ISSN 2229-5518

Web Search Engine

G.Hanumantha Rao*, G.NarenderΨ, B.Srinivasa Rao+, M.Srilatha*

Abstract—This paper explains how to make the process of searching in the search engine very efficient and more accurate and also how to reduce the number of unwanted search results in the searching process. They answer tens of millions of queries every day. Search engines index tens to hundreds of millions of web pages involving a comparable number of distinct terms. Instead of searching the keywords in total indexes present in the indexer if we reduce the number of search indexes by providing some conditions then we can reduce the number of search results. In this paper we present details of how to reduce the number of indexes that are not exactly suitable for our query that contain the number of keywords.

Index Terms— Search Engine, Index, Web page and keyword.

1 INTRODUCTION

Search engines are one tool used to answer information needs. Search engines are huge databases of web pages as well as software packages [1] for indexing and retrieving the pages that enable users to find information of interest to them. The search engines directories, portals and indexes are the web's "Catalogues" allowing a user to carry out the task of searching the web for information that he/she requires. Users express their information needs as queries. Usually informally expressed as two or three words (we call this a ranked query). Around 48.4% of users submit just one query in a session, 20.8% submit two, and about 31% submit three or more. Less than 5% of queries use Boolean operators (AND, OR, and NOT), and around 5% contain quoted phrases. The Web searches the Web-based content.

The web creates new challenges for information retrieval. The amount of information on the web is growing rapidly as well as the number of new users. The search engines that rely on keywords matching usually return too many, low quality matches. To make matter worse and high quality we have to build a large scale search engines which address many of the existing system.

2 METHODOLOGY

Normally the search engine data bases of web pages are built and update automatically by Web crawlers. When one searches the web using one of the search engines, one is not searching the entire web. Instead one is only searching the database that has been compiled by the search engine and this database.

* Assistant Professor, Department of CSE, Vasavi College of Engineering, Hyderabad-500031, INDIA.

Ψ Associate Professor, Department of CSE, Keshav Memorial Institute of Technology, Hyderbad,INDIA.

+ Assistant Professor, Department of IT, Gudlavalleru Engineering College,Gudlavelleru, Krishna-521356, INDIA.

hanu.abc@gmail.com, guggillanarender@gmail.com, bskdata@gmail.com, srilatha.manam@gmail.com

2.1 Search Engine Architecture

Typical search engine architecture consists of many components including the following three [4] major components.

1. The crawler and the indexer: It collects pages from the web, creates and maintains the index.
2. The user interface: It allows submitting queries and enables result presentation.
3. The database and the query server: It stores information about the Web pages [8] and processes the query and returns results.

All the search engines essentially include a crawler, an indexer and a query server although the algorithms used in these components and quality of the algorithms may vary significantly.

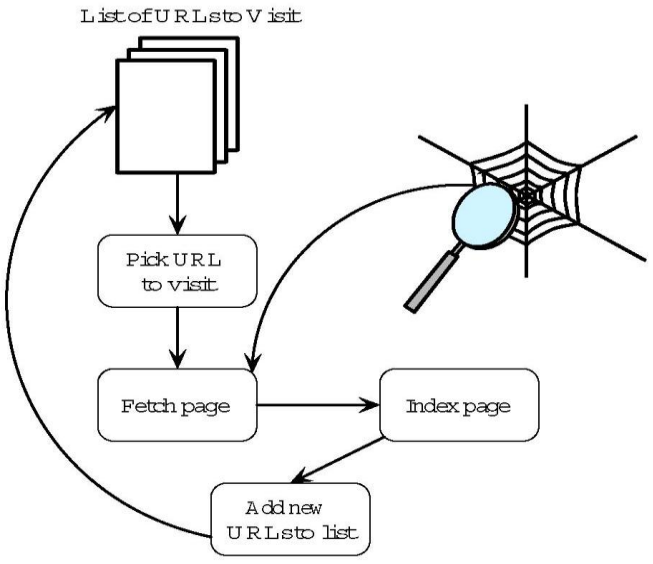

Fig. 1. Crawling the Web

2.2 The Crawler

The crawler is an application program that carries out a task similar to graph traversal. It is given a set of starting URLs that it uses to automatically traverse the Web by re-

trieving a page, initially from the starting set. Since a crawler has a relatively simple task, it is not CPU-bound. It is bandwidth bound. In crawling, bandwidth can become a bottleneck.

A Web crawler [3] starts with a given set of URLs and fetches those pages. This continues until no new pages are found or a threshold is reached. While the crawler if fetching pages new pages are being put o the web and old pages are deleted or being modified. Therefore, a crawler could potentially continue finding new or modified pages forever.
Crawlers follow an algorithm like the following:

- Find base URLs: a set of know and working hyperlinks[9] are collected.
- Build a queue: put the base URLs in the queue and add new URLs to the queue as more are discovered.
- Retrieve the next page: retrieve the next page in the queue, process and store in the search engine database.
- Add to the queue: check if the out lines of the current page have already been processed.
- Continue the process until some stopping criteria are met.

2.3 Indexer

An index [1] is essential to reduce the cost of query evaluation. These indexes can be built while the crawler is collecting the web pages or this could be done later. Indexing can be a very resource intensive process and often CPU-bound since it involves a lot analysis of each page.

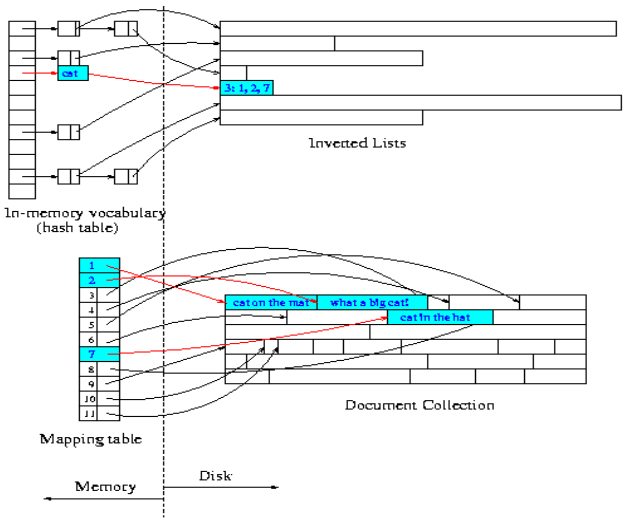

Fig. 2. Indexing Data

Building an index requires document analysis and term extraction. Term extraction may involve extracting all the words from each page, elimination of stop words (common words like "the", "it", "and", "that") and stemming (transforming words like "computer", "computing" and "computation" into one word, say "computer", since there is little point in treating them as different words and similarly transforming "find" and "found" into one word) to help modify, for example, the inverted file structure. It may also involve analysis of hyperlinks.

2.4 Query Server

First of all, a search engine needs to receive the query and check the spelling of keywords that the user has typed. If the search engine cannot recognize the keywords as words in the language or proper nouns it is desirable to suggest alternative spelling to the user. Once the keywords are found to be acceptable, the query [4] may to be transformed. Queries are resolved using the inverted index. Consider the example query "Cat Mat Hat". This is evaluated as follows:

- Select a word from the query (say, "Cat")
- Retrieve the inverted list from disk for the word
- Process the list. For each document the word occurs in, add weight to an accumulator for that document based on the TF, IDF, and document length
- Repeat for each word in the query
- Find the best-ranked documents with the highest weights
- Lookup the document in the mapping table
- Retrieve and summarize the documents, and present to the user

A user has to submit a query a number of times using somewhat different keywords before more or less the "right" result is obtained. A search engine providing query refinement based on used feedback would be useful. Search engines often cache the results of a query and can then use the cached results if the refined query is a modification of a query that has already been processed.

In database system, query processing requires that attribute values match exactly the values provided in the query. In search engine query processing, an exact match is not always necessary. A partial match or a fuzzy match may be sufficient. Essentially a cache is supposed to cache a user's or client's request, get the page that the user has requested if one is not available in the cache and then save a copy of it in the cache.

2.5 Search Query Processing

In search, true success comes from understanding what the user is asking from their query. Some user queries are simply stated, while others are stated in a Boolean format ("apples AND oranges OR bananas"), or presented as whole paragraphs, passages, or documents with a request to "find similar" information. So the search platform must

have a range of tools in order to accurately understand what is being asked.

The challenge with information retrieval revolves around two basic problems: 1) getting a good query from search users with the aim of helping them craft better questions, and 2) presenting "easy-to-judge" results to minimize what the user has to read through. In general, queries from the user come into the query processing and transformation [5] subsystem. This framework takes the original query, analyzes it, transforms it with, corrections of spelling mistakes and then sends the query to the search engine.

The fig.3 shows the elements of query and results processing. The node in the search matrix that receives the query performs its retrieval operation and returns its results to the results-processing subsystem. The raw results are passed to the results-processing subsystem (which performs duplicate removal), results merging (from different search nodes), sorting, rank ordering, etc. All results are then sent to the search user.

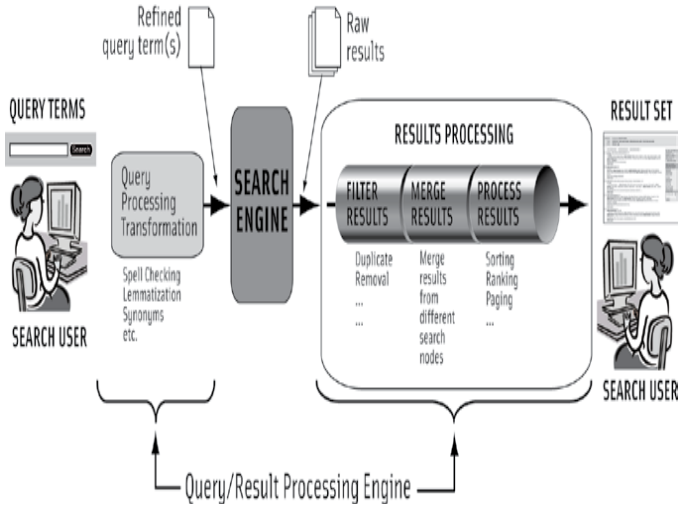

Fig. 3. Elements of Query and Result Processing

3 Technical Details

The Web crawling (downloading the web pages) is done by several distributed crawlers. There is a URL server that sends lists of URLs to be fetched to the crawlers. The web pages that are fetched are then sent to the storeserver.

The storeserver then compresses and stores the web pages into a repository. Every web page has an associated ID number called a docID which is assigned whenever a new URL is parsed out of a web page. The indexing function is performed by the indexer and the sorter. The indexer performs a number of functions.

It reads the repository, uncompressed the document, and parses them. Each document is converted into a set of word occurrences called hits. The hits record the word position in document, an approximation of font size, and capitalization. The indexer distributes these hits into a set of "barrels", creating a partially sorted forward index. The indexer performs another important function. It parses out all the links in every web page and stores important information about them in an anchors file. This file contains enough information to determine where each link points from and to, and the text of the link. The URLresolver reads the anchors file and converts relative URLs into absolute URLs and in turn into docID. It puts the anchor text into the forward index, associated with the docID what the forward index, associated with docID that the anchor points to. It also generates a database of links which are pairs of docID. The links database is used to compute page ranks [7] for all the documents. The sorter takes the barrels, which are sorted by docID, and resorts them by worded to generate the inverted index. This is done place so that little temporary space is needed for this operation. The sorter also produces a list of wordIDs and offsets into the inverted index.

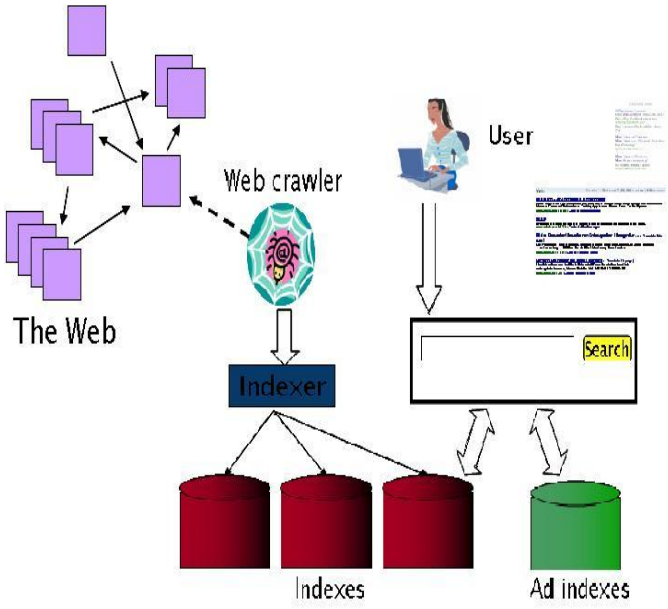

Fig. 4. Web Search Basics

A program called dump Lexicon takes this list together with the lexicon produced by the indexer and generates a new lexicon produced by the indexer and generates a new lexicon to be used by the user the lexicon built by Dump lexicon together with the inverted index and the Page Ranks[4] to answer queries.

4 Problem Overview

The major problem with search engine [10] is that of abundance. Users are very impatient. They demand immediate results otherwise they abandon the search and move on to something else. Some times a search may return a very large number of results making it difficult to find the most suitable resources. Users rarely read more than one or two screens of results returned by a search engine even

when the query they have posed is short and ambiguous and is not carefully thought out. More and more users are looking to search engines for all kinds of information including the latest news.

Generally we think what is a good answer to a query [6]?

- The good answer to a query is one that is relevant to the user's information need.
- Search engines typically return ten answers-per-page, where each answer is a short summary of a web document.
- Likely relevance to an information need is approximated by statistical similarity between web documents and the query.
- Users favor search engines that have high precision, that is, those that return relevant answers in the first page of results.

5 SOLUTIONS

Generally in the searching process of a query that is entered by the user is done by considering the each and every word in the query as a keyword. Then it will compare the all the keywords present in the query with the indexes of a web pages present in the indexer.

Instead searching the all indexes entirely for all keywords we have avoid some indexes or part of indexes of a web pages then we can reduce the number of results of a user query. This can be done in the following way.

In the most of cases the query may contain more than two keywords. Generally the first keyword in a query is searched first and than second keyword and so on. If the first keyword found in a particular position in the index of a particular webpage then the second keyword id searched in the right side part of an index of that webpage in which the first key word is found. If at all the second keyword is present in the right part then only the searching of another keyword is continued in same way otherwise the searching of keywords in that particular index is skipped and the web page will not be displayed in the search results of a user query.

Simply if the query contains n number of keywords then the ith keyword must be searched in right side part of the index present in the indexer where the (i-1)th keyword is found.

5.1 Experimental Result

It is written entirely in Java. The search engine can run on any machine that has stable Java virtual machine. It is an easy to understand and powerful language, it is object oriented for even greater extensibility. It is very small just over 100K including source code, binaries and configuration files at last count. It index are very small they are on the order of 10% of the size of the text on the sight even though the index every single alpha numeric word.We have start the web crawler to identify the keywords in the stored web pages and it is represented in the following figure 5.

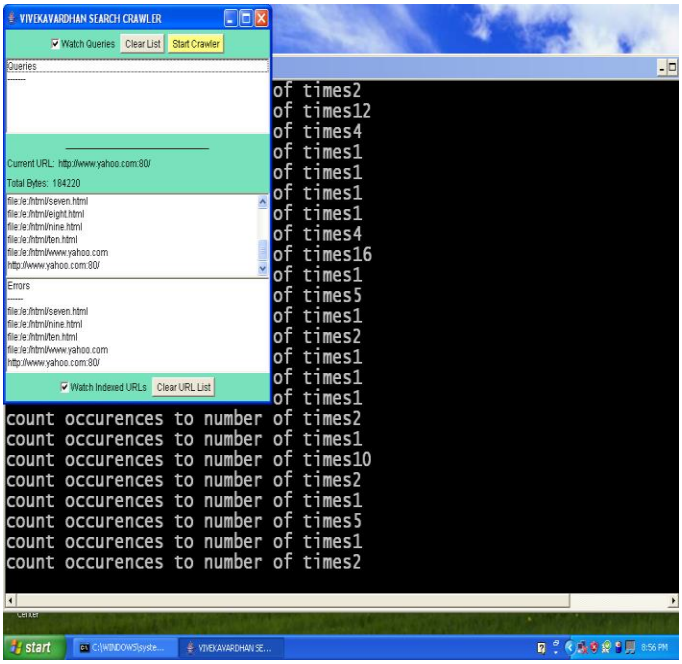

Fig 5: Starting the web crawler

OPEN the browser and select the search.html to do the searching, if the keyword is available then it displays in which files the data is present and with no. of times it is present.The following figure 6 shows for searched keyword "computer" isshown.

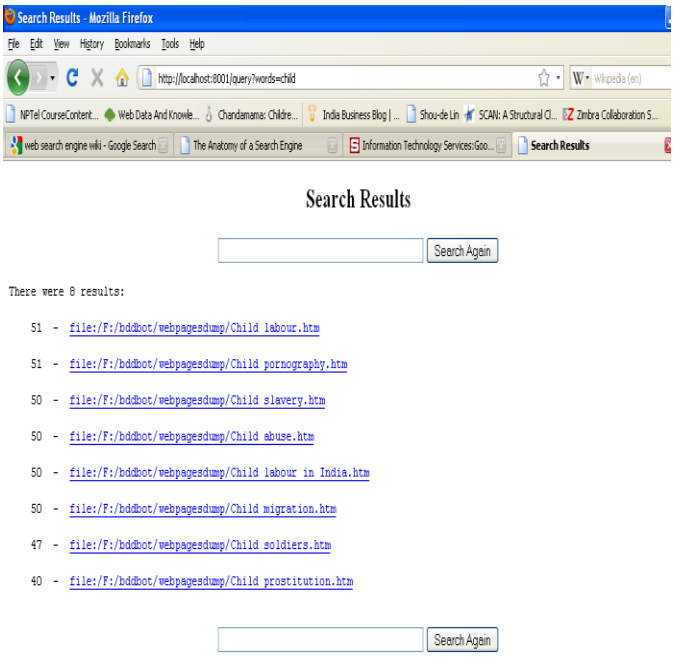

Fig 6. Results of "computer" keyword

Suppose if the keyword is not present in the web pages.It will dislay as no results found.The following figure 7 shows the result for the keyword "hauyt".

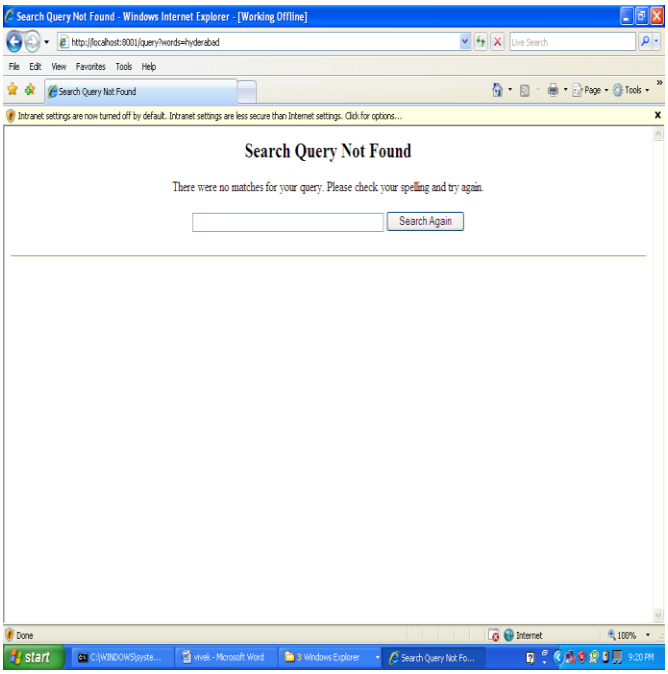

Fig 7. Results of "hauyt" keyword

6 Conclusion

The conclusion is made that if we are giving the query that contain more then one keywords than it is possible to reduce the number of search results that given by the search engine. That means it returns the web pages which are contain all the keywords present in a query in the same order as like in the query. According to the solution mentioned above if we are entered the query as Indian citizen' then it will display the web addresses which contain Indian citizen but it will not display the web pages which contain citizen of India. It is not possible to reduce the number of pages in the output of a query which we are given to the search engine if it contains only one keyword.

References

[1] Google Search Engine http://google.stanford.edu/
[2] Search Engine Watch http://www.searchenginewatch.com/
[3] SearchEnginesWork Shops http://www.searchengineworkshops.com/
[4] Brin, S. and L. Page, 1998, the anatomy of a large scale hypertextual Web search engine, Proc. 7th World Wide Web Conference
[5] V o Ngoc Anh and Alistair Moffat. Improved retrieval effectiveness through impacttransformation.I n *Proceedings of the Thirteenth Australasian Database Conference*, Melbourne, Australia, in press.
[6] Broder, A. (2002): taxonomy of web search. SIGIR Forum 36(2).http://www.acm.org/sigir/forum/F2002/broder.pdf [22.8.2005]
[7] Page, L., Brin, S., Motwani, R., Winograd, T. (1998): The PageRank citation ranking: Bringing order to the Web. http://dbpubs.stanford.edu:8090/ pub/1999-66 [22.8.2005]
[8] P.Faraday."Attending to web pages" CHI2001 Extended Abstracts (poster), 2001, pages 159-160.
[9] N.Craswell, D.Hawking and S.Robertson,"Effective site finding using Link Anchor Information" In proceeding of the ACM SIGIR conference on research and development in information retrieval, 2001.
[10] Singhal, Amit (2004): Challenges in Running a Commercial Search Engine. http://www.research.ibm.com/haifa/Workshops/searchand-collaboration2004/papers/haifa.pdf