In Sect. 6 we present some concluding remarks.