Map-Reduced Model for Topic Sensitive PageRanking

Dupinder Kaur
M. Tech. Scholar
Department of Computer Science & Applications
Chaudhary Devi Lal University, Sirsa(HR)

Abstract

Search engines are effectively used today to find the content on web as well as it can rank the fetched page(s) in a useful manner. When a user search for a content in the search engine, the search engine fetches the web pages from the database. Most search engines work by crawling the web pages and then they build an inverted index by listing all the words or other strings found in which page. When a search query is fired, the terms are searched in the inverted index and all the web pages which contains the search term. A TrustRank is calculated based on several factors .After fetching the web pages and calculating the TrustRank, it is matched with the Pageranking value the overall rank of a web page and the fetched pages are sorted and displayed according to their ranks. Hence Page-ranking plays an important role in case of a search engine. If we analyze the web a little; we can observe that it forms a graph where each node represents a web page and each edge represents one hyper link. More specially we can consider this graph to be directed. Considering these factors a map-reduce model is developed and which can be easily implemented in any Hadoop like environment. Map-Reduce Model has the advantages of parallel processing in a cluster and sparseness of the web matrix. In this paper, Topic sensitive PageRank is studied with map-reduce model and a comparison is made with iterative model.

Keywords: Crawler, Fetching, Map-Reduce, Pagerank, Trustrank

I. INTRODUCTION

PageRank is a method to assign real values to web pages in the Web (the part of web that has been crawled). The logic is simple, that the higher the PageRank of a web page, the more useful and important it is [1]. Each page fetched from the database of a search engine is passed through two computations: TrustRank and PageRank. TrustRank is calculated based on a lot of property of a web page, like how a search query matches with the content of a search engine, position of the search terms with in document and number of hits. After fetching the web pages and calculating the TrustRank, it is matched with the PageRank value to And the overall rank of a web page and the fetched pages are sorted and displayed according to their ranks. So a TrustRank tells us how much the web page is relevant to the search term and PageRank tells us how much the website is important in the web [2].

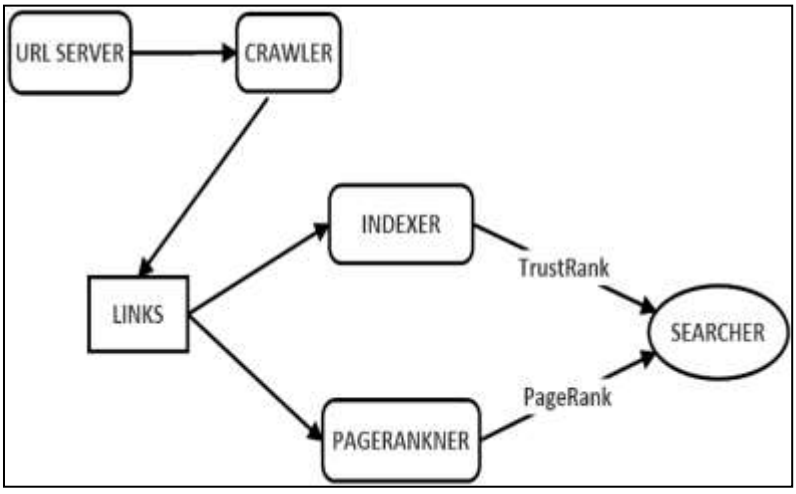

Fig. 1: Basic Design Blocks of a Search Engine

Google like search-engine uses 100s of properties to calculate the TrustRank of any web page. After the calculation of TrustRank, it is mixed with PageRank to calculate the overall rank of a web page as shown below in figure. Several changes and improvements can be done to PageRank. One is that we can give more priority to certain pages and rank them more because of their topic. Searching with the help of a search engine with pageranking works great, but sometimes same search query refers to

different interests. Hence topic sensitive Pageranking is introduced. The best example is searching python in the web. A search query python can refer to a programming language or sometimes it can be referred to a class of snake [3].

II. OBJECTIVES

Apart from pageranking, topic sensitive pageranking calculation is also termed as important as the title says, it computes biased PageRank values according to the topic of the web page. This paper designed a map reduce algorithm for topic sensitive page rank. Its objectives are:

- To simplify the PageRank calculations with the use of parallel processing.
- To increase the computational speed of search engine for deciding page rank.
- To make efficient data processing with parallel processing in clusters at huge data centers.

III. MAP REDUCE MODEL

Processing huge data set is a challenge; a standalone system cannot handle the computing requirement for that kind of computation. Memory and computing limitations in a single system drives us to use parallel system. But writing a simple program in old parallel system is a very tedious task. So Google came up with a very simplified programming model called map-reduce and that can be effectively used for batch processing of a huge dataset. Map-reduce is based on the concept of scaling out rather than scaling up. Adding more new system to a system is much cheaper than scaling up the performance of a system. This programming model works completely on a clustered or distributed environment [6]. User has to write a map and a reduce code and this code is executed in every single system in mapper phase and reducer phase in the distributed file system [4].

The whole dataset is not kept in a single centralized system, rather it is distributed in all the working system. Output of map data is written to the local disk and the reducer take data directly from the mapper. The dataset is represented as (Key,Value) [8] pair both in mapper and reducer.

A. Programming Model

1) Map (in-key, in-value) -> list(out-key, intermediate-value)
Processes input key/value pair
Produces set of intermediate pairs
2) Reduce (out-key, list(intermediate-value)) -> list(out-value)
Combines all intermediate values for a particular key
Produces a set of merged output values (usually just one) [7]

Web crawling is an important part for implementing pageranking algorithms. By crawling the web, we can build the web graph. Web crawler will encounter multiple links to the same document. To avoid downloading and processing same document multiple times, a URL-seen test is generally performed on each extracted link and based on the test it is decided whether to add the link to frontier or not. Generally MD5 or SHA1 hashing is used for that.

B. Resolving Relative URLs

Every crawled page has a base URL.A base URL is the consistent part of your web address. Fetch the base URL of the selected page and now join the base URL with the relative URL to and the absolute path. Relative URL path is the path of the web page inside the web server .After finding, web server is called. So during scraping we need to find the absolute path from the relative URL. And then only the page is added to the queue. Every URL has many parts.

When we want to implement pageranking algorithm most of the part of the fetched URLs are useless and hence we need to process the URLs to get the meaningful part of URLs only. Using any URLl-parser we can parse the URL and get required URL easily [5].

Scheme is generally the protocol used to address the web site or network. e.g. https, ftp, http etc.

Netloc is the network location or the web site address of the website. For example it is the server's base address of web sites. e.g www.google.com, www.yahoo.co.in

Path is the path of the web page inside the web server and the required webpage or web server is called. query is the part of the URL usually the terms used to send data to a web-page using GET request.

Fragment is used to address to a web page. it is used to address one object with in a webpage.

Assume 2 pages are there in www.example.com i.e. www.example.com/a.php and www.example.com/b.php. Now we perform 2 search queries in these web pages and passed with a GET query and move to a fragment of webpage. so assume 2 queries like

- www.example.comna:php?q=query1#m
- www.example.comnb:php?q=query2#q

These two refers to two different web-pages, but these makes these add same interest to the same web site. So keeping the search query and fragment doesn't make any sense to our PageRank algorithm. So post process it to remove the unwanted part of the URLs. We will get same www.example.com for both crawled web page. So PageRank will be accumulated for www.example.com only. This makes more sense to our pageranking algorithm. Each data scraped from the web has a format

source-link, dest-link stored in a raw file. Now we have to change into a required format for our pageranking algorithm. In order to change the format of such huge dataset we again one more map-reduce algorithms. This Mapper and reducer takes the parsed URLs and format it for the requirement input type of map reduce.

The proposed algorithm is designed as

C. *Algorithm for Mapper*

Require: key[comma_sep_urls],value[blank]
comma _sep_urls: src,dest

1) src,dest=comma_sep_urls.split(',')
2) emit(src,dest)
3) emit(dest,blank)

D. *Algorithm for Reducer*

Require: key[url],value[dest-urls]
Dest_urls: all the urls that are pointed by the current key url

1) pagerank=1.0
2) adj=”
3) for url in dest-urls do
4) adj.append(url+',')
5) end for
6) emit(url,[pagerank,adj])

IV. CONCLUSION

A spider is designed with SCRAPY and it's run against different URLs to fetch data and post processed. As mentioned above this process requires various web sites www.cet.edu.in,www.iiit-bh.ac.in and similar sites are crawled to collect the web graph. With the crawled data with around 1 lakh nodes iterative version will throw segmentation fault, but at the same time map reduce will produce result in a pseudo distributed ubuntu system. with all these testing map reduce version of the topic sensitive pageranking algorithm is proved to be correct and hence with the feature of map-reduce it can be easily scaled out. Figure 2 clearly shows that number of nodes is reduced with map-based model in topic sensitive page rank.

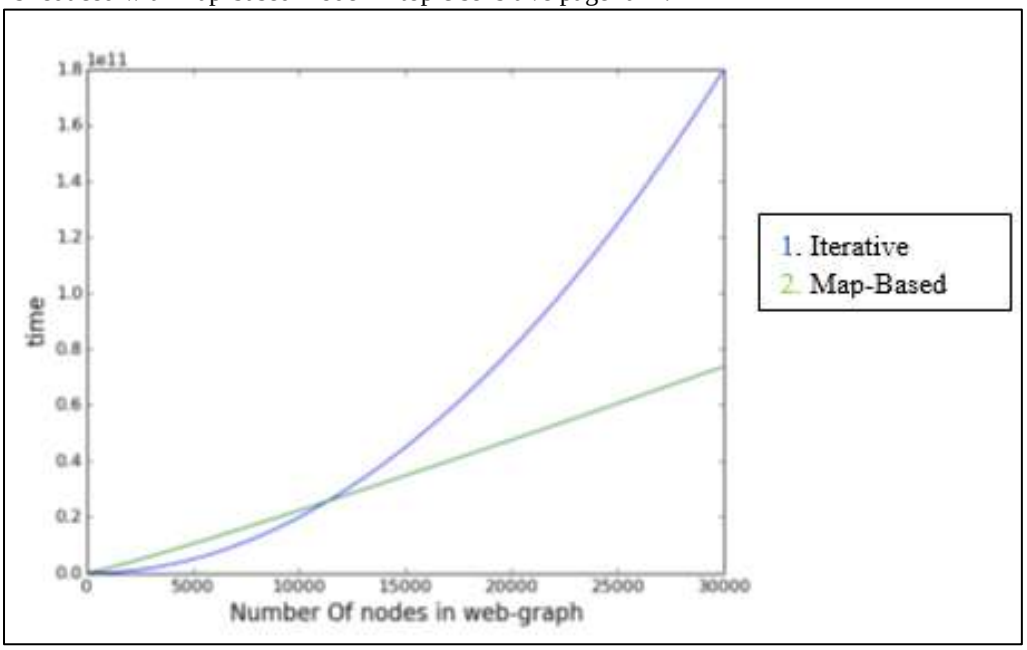

Fig. 2: Time Comparison Iterative Vs Map-Reduced

Based on the intuition of topic and priority sensitive pageranking a smaller change to the initial pagerank is studied. Effective calculation of pagerank is performed with map-reduce in pseudo distributed mode in ubuntu system. The Map-reduce topic sensitive pageranking is proposed and hence implemented for the scraped data. The equation can be further modified for better accuracy and performance.

REFERENCES

[1] Lawrence Page, Sergey Brin, Rajeev Motwani, and Terry Winograd, "The pagerank citation ranking: Bringing order to the web".
[2] Andrei Broder, Ravi Kumar, Farzin Maghoul, Prabhakar Raghavan, "Graph structure in the web", Computer networks, vol:33(1), p.p:309-320.
[3] Taher H Haveliwala, "Topic-sensitive pagerank" in Proceedings of the 11th international conference on World Wide Web, p.p: 517-526. ACM.
[4] Youcef Saad and Martin H Schultz. Gmres , "A generalized minimal residual algorithm for solving nonsymmetric linear systems", SIAM Journal on scientific and statistical computing, vol:7(3), p.p:856-869.
[5] Taher Haveliwala, Sepandar Kamvar, Dan Klein, Chris Manning, and Gene Golub, "Computing pagerank using power extrapolation", in Stanford University Technical Report, 2003.
[6] Nan Gong. "Using map-reduce for large scale analysis of graph-based data". 2011.
[7] Dean and Sanjay Ghemawat, "Map-reduce: simplified data processing on large clusters", Communications of the ACM, vol. 51, p.p:107-113, 2008.
[8] Dean and Sanjay Ghemawat, "Distributed programming with map-reduce", Beautiful Code Sebastopol: OReilly Media, Inc, p.p:384, 2007.