International Journal of Innovative Computing, Information and Control
Volume **9**, Number **2**, February **2013**

ICIC International ©2013 ISSN 1349-4198
pp. **741–751**

INFORMATION RETRIEVAL: COMPUTATION CONTROL FOR POINTER ADDRESS

Hameed Ullah Khan

Department of Information Systems
College of Computer and Information Sciences
King Saud University
Riyadh 11451, Kingdom of Saudi Arabia
hukhanafridi57@yahoo.com; hkhan@ksu.edu.sa

Received December 2011; revised May 2012

Abstract. *The magnitude of data archiving and recovery is well known to both researchers and users. Recent advances in information technology have made it possible to store large amounts of data and have made document repossession more complex, therefore constituting the need for finding efficient data retrieval implementations. As the scopes and horizons for information technology began to establish new boundaries, long-standing data storage modules recognized the need to widen their scale. Traditional systems available for data retrieval amalgamated with newer methods resulting in several innovative algorithms. These systems, set to function according to their respective requirements and hardware demand, have allowed for effectual and prompt data retrieval.*
Keywords: Databases search, Hashing, Information retrieval

1. Introduction.

The requirement to accumulate and retrieve inscribed information has become increasingly important over centuries. The invention of personal computers transformed mankind's approach to the accumulation and management of penned knowledge, forevermore revolutionizing the way humans conserve texts. Searching archives of text can be achieved these days due to the availability of concrete descriptions from novel approaches. Several works emerged that elaborated upon the cornerstone idea of searching text with a computer. One of the most influential methods involved using words as indexing units for documents and measuring word overlap as a criterion for information retrieval (IR).

Other systems have also been used in the field, of which most notable were the development of the SMART systems; tests developed to gauge an evaluation methodology for retrieval systems that is still in use by IR systems today. The SMART system, on the other hand, permitted researchers to experiment with newer innovations to convalesce upon existing search engines. A system for experimentation coupled with good evaluation methodology allowed rapid progress in the field, and paved way for many critical developments [1].

Numerous progresses were reported and various simulations for document retrieval were developed and these new reproductions were experimentally proven to be effective on small text collections (several thousand articles) available to researchers at the stage. However, due to lack of substantial text collections, the question of whether these models and techniques would scale to larger corpora remained unanswered. This changed with the inception of (TREC) Text Retrieval Conference; TREC is a series of evaluation conferences sponsored by various US Government agencies under the auspices of NIST (National

Institute for Science and Technology), which aims at encouraging research in IR from large text collections [2].

Dated systems were modified and many new methods were developed for effective retrieval over sizable data collections. TREC has also branched IR into related but important fields like retrieval of spoken information, non-English language retrieval, information filtering, and user interactions with a retrieval system. The algorithms developed in IR were the first ones to be employed for searching the (WWW) World Wide Web from 1996 to 1998. However, web search, matured into systems that took advantage of the cross linkage available on the web by describing the evolution of modern textual IR systems [3,4].

In this paper, Section 2 provides the background study, and many research capacities are merging together in the IR. In Section 3, the theoretical findings from the proposed approach have been discussed in detail. Section 4 provides implementation methodology. The simulation results have been carried out along with other algorithms in Section 5 and Section 6 provides conclusions.

2. Background Study.

Due to extensive theory on the subject, this section is divided into three sub-sections namely early approaches, mid-term and recent approaches in this technology which altogether link the past and present technologies.

2.1. Early approaches.

This section discusses several initial approaches as well as the systems used for the most part in the data retrieval process.

2.1.1. *Vector space model (I, II, III).* In the three vector space model texts are represented by a vector of terms. The definition of a term is not inherent in the model, but terms are typically words and phrases.

The similarity between two vectors if $\vec{D}$ is the document vector and $\vec{Q}$ is the query vector, then the similarity of document D to query Q (or score of D for Q) can be represented as:

$$\mathrm{Sim}(\vec{D},\vec{Q}) = \sum\nolimits_{t_i \in Q,D} \omega_{t_{iQ}} \cdot \omega_{t_{iD}} \tag{1}$$

In Equation (1), $\omega_{t_{iQ}}$ is the value of the i^{th} component in the query vector $\vec{Q}$, and $\omega_{t_{iD}}$ is the i^{th} component in the document vector $\vec{D}$. From $\omega_{t_{iQ}} \cdot \omega_{t_{iD}}$ is quite critical to search effectiveness of an IR system. The $\omega_{t_{iD}}$ is often referred to as the weight of term-i in document D. For more details see reference [5].

2.1.2. *Probabilistic models.* In probabilistic IR models the probability of relevance of documents for a query is estimated. The probability of relevance for document D is presented by $P(R|D)$. Assuming that, $P(R)$ and $P(\overline{R})$ are just scaling factors for the final document and can be removed from the formulation. This further simplifies the formulation to:

$$\log \frac{P(D|R)}{P(D|\overline{R})} \tag{2}$$

In Equation (2), $P(D|R)$ is a probability of a term in relevant/non-relevant documents:

$$\log \prod\nolimits_{t_i \in Q,D} \frac{p_i \cdot (1-q_i)}{q_i \cdot (1-p_i)} \quad \text{or} \quad \sum\nolimits_{t_i \in Q,D} \log \frac{p_i \cdot (1-q_i)}{q_i \cdot (1-p_i)} \tag{3}$$

In Equation (3), it was assumed that p_i is the same for all query terms and $\frac{p_i}{(1-p_i)}$ is a constant and can be ignored for ranking purposes. If $\log \frac{p_i \cdot (1-q_i)}{q_i \cdot (1-p_i)}$ as the weight of term-i in document D, this formulation becomes very similar to the similarity formulation in the vector space model with query terms assigned a unit weight [6].

2.1.3. *Inference network model.* In this model, document retrieval is modeled as an inference process in an inference network. In the simplest implementation of this model, a document instantiates a term with certain strength, and the credit from multiple terms is accumulated a query to compute the equivalent of a numeric score for the document. The strength of instantiation of a term for a document can be considered as the weight of the term in the document and document ranking in the simplest form of this model becomes similar to ranking in the vector space model and the probabilistic model. The strength of instantiation of a term for a document is not defined by the model and any formulation can be used [7].

2.1.4. *Term weighting.* Various methods for weighting terms have been developed in the field. Weighting methods developed under the probabilistic models that depend heavily upon better estimation of various probabilities. Methods developed under the vector space model are often based on researchers' experience with systems and large scale experimentation. In both models, three main factors come into play in the final term weight formulation, such as, term frequency, document frequency and document length [8].

After first TREC, using raw term frequency of terms is non-optimal, and a dampened frequency (e.g., a logarithmic t_f function) is a better weighting metric. Most research groups at TREC currently use some variants of these two weightings [9].

2.1.5. *Query modification.* Early research in IR relied on a thesaurus to find synonyms. Researchers developed techniques to automatically generate thesauri for use in query modification. Most query augmentation techniques were based on automatically generating thesauri, but had very limited success in improving search efficacy. Additional systems proposed using relevance feedback for query modification. Using relevance judgments, a system can then automatically generate a better query (e.g., by adding related new terms) for further searching. Relevance feedback has been shown to work quite effectively across test collections. New techniques to do meaningful query expansion in absence of any user feedback were developed. Most notable of these is *pseudo-feedback*, a variant of relevance feedback. Pseudo feedback has been shown to be a very effective technique, especially for short user queries [10,11].

2.2. **Mid-term approaches.** This section presents those approaches which were used mostly are as follows:

2.2.1. *Data structure based techniques.* Many other systems have been developed over the years and have met with varying success. Cluster hypothesis states that documents that cluster together will have a similar relevance profile for a given query. Natural Language Processing (NLP) has also been proposed as a tool to enhance retrieval effectiveness. The field has developed techniques to tackle many different problems like information filtering, topic detection and tracking (TDT), speech retrieval, cross-language retrieval, question answering, and more [9,12,13].

2.2.2. *Hashing approach.* Hash tables are efficient for doing a quick search. By applying a hash function to find an index or position that wanted to be accessed. It has two approaches:

In first approach, a collection of n elements whose keys are unique integers in $(1, m)$, where $m >= n$, then it can store the items in a *direct address* table as shown in Figure 1. Searching a direct address table: for a key, k, access T_k,

- if it contains an element, return it,
- if it does not then return a NULL.

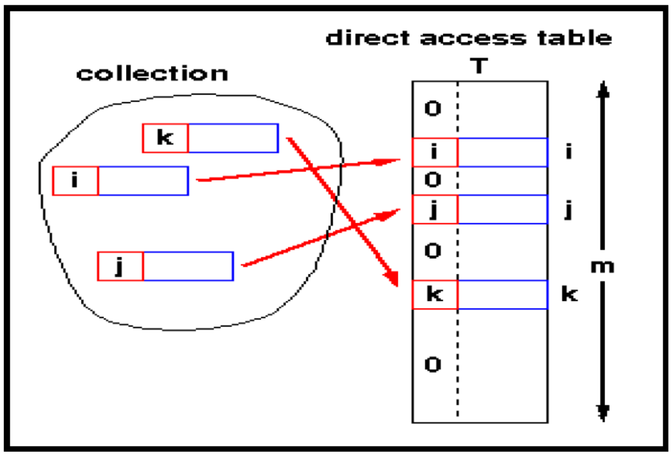

FIGURE 1. Direct address table

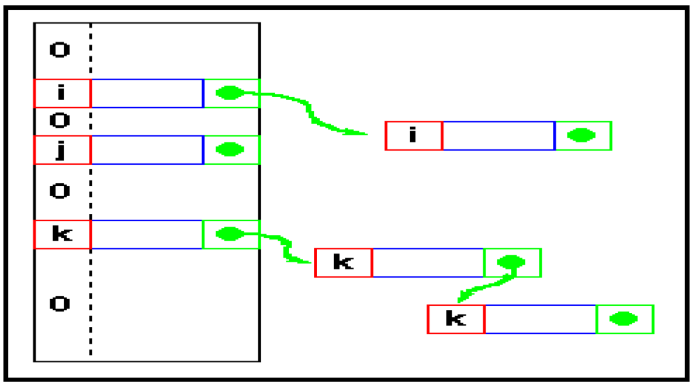

FIGURE 2. Direct address table with link

In second approach, if duplicates are the exception rather than the rule, then n_{dup}^{max} is much smaller than n and a direct address table will provide good performance. But if n_{dup}^{max} approaches n, then the time to find a specific element is $O(n)$ and a tree structure will be more efficient as shown in Figure 2.

Direct addressing is easily generalized to the case where there is a function;

$$h(k) => (1, m)$$

which maps each value of the key, k, to the range $(1, m)$. In this case, place the element in $T[h(k)]$ rather than $T[k]$ and search in $O(1)$ time as before. For details see references [14-17].

2.2.3. *Binary search trees.* An optimal search tree is one in which the probability of occurrence of all keys is equal (or is unknown, in which case assume it to be equal). For example, consider a dictionary of words used by a spelling checker for English language documents. It will be searched many more times for ‘a’, ‘the’, ‘and’, etc, than for the thousands of uncommon words which are in the dictionary just in case someone happens to use one of them. It is also reasonably easy to produce a table of the frequency of occurrence of words. If key, k, has relative frequency, r_k, then in an optimal tree:

$$\text{sum}(d_k r_k)$$

where, d_k is the distance of the key, k, from the root (i.e., the number of comparisons which must be made before k is found) [18].

2.3. **Recent approaches.** This section presents those systems which are used mostly recently:

2.3.1. *Data mining.* Data mining (DM) is understood as a process of automatically extracting meaningful, useful, previously unknown and ultimately comprehensible information from large databases. Data mining is not a single method or algorithm; rather it is a collection of various tools and approaches sharing the common purpose to "torture the data until they confess". Three methods presented below are perhaps the most common tools used in data mining [19,20].
2.3.1.1. Association rules. Association rules unearthing is perhaps the most spectacular example of Data mining, because it can quickly contribute when correctly implemented. Association models find items that occur together in a given event or record. To achieve results it uses rules for making strategic decisions.
2.3.1.2. Classification & clustering. The data that we are encountering is very rarely homogenous and in most cases it can be cataloged using various criteria. The characteristics of such segments and their numbers provide substantial information about the nature of data. In data mining there are two types of segmentation processes. The first classification maps and classifies, a data object into one, or several, and predefined classes. The second segmentation process known as clustering maps data object into one of several categorical classes which is to be determined from the examined data.
2.3.1.3. Statistical analysis. Statistical analysis is usually regarded as the most traditional method used in data mining. The most widely used simple statistical method is regression which builds models based on existing values to forecast what other values, not present in input data set, could be. There are many possible applications of regression, the most obvious being product demand forecasts or simulation of natural phenomena.
2.3.1.4. Sequential characteristic. The sequential characteristic refers to problems involved in data time correlation with data mining when the technologies, such as mining with the temporal association rules of the clustering method, Knowledge Discovery in Databases (KDD) are used. The data source of Web data is heterogeneous, complicated and continuously accumulative. Therefore, the sequential characteristic of data is a kind of statistical characteristic. TimelD in the properties of Web data objects is a time mark, showing the successive value that data change with time. The time interval for dereferencing of semi-structured data with TimelD is random, such as relevant statements about various kinds' of events in statistical departments. Generally, structured data are collected at equal intervals, such as daily transaction data of stock, daily transaction data of bank.
2.3.1.5. Internet search engines. Internet search engines extract information using the World Wide Web. This is achieved by using graphical query languages that is XML. Almost all search engines use classical keyword-based methods for information retrieval. These searches try to match user specified pattern (i.e., query) to texts of all documents in their database, and return the documents that contain terms from the query. However, sometimes documents cannot be retrieved because the specified pattern was not matched exactly. Search tools must deal not only with hypertext documents (in the form of WWW pages) but also with free-text repositories (message archives, e-books etc.), FTP and Usenet servers and with many sources of non-textual information such as audio, video and interactive content [21].

All the above subject matter was combined to accumulate the information relating to IR. Another aspect of these details was to establish link through the background study between the previous approaches with the modern approaches which leads to the better future.

3. **Proposed Approach Theoretical Findings.** In the proposed approach information, retrieval is based on the combination of both sequential and non-sequential approaches; and only the input numeric whole number is required. Based on the combination of any length for natural number sequence, address located is as shown in Figure 3.

In Figure 3, the input numbers of length "X" (any integer numeric number) represent the address. "X" is further divided into two sections, that is, "M" and "N". The length of "M" is fixed only one number (one character of numeric value which could be any number between 0-9). Part-I is represented by "C" as illustrated in Figure 4.

In the case of "N", the length is not fixed (any number any length). "N" is further divided into two sub-sections; Part-II is represented by "B" (one character of numeric value which could be any number between 0-9), which is used as a second number for address as a base address.

The Part-III of "N" is represented by "A" which is used to append with "B" to complete the search address, as shown in Figure 5.

In Figure 5, the number placed in "B" is used to development the table. The number "B" will provide the starting address for any location in the list of addresses present in the database to begin the search. All the memory addresses for each location starting from "0-9" are as given in Table 1.

Whereas, the Part-III of "N" is "A" which is to append with "B" to complete the address search. Let us suppose a number is received for "X = 78" as an address for

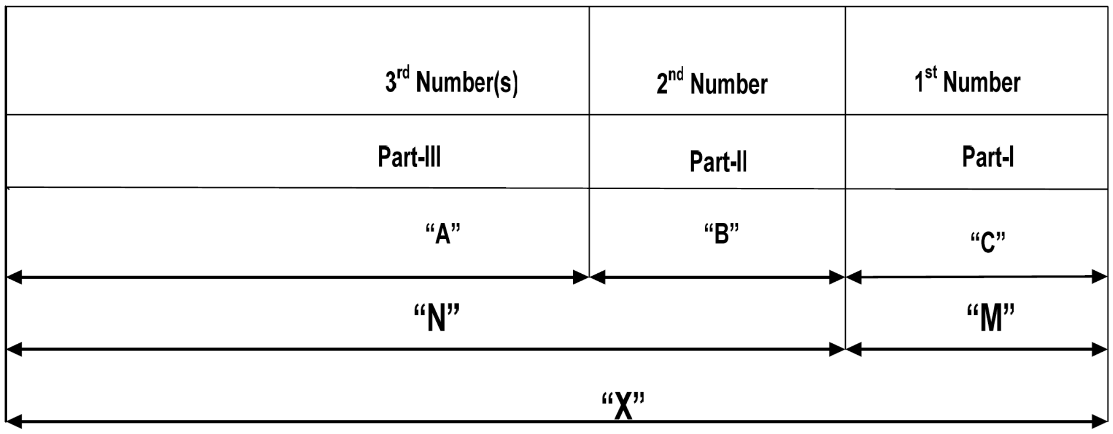

Figure 3. Format of address location for numbers "X"

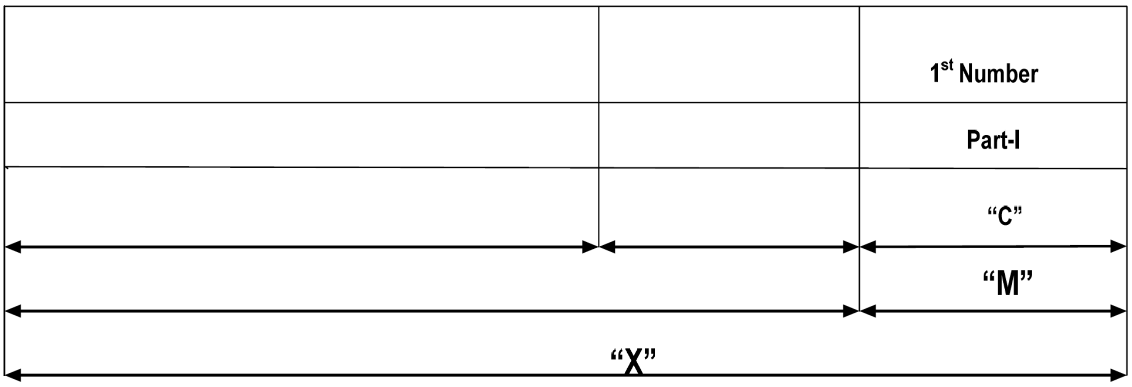

Figure 4. Format of numbers "M" showing position of "C"

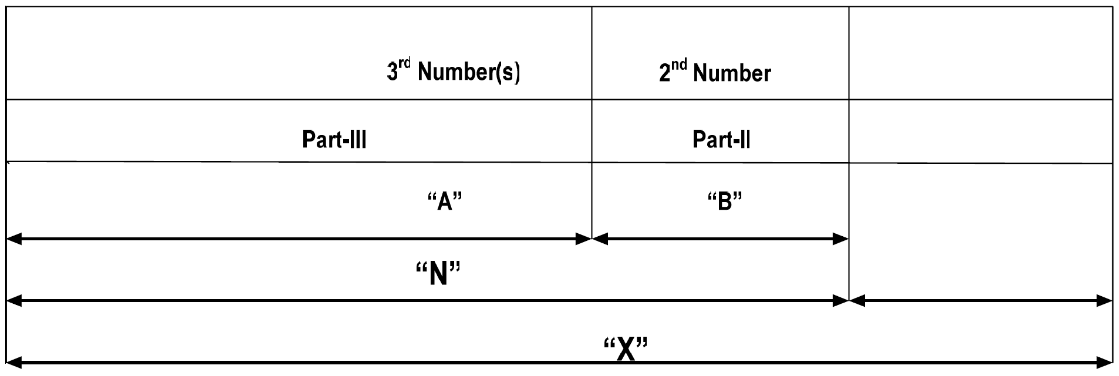

Figure 5. Format of numbers "N" showing splitting into "B" & "A"

Table 1. Basic memory structure belong to '0-9' addresses

Address			Data
A	B	C	
		0	
		1	
		2	
		4	
		5	
		6	
		7	
		8	
		9	

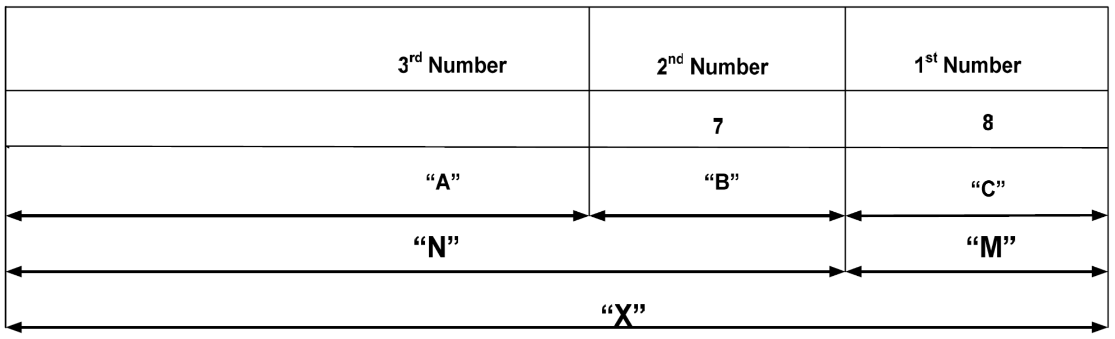

Figure 6. Format for searching "78"

information retrieval; a scenario is set up to reach to an exact location as illustrated in Figure 6.

In Figure 6, the length of "X = 2", therefore, "M = 1" and "N = 1". The value of "C = 8" is the only value. Further in case of "N", the value of "B = 7", and the value of "A" is null (empty) in this case. From the value of "B" a base of table is developed as shown in Table 2.

In Table 2, the number of "B" provides the start of base integer and the second number provides the incremental integer values from 0-9. When the table is established, the required address (which is "78" in this case) is selected and the information associated is retrieved. In this case, "A" has no value as the length of input number was 2 characters. Let us suppose another example with the value of "X = 178" which means the

Table 2. Memory structure for 70-79 addresses

Address			Data
A	B	C	
	7	0	
	7	1	
	7	2	
	7	4	
	7	5	
	7	6	
	7	7	
	7	8	**Required address for IR**
	7	9	

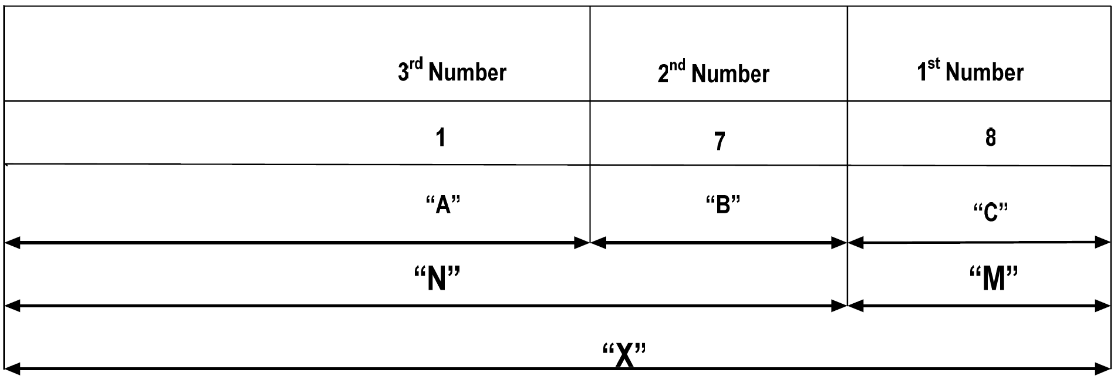

Figure 7. Format for searching "178"

Table 3. Memory structure for 170-179 addresses

Address			Data
A	B	C	
1	7	0	
1	7	1	
1	7	2	
1	7	4	
1	7	5	
1	7	6	
1	7	7	
1	7	8	**Required address for IR**
1	7	9	

combinations of "170-179" are the required addresses. As the length of "X = 3" in this case, therefore, the value of "C = 8", the value of "B = 7" and the value of "A = 1". The second number in "178", from right to left are selected, that is, "7" as "B" in this case. Add to "7" first "0" as a starting address and then "9" as last address and later append the third number "1" which belong to "A" to yield "170" and "179" addresses, respectively, as shown in Figure 7 and Table 3.

The required address is "178" and the required information is passed-on for retrieval.

4. **Implementation Methodology.** The implementation methodology of the proposed approach is based on the following pseudo codes:

```
BEGIN  {main start}
      INPUT DATA = X;{string of numeric whole number of any length}
      ASSIGN VALUE = N & M; {split X into N and M, respectively}
      ASSIGN VALUE = A & B; {split N into A and B, respectively}
      THEN SELECT:  $k_0$= STARTING POINT {for position of "M", which is represented by C}
                                    and
$k_9$ = FINISHING POINT  {for  position of "B" to "A"}
                                  If X = 2;  {then select second number from right to left for "B"}
                              Else
                                   If X = 3;  {then select second number from right direction "B" and  append "A"}
1.  X = N – C;{ suppose string length of X = 10, X= N-M, X=10–1 = 9, select first char for C}
            2.  N = A – B;  {from string of N= 9 numbers, again N= A- B= 9-1= 8, select first char for B}
            3.  A = append   {in this case 8 characters appended with "B&C" to generate the required address}
            GENERATE TABLE;   {starting from: $k_0$ and $k_9$}
             FOR ($k_0$ = 0;  $k_9$ <=9;  $k_0$++){$k_0$ = M,  $k_9$ = B, and A for any remaining numbers to append}
               BEGIN     {Start loop}
                   READ NUMBER;
                   PRINT NUMBER;
                   X= N - C;
                     N= A – B;
Concat = A + N; {attached rest of numbers with "A" with $k_0$to $k_9$}
                     PRINT Table:
        END;      {end loop}
END.    {main end}
```

The above pseudo codes were deployed in algorithm for testing operations.

5. **Simulation Results.** All the four algorithms were developed and run on PC with specifications; Intel Core ((TM) i7 x990 @ 3.47 GHz Processor RAM 12 GB 64) and the following results were obtained as shown in the Table 4.

Graph was depicted based on Table 4; for the five algorithms along with the different data sizes and search time for each algorithm, as shown in Figure 8.

TABLE 4. Results for different data sizes vs. retrieval time for five types

	Different search approaches				
Data size	Proposed approach search time	Data Mining sequential approach search time	Hash approach search time	Pattern matching approach search time	Binary search tress search time
20k	10ns	12ns	14ns	16ns	19ns
40k	18ns	22ns	24ns	28ns	31ns
60k	29ns	33ns	37ns	41ns	46ns
80k	40ns	45ns	49ns	55ns	60ns
100k	54ns	58ns	61ns	68ns	74ns

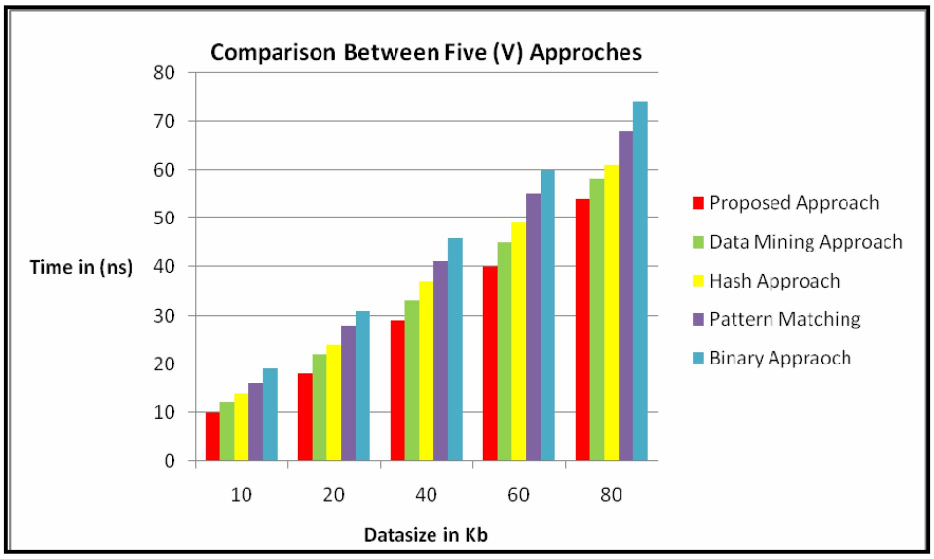

FIGURE 8. Data size vs. search time for five algorithms

During processing of data, it was noticed that all the five algorithms are producing promising results according to its own capabilities.

6. **Conclusion.** In this paper, steps have been formed for information retrieval techniques and explained the approach to erect a system. In preliminary experiments, it was shown that the proposed technique contributes not only the rules but also providing users a new viewpoint towards these discovered rules and a motivation to invoke a new method. In addition, it was presented that the database progresses either by contiguous or non-contiguous approaches to increment the storage data addresses/locations. The pointer moves to retrieve either one-by-one, sequential/continuous/contiguous address in incremental order or by using index methods, in another approach non-sequential/non-continuous/non-contiguous to retrieve the database or memory locations for data retrieval. Furthermore, the proposed method is a novel approach to the field of Information Retrieval. Hence, on the basis of study, simulation and comparison with other four methods, it was concluded that the performance of proposed technique is faster in information retrieval by consuming less search time as compared with others and moreover it is easier to implement for retrieval of information even in large database; therefore, it yields promising results.

Acknowledgment. The author is grateful to the Research Centre (RC), College of Computer and Information Sciences (CCIS), for extending all categories of support and continuous assistance during this research work. The author is also thankful to the Dean CCIS and the Chairman Information Systems Department, for their time to time encouragement.

REFERENCES

[1] G. Salton, *The SMART Retrieval System – Experiments in Automatic Document Retrieval*, Prentice Hall Inc., Englewood Cliffs, New Jersey, 1971.
[2] D. Harman, Overview of the first text retrievals conference, *Proc. of the 1st Text Retrieval Conference*, pp.1-20, 1993.
[3] G. Salton and M. McGill, *Introduction to Modern Information Retrieval*, McGraw Hill, New York, 1983.
[4] K. Sparck Jones and P. Willett, *Readings in Information Retrieval*, Morgan Kaufmann, 1997.

[5] G. Salton, A. Wong and C. Yang, A vector space model for information retrieval, *Communications of the ACM*, vol.18, no.11, pp.613-620, 1975.

[6] W. Croft and D. Harper, Using probabilistic models on document retrieval without relevance, *Information Journal of Documentation*, vol.35, pp.285-295, 1979.

[7] H. Turtle, *Inference Networks for Document Retrieval*, Ph.D. Thesis, University of Massachusetts, 1990.

[8] S. Robertson and S. Walker, Some simple effective approximations to the 2-poisson model for probabilistic weighted retrieval, *Proc. of ACM SIGIR'94*, pp.232-241, 1994.

[9] A. Singhal, J. Choi, D. Hindle, D. Lewis and F. Pereira, AT&T at TREC-7, *Proc. of the 7th Text Retrieval Conference*, pp.239-252, 1999.

[10] C. Buckley, J. Allan, G. Salton and A. Singhal, Automatic query expansion using SMART: TREC 3, *Proc. of the 3rd Text Retrieval Conference*, pp.69-80, 1995.

[11] S. Robertson, S. Walker and M. Beaulieu, Okapi at TREC-7: Automatic ad hoc, filtering, VLC and filtering tracks, *Proc. of the 7th Text Retrieval Conference*, pp.253-264, 1999.

[12] C. Buckley, G. Salton and J. Allan, Automatic retrieval with locality information using SMART, *Proc. of the 1st Text Retrieval Conference*, pp.59-72, 1993.

[13] A. Singhal, C. Buckley and M. Mitra, Pivoted document length normalization, *Proc. of ACM SIGIR'96*, pp.21-29, 1996.

[14] M. Singh and D. Garg, Choosing best hashing strategies and hash functions, *Proc. of Advance Computing Conference*, pp.50-55, 2009.

[15] W. Luo and G. Heileman, Exponential hashing in finite fields, *Proc. of International Conference on Embedded and Ubiquitous Computing*, pp.333-338, 2008.

[16] O. Kara and A. Atalay, Preimages of hash functions through rainbow tables, *Proc. of the 24th International Symposium on Computer and Information Sciences*, pp.304-309, 2009.

[17] M. Masud, G. Chandra Das, A. Rahman and A. Ghose, A hashing technique using separate binary tree, *Data Science Journal*, vol.5, pp.143-161, 2006.

[18] A. Bagheri and M. Razzazi, Drawing complete binary trees inside rectilinear polygons, *International Journal of Computer Mathematics*, vol.87, no.14, pp.3138-3148, 2010.

[19] Z. Ou, Data structuring and effective retrieval in the mining of web sequential characteristic, *Proc. of Electronic and Mechanical Engineering and Information Technology*, Harbin, China, pp.3551-3554, 2011.

[20] B. Mukhopadhyay and S. Mukhopadhyay, Data mining techniques for information retrieval, *Proc. of the 2nd Inter. CALIBER*, 2004.

[21] M. Ykhlef and S. Alqahtani, A survey of graphical query languages for XML data, *Journal of King Saud University – Computer and Information Sciences*, vol.23, no.2, pp.59-70, 2011.