MapReduce Based Information Retrieval Algorithms for Efficient Ranking of Webpages

Srinivasa K.G., M.S. Ramaiah Institute of Technology, India
Anil Kumar Muppalla, M.S. Ramaiah Institute of Technology, India
Bharghava Varun A., M.S. Ramaiah Institute of Technology, India
Amulya M., M.S. Ramaiah Institute of Technology, India

ABSTRACT
In this paper, the authors discuss the MapReduce implementation of crawler, indexer and ranking algorithms in search engines. The proposed algorithms are used in search engines to retrieve results from the World Wide Web. A crawler and an indexer in a MapReduce environment are used to improve the speed of crawling and indexing. The proposed ranking algorithm is an iterative method that makes use of the link structure of the Web and is developed using MapReduce framework to improve the speed of convergence of ranking the WebPages. Categorization is used to retrieve and order the results according to the user choice to personalize the search. A new score is introduced in this paper that is associated with each WebPage and is calculated using user’s query and number of occurrences of the terms in the query in the document corpus. The experiments are conducted on Web graph datasets and the results are compared with the serial versions of crawler, indexer and ranking algorithms.

Keywords: Crawler, Information Retrieval, MapReduce, PageRank, Retrieval Algorithms, Webpage Rankings

1. INTRODUCTION

World Wide Web is growing very rapidly from few thousand pages to more than two billion pages at present. The use of web has elevated significant social concerns in the fields of privacy, censorship and access to information (Pinkerton, 2000). Web data consists of enormous number of web pages and links connecting them. Due to its size and dynamic nature it becomes highly difficult to locate the information efficiently across the web. Hence search engines play a vital role in retrieving the relevant and accurate data from the web (Butcher & Clarke, 2010) and they rely upon the huge collection of web pages obtained by the web crawler, a computer program which browses the World Wide Web in a completely ordered and automated fashion.

Mining for specific data using sequential information retrieval algorithms is time con-
suming and inefficient. Hence there is a need to develop more parallel algorithms to search and retrieve the user specific data from the ever expanding World Wide Web. The aim of this paper is to utilize a MapReduce framework to build a crawler, indexer and ranking system to retrieve information from the Web and cater to the personalization of the data using Machine learning concepts. The problem with the retrieved results is the order in which they are displayed, as millions of results are obtained it becomes difficult for the user to locate the relevant one. Hence machine learning concept is used to identify the most popular items from the Web and rank them so that it becomes easier for the user to retrieve the popular and important information from the Web (Croft, Metzler, & Strohman, 2010).

PageRank (Brin & Page, 1998) is a link analysis algorithm that assigns a numerical weighting (Qin, Liu, Xu, & Li, 2007) to each element of a hyperlinked set of documents, such as the World Wide Web, with the purpose of “measuring” its relative importance within the set. PageRank hypothesizes that a link from page A to B implies the author of A recommends having a look at page B. A page is said to have a high PageRank value if there are many pages that point to it (Kim & Lee, 2002; Shenoy et al., 2004). The PageRank algorithm and implementation details are explained in detail in the later section. The algorithm represents the structure of the web as a matrix and PageRank as a vector. The PageRank vector is derived by computing matrix-vector multiplications repeatedly. The sum of all PageRank values should be one during computation. In this paper we use MapReduce framework to compute the ranks of the web pages correctly. Experimental evaluation is carried out and its results are briefly discussed.

The objective of this paper is to develop a web portal application with searching capability that runs as a website. As an outcome, the paper also aims at testing the performance of the proposed algorithms on various data sets. Parameters like retrieval time, computation time while ranking of web pages based on popularity and others are tested. The remainder of the paper is organised as follows. Section 2 discusses the related work and Section 3 gives the contributions of the paper. Sections 4 and 5 contain the details of the architectures of crawler, indexer and the ranking system proposed algorithms followed by the pseudocode of all the techniques used. An illustrative example is given in Section 6. Experimental results including timings of the iterative methods on graphs are discussed in Section 7 followed by conclusions.

2. BACKGROUND AND RELATED WORK

According to Google’s MapReduce Programming Model (Lammel, 2008) follows the following five simple steps, i.e., (i) iteration over the input; (ii) computation of key/value pairs from each piece of input; (iii) grouping of all intermediate values by key; (iv) iteration over the resulting groups; and (v) reduction of each group. MapReduce framework is used to develop parallel algorithms. It is a programming model and an associated implementation for processing and generating large data sets. Users specify a map function that processes a key/value pair to generate a set of intermediate key/value pairs, and a reduce function that merges all intermediate values associated with the same intermediate key. In general the master node of the Map function takes the input, divides the work into smaller sub problems and assigns it to worker nodes. This is done repeatedly forming multi-level tree structure. The reduce function combines the results of all the sub-problems and obtains the solution to the original problem. Thus the MapReduce framework transforms a list of (key, value) pairs into a list of values. Many real world tasks are expressible in this model.

The architecture of serial web crawler is shown in Figure 1. It is used for crawling through a whole site on the Inter/Intranet. We specify a start-URL and the crawler follows all links found in that HTML page. Web crawler begins