International Journal of Computer and Communication Technology

Volume 2 | Issue 4 Article 10

October 2011

An Alternate Idea for Storage Optimization in Search Engine

Anirban Kundu
1Netaji Subhash Engineering College, West Bengal University of Technology, Calcutta 700152, India, anik76in@gmail.com

Siddhartha Sett
1Netaji Subhash Engineering College, West Bengal University of Technology, Calcutta 700152, India, sett.siddhartha@gmail.com

Subhajit Kumar
1Netaji Subhash Engineering College, West Bengal University of Technology, Calcutta 700152, India, papansk@gmail.com

Shruti Sengupta
1Netaji Subhash Engineering College, West Bengal University of Technology, Calcutta 700152, India, shruti.nsec@gmail.com

Srayan Chaudhury
1Netaji Subhash Engineering College, West Bengal University of Technology, Calcutta 700152, India, srayan.manutd@gmail.com

An Alternate Idea for Storage Optimization in Search Engine

Anirban Kundu[1,2], Siddhartha Sett[1,2], Subhajit Kumar[1,2], Shruti Sengupta[1,2], Srayan Chaudhury[1,2]

[1]Netaji Subhash Engineering College, West Bengal University of Technology,
Calcutta 700152, India
[2]Innovation Research Lab (IRL), Capex Technologies, West Bengal 711103, India
{anik76in, sett.siddhartha, papansk, shruti.nsec, srayan.manutd}@gmail.com

Abstract - **We propose an alternate indexing storage technique of Search Engine. In this approach, we achieve reduced space complexity. We try to decrease time complexity for faster data retrieval and decrease storage space for efficient utilization of space. This paper provides an algorithm of indexing mechanism by which effective storage space is reduced. Space complexity of a Search Engine depends on the indices created by the indexing algorithm. Hamming distance concept is used to achieve better result.**

Keywords: Search Engine, Forward indexing, Inverted indexing, Hamming distance.

I. INTRODUCTION

A Search Engine searches information on WWW [1-2]. A history on evolution of Search Engine [3-4] is briefly described. Before 1992, there was a complete list of all Web servers which was edited by Tim Berners-Lee and hosted on the CERN Web servers. The first tool used for searching on the Internet was a tool named 'ARCHIE' invented by 'Alan Emtage' in the year 1990. Two new search tools 'VERONICA' and 'JUGHEAD' came into existence in the year 1991. Mathew Gray invented a PERL based tool named 'World Wide Web Wanderer' in 1993. The first Web Search Engine 'ALIWEB' was also invented in 1993. 'JUMPSTATION' was the first tool to combine the three essential features of a Web Search Engine that is crawling, indexing, and searching. In the year 1994, a full text crawler based Search Engine named 'WebCrawler' came into existence. Unlike its predecessors, it late the users search for any word in any Web page. It was the first tool to be widely used by the public. Also in the year 1994, 'Lycos' was launched. It became a major commercial success. In the recent few years, many Search Engines have appeared and have gained popularity. These included 'Magellan', 'Excite', 'Infoseek', 'Inktomi', 'Northern Light', 'Altavista', 'Yahoo'. In 1996, 'Netscape', a popular browser struck deals with five major Search Engines. The deal was to feature these Search Engines on the Netscape Search Engine page. These Search Engines were 'Yahoo', 'Magellan', 'Lycos', 'Infoseek', and 'Excite'. The Web started out as a dispersed elucidation for accessing through the WWW documents hosted by various organizations on their servers. The most revolutionary feature of the Web is that it allows the documents to contain hyperlinks pointing to other documents on a server somewhere on the Internet [5-7]. These hyperlinks form a global Web of connected documents hence the name "World Wide Web". Users submit data to the servers, corresponding Web pages are being generated, and rich form of interaction between the user and the Web client are supported. The information present on the WWW is accessed through Search Engine. A Search Engine searches information from various databases. The searching done by the Search Engine is based on the indexing mechanism of the Search Engine. Indices are of two types, forward index and inverted index. In typical Search Engines forward indices are created by extracting all the words from the Web pages and storing them with respect to Web page number. In existing Search Engines inverted indices are created by storing every word with corresponding Web page numbers.

Typically, in a Search Engine forward indices are created by extracting all the words from Web pages and storing them with respect to Web page number as referred in Table 1.

Table 1: Typical Forward Indexing

Html Pages	Words
1.html	Ant(5), Bat(9), Dog(7)
2.html	Girl(12), Bat(2)
3.html	Girl(5), Ant(2), Bat(10)

In a typical Search Engine, inverted indices are created by storing every word with corresponding Web pages. Each and every word is stored in inverted index as referred in Table 2.

Table 2: Typical Inverted Indexing

Words	HTML Pages
Ant	1.html, 3.html
Bat	1.html, 2.html, 3html
Dog	1.html
Girl	2.html, 3.html

The function of indexing algorithm is to collect, parse and store data to facilitate fast and accurate information. Popular Search Engines focus on the indexing of full text. Partial text services restrict the depth indexed to reduce index size which is in contrast to full text indices [3, 8].

Organization of rest of the paper is as follows: Proposed work has been described in Section II. Experimental results

have been shown in Section III. Conclusion has been depicted in Section IV.

II. PROPOSED WORK

Web pages are first downloaded from Internet. Forward indexing has been done on downloaded Web pages for maintaining record of words as shown in Table 1. After that, the words are reorganized based on occurrence as shown in Table 3. Hamming distance concept is then utilized to minimize the storage space of forward indexing. Inverted indexing is then carried out in typical way as shown in Table 2. On the other hand, predefined dictionary is exploited for maintaining mapping between the synonymous words. So, advanced inverted indexing is done using lexicon and inverted indexing file based on the occurrence on the Web pages as shown in Table 4. Figure 1 shows the overall view of our proposed system.

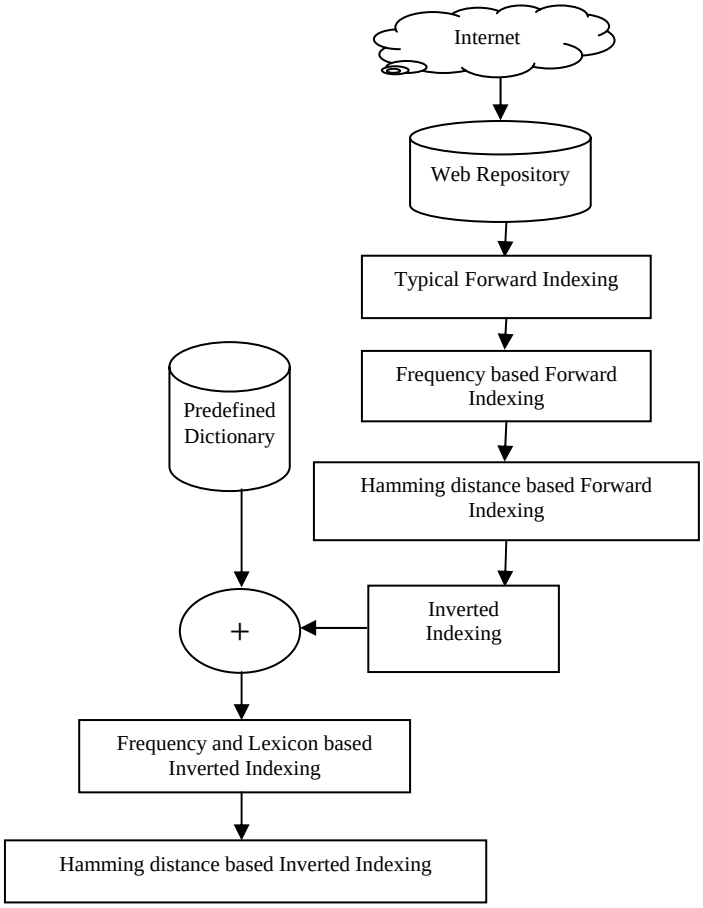

Figure 1: Proposed approach using Hamming distance

A hypothesis on indexing method of Search Engine has been prepared using Hamming distance which helps in decreasing space requirement. As a result, time and space requirement have been reduced using our methodology. In this approach, textual matters are only handled. Thus, the alphabets ranging from 'A' to 'Z' are used for Hamming calculation. It is calculated that the maximum Hamming distance would be '4' between the English alphabets, if any alphabet is chosen as baseline as shown in Table 5 and Table 6 later.

The architecture of a Search Engine has a part called lexicon which stores a dictionary. The dictionary is used for checking the meanings and spellings of keywords which are entered by the user for searching. The idea is to introduce a table of synonyms which has been introduced in the dictionary. If the Search Engine comes across a word which has one or more synonyms, it treats all the synonymous words as a single word and thus reduces the redundancy of storing the same meaning words.

In typical Search Engines forward indices are created by extracting all the words from Web pages and storing them with respect to Web page numbers as shown in Table 1. Then, the words are reorganized using the number of occurrence of words. Table 3 is the modified version of Table 1.

Table3: Occurrence based Forward Indexing

Html Pages	Words
1.html	Bat(9), Dog(7), Ant(5)
2.html	Girl(12), Bat(2)
3.html	Bat(10), Girl(5), Ant(2)

An occurrence based technique is used for creating the inverted index as shown in Table 4. The technique is to store each word present on Web pages based on frequency of words in a decreasing manner. Table 4 is the advanced form of Table 2.

Table 4: Occurrence based Inverted Indexing

Words	HTML Pages
Bat	1.html, 2.html, 3.html
Ant	1.html, 3html
Girl	2.html, 3.html
Dog	1.html

We have applied concept of Hamming distance on the words having three letters to at most eight letters. The Hamming distance between two strings of equal length is the number of positions at which the corresponding symbols are different. It measures the minimum number of substitutions required to change one string into another. The words begin with any letter from the alphabet set 'A' to 'Z'. For three letter words 26 stacks have been created for each letter in the alphabet set. For each stack, a reference value has been stored with only the beginning letter changing. For example, the letter 'A' has a stack with the reference value 'AAA'; letter 'B' has a stack with the reference value 'BAA'. Similarly, all the letters have individual reference values. In similar way, we have created such stacks for four letters, five letter words up to eight letter words. A letter is represented by its 'ASCII' value, for storing a letter on memory using 8 bits in typical case. But, in our approach, a letter is stored using 3 bits applying the Hamming distance concept in both the cases of forward and inverted indexing. Thus, five bits are saved for storing a single character. Table 5 shows the mapping between the alphabets and its corresponding 'ASCII' values in binary format. It is observed that the 6^{th}, 7^{th} and 8^{th} bit positions of the alphabets are fixed. Thus, effectively five bits are essential for Hamming distance calculation.

Table 5: Hamming distance of English alphabet ('A' to 'Z') with respect to 'A'

Letter	8 bit ASCII value	Effective 5 bit ASCII value	Hamming distance
A	01000001	00001	0
B	01000010	00010	2
C	01000011	00011	1
D	01000100	00100	2
E	01000101	00101	1
F	01000110	00110	3
G	01000111	00111	2
H	01001000	01000	2
I	01001001	01001	1
J	01001010	01010	3
K	01001011	01011	2
L	01001100	01100	3
M	01001101	01101	2
N	01001110	01110	4
O	01001111	01111	3
P	01010000	10000	2
Q	01010001	10001	1
R	01010010	10010	3
S	01010011	10011	2
T	01010100	10100	3
U	01010101	10101	2
V	01010110	10110	4
W	01010111	10111	3
X	01011000	11000	3
Y	01011001	11001	2
Z	01011010	11010	4

Table 5 shows the Hamming distance values of different English alphabets with respect to 'A'. Maximum '4' Hamming distance is found using effective 5 bits (0 to 4). Thus, only three bits are required for storing the effective position changes using the Hamming distance having maximum value '4' which can be represented by three bits (100) using binary convention.

The words found on Web pages have been arranged using several tables. Tables are formed in accordance with number of letters containing the words; e.g., 3 letter word table, 4 letter word table, etc.. Generally 1 and 2 letter words are articles, prepositions, stop words, etc.. So, the tables are of 3 letters, 4 letters, ..., 8 letters. Each table is

further divided into 26 stacks. These stacks are designed to find Hamming distance of any letter from 'A' to 'Z' according to the initial letter of the particular word. So, there are total 6 (3 to 8) tables and each table has 26 stacks ('A' to 'Z'). It is being observed that words having more than eight (8) characters are very rare in practical cases. So, these types of words having more than eight characters are stored in typical manner.

III. EXPERIMENTAL RESULTS

The application of our proposed indexing mechanism is shown in this section. Millions of Web pages are downloaded and then parsed for forward and inverted indexing as mentioned in Section 2.

Experiment has been conducted for 3 to 8 letter words; because 1 and 2 letter words (except stop-words) are very rare in English language. So, the implementation of Hamming distance is only for 3 to 8 letter words. The indexing storage system for 1, 2 and more than 8 letter words is as same as typical Search Engines.

A comparative study has been done between our approach and typical Search Engine indexing storage system. A graph has been shown in Figure 2.

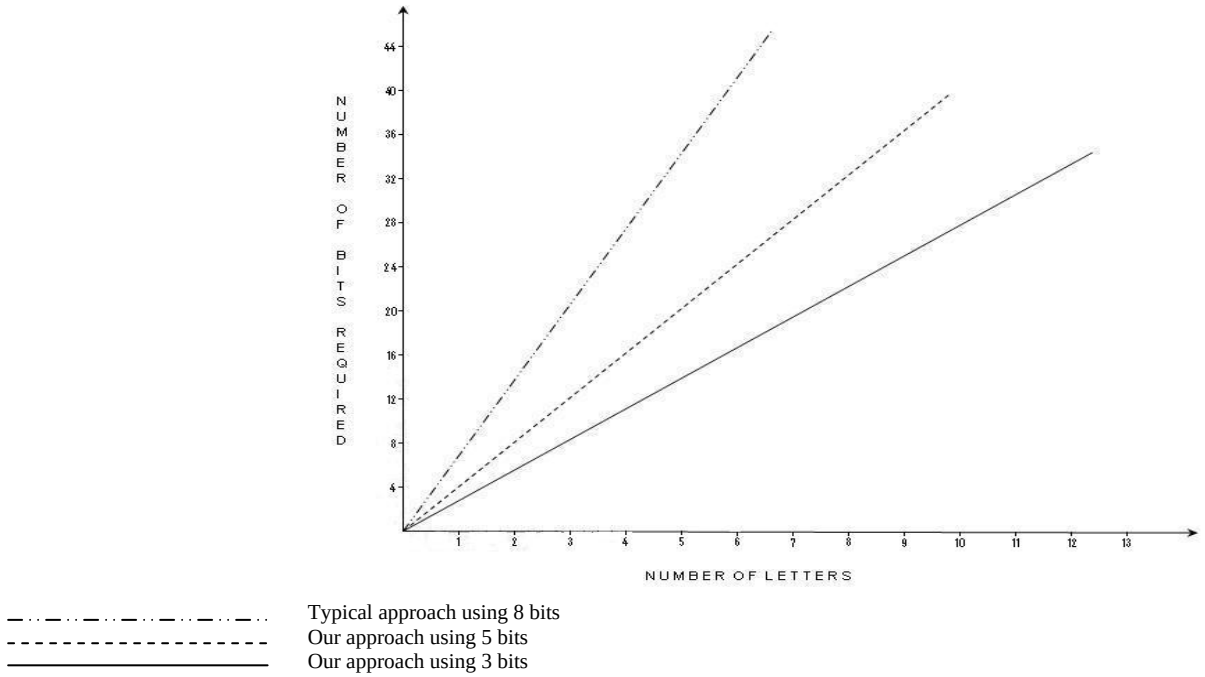

Figure 2: Comparative study between typical and our approach

IV. CONCLUSION

The indexing assumption has been fruitfully implemented and has achieved excellent outcome. Our scheme escorts to diminution in the space required to accumulate the indices. Huge number of words can be stored using smaller memory space. Identical words have been deleted. Thus, less time would be required to retrieve data. The construction is designed in such a way so that it leads to efficient utilization of space. Hamming distance between the characters of words has been exploited for minimizing the data storage of indexing part of Search Engine.

REFERENCES

[1] Arasu, A., Cho, J., Garcia-Molina, H., Paepcke, A., Raghavan, S., "Searching the Web", ACM Transactions on Internet Technology, Volume 1, Issue 1, August 2001.

[2] Kobayashi, M., Takeda, K., "Information retrieval on the web", ACM Computing Surveys (ACM Press) 32 (2), 2000.

[3] Brin, S., Page, L., "The Anatomy of a Large-Scale Hypertextual Web Search Engine", Proceedings of the Seventh International World Wide Web Conference, Brisbane, Australia, April 1998.

[4] Risvik, K. M., Michelsen, R., "Search Engines and Web Dynamics", Computer Networks, Volume 39, June 2002.

[5] Glover, E. J., Tsioutsiouliklis, K., Lawrence, S., Pennock, D. M., Flake, G. W., "Using Web Structure for Classifying and Describing Web Pages", WWW2002, Honolulu, Hawaii, USA, May 2002.

[6] Andreessen, M., Bina, E., "NCSA Mosaic: A Global Hypermedia System", Internet Research: Electronic Networking Applications and Policy 4 (1), 1994.

[7] Lawrence, S., Giles, C. L., "Accessibility of information on the web", Nature 400, 1999.

[8] Clarke, C., Cormack, G., "Dynamic Inverted Indexes for a Distributed Full-Text Retrieval System", TechRep MT-95-01, University of Waterloo, February 1995.