



A Sequence Analysis Approach to perform Resource Allocation in Mobile Networks

Ritu Arora*, Mr. Gurdev Singh**

*M.Tech Scholar in JIET, Jind, Haryana, India.

**Assistant Professor & HOD CSE JIET, Jind, Haryana

Abstract: Search is one of the most effective operations performed on different kind of dataset including the text document or the database search. The text mining, web mining are the key areas that are associated with string search operation. In this present work an effective search method is suggested by modifying the input string structure. In this work, the input string is converted to the 2D structure based on the next character identification and the character analysis. Once the string is stored, the search operation becomes very effective. The optimality of the presented operation is based on the number of repeated characters in the string. The complexity analysis and the results obtained from the work are very satisfactory.

Keywords: Optimum Search, Text Mining, Complexity Analysis.

I. INTRODUCTION

Search techniques continue to advance, integrating new and traditional searching methods. An effective search tool should accommodate continually expanding collections, a characteristic of most information repositories. We have reviewed four search techniques. The first three, keyword search, full text search, and classification based search, are index based search in a structured repository, while the last one, hypertext-based search, and navigates the repository by following the predefined links. The hypertext technique can be combined flexibly with other search techniques as a complement to empower information retrieval process. The automatic indexing feature of a full text search has proven that it can be widely used in text-intensive documents like articles and books. Meanwhile, as keyword search and classification based search predefine the keywords or catalogues used as an index to describe concepts relevant to the domain of discourse, they introduce semantic information absent in full text search. Each search technique has advantages and is appropriate in specific circumstances. A suitable search technique should be proposed based on the features of information repository [1].

Searching Schemes	Advantage	Disadvantage
Enumerative Scheme	Fastest method	difficult to expand
Faceted Classification	Easily expandable, Users can freely choose features available for search	Components must fit the classification scheme otherwise irrelevant components are retrieved, Some components may overlap categories
Attribute Classification	no ordering	Slowest method
Free Text Classification	Fast and easy to implement, Users can submit any query	Ambiguous, indexing costs, Generally produces results with low precision and low recall.
Formal Specifications	Automated, more precisely characterize functionality	Difficult to understand
Knowledge- Based Approach	More powerful than traditional methods	Require enormous human resources, manually populated
Signature Matching	Provides good information about program functions, good for searching components based on source codes	Only suitable for strongly-typed programming languages, not good for metadata search

Figure 1: Searching Technique Comparison

Matching string sequence is a problem that computer programmers face on a regular basis. Some programming tasks, such as data compression or DNA sequencing, can benefit enormously from improvements in string matching algorithms. Pattern Retrieval often needs to provide a user with a list of components relevant to some search query. A lot of research goes into how to determine the best component to return. To provide results quickly even on large data sets some sort of index is required to map query terms to components, as well as appropriate algorithms for building the index ahead of time. Different kinds of indexes are used for the different qualities they have. Some indexes work for different kinds of data but most are designed for indexing words in

plain-text documents [2,3]. A reverse index tracks all the words in all indexed documents along with which documents they occur in

In this section, the work on the search algorithm is been defined. This section has defined the advantages and disadvantages of different search approaches. In section II, the work done in this area is discussed in detail. In section III, the proposed methodology is defined along with algorithm and the example. In section IV, the results obtained from the work are defined. In section V, the conclusion driven from the work is defined.

II. LITERATURE SURVEY

In this section, the work done by the earlier authors in the area of text mining, pattern search is defined. Jiong Yang performed a work, "TrajPattern: Mining Sequential Patterns from Imprecise Trajectories of Mobile Objects". Author are only able to obtain the imprecise location of a mobile object at a given time. Sequential patterns are a popular data mining model[3]. Gyoza Gid'ofalvi performed a work, "Mining Long, Sharable Patterns in Trajectories of Moving Objects". This paper presents a database projection based method for efficiently extracting such long, sharable frequent routes[4]. Chin-Feng Lee performed a work, "exploiting data mining techniques to generate prefetching mechanism and broadcasting strategies in mobile computing environment". In this paper, Author will develop a cyclic sequential mining to discover a set of location-based cyclic sequential patterns (LBCS) which are frequently requested by a number of mobile users over time.[5] Na Yao performed a work "Managing Moving Congestion Patterns in Mobile Networks". Managing congestion is essential to maximise traffic carried in a mobile network and in this paper a technique is presented that uses learning to co-operatively change antenna patterns across a cluster of cells, so minimising blocking. Moreover, the technique allows the changing radiation patterns to track congestion as it moves[6]. Po-Ruey Lei performed a work, "Mining Spatial-Temporal Movement Profile of Mobile Users for Social-Aware Applications". In order to explore the community in their movement, movement behavior mining for individual user is a fundamental and essential task. To enhance the accuracy and conciseness of movement behavior modeling, in this paper, Author propose a spatial-temporal movement profile model (abbreviated as STMP) to capture movement behavior of objects. STMP is represented as a probabilistic suffix tree with spatial and temporal information of movements[7].

Shuai Ma performed a work, "Combining Clustering with Moving Sequential Patterns Mining: A Novel and Efficient Technique". In this paper a novel and efficient technique is proposed to mine moving sequential patterns. Firstly the idea of clustering is introduced to process the original moving histories into moving sequences as a preprocessing step. And then an efficient algorithm, called PrefixTree, is presented to mine the moving sequences. Performance study shows that PrefixTree outperforms LM algorithm, which is revised to mine moving sequences, in mining large moving sequence databases[9].

Grzegorz Cielniak proposes a technique for learning collections of trajectories that characterize typical motion patterns of persons. Data recorded with laser-range finders is clustered using the expectation maximization algorithm. Based on the result of the clustering process Author derive a Hidden Markov Model (HMM). This HMM is able to estimate the current and future positions of multiple persons given knowledge about their intentions[10]. Yen-Ssu Chou performed a work, "Mining User Movement Behavior Patterns in a Mobile Service Environment". In this paper, Author define the match join of mobile users U , movement locations L , staying time in timestamps T , and service requests S with weight w to be a subset of the w -join of U , L , T , and S such that each tuple of U , L , T , and S contributes to at least one result tuple. [11].

III. PROPOSED APPROACH

The presented work is an effective search technique in which the search environment is created in the form of two dimensional structure. In this structured format, the input text sequence is placed in the structure as the input format of the data. Once the data is organized, now the search is performed on this organized dataset instead of input string. The presented approach gives the character and position based search with the direct indirect index concept. It means we can move to the particular character occurrence directly without performing the sequence visit. When some word is to search, the search sequence is explored character by character. Each row is compared to the character and the character position directly so that the sequence match will be performed effectively. If some mismatch occurs, then the search will be terminated immediately. The presented work is divided in two main stages. First to build the effective two dimensional structure in which the input text string will be placed and in second stage the search over the text pattern will be performed. The process defined to place the text in 2D matrix is shown in figure 2.

As we can see in the algorithm. The presented work is the analytical approach to retrieve the information from the input text and then build the 2D matrices to store the character and the frequency of the text character. To generate the matrix at first, the number of unique characters from the text will be identified. Once the character identification will be done. It will identify the frequency of each character. The length of maximum frequency character will define the number of columns in two dimensional structures and number of unique characters will

define the number of rows to the structure. Now the input text will be scanned sequentially and identify the occurrence of each character as well as its follower. Each matrix position will contain two information one is the next character and second is the frequency of the character.

```

Match(Text,Pattern)
/*Text is the actual text available and pattern is the string we have to
search*/
{
Step 1: Identify the unique characters over the pattern called uchar
Step 2: Set n=length(uchar)
        [Identify no of unique character over the string]
Step 3: Identify the maximum occurrence of any character in String called m
Step 4: Define a character matrix of size(n,m*2)
Step 5: for i=1 to length(Text)
        [Process all the text characters]
        {
            for j=1 to length(uchar)
            [identify all characters in the list]
            {
Step 6: Identify the character and the next character and its frequency call k
Step 7: Matrix(j,k)=uchar(j)
        Matrix(j,k+1)=f
        [Store character in matrix]
            }
        }
Step 8: Exit
}

```

Figure 2: Proposed Algorithm

Once the information is placed over the matrix, the next work is to perform the search over the string. This search mechanism will be performed character by character. For each character, the relative row will be searched and the next follower character will be identified. If the character is same as of next character in search pattern. The search will be continued. Otherwise, the search process will be terminated. The example of the search and placement process is shown in example given as under

A) Example

Let we have an input Text String given as

ABCDADBCABDDAABD

Then, with the first stage, the number of unique characters will be identified and the index key will be set. Based on these characters, the number of rows in the two dimensional structure will be defined.

A => 1

B => 2

C => 3

D => 4

Now based on the algorithm defined above, the 2D structure will be generated. The structure formation is shown in figure 3.

	1	2	3	4	5
A	2,1	4,2	2,3	1,4	2,4
B	3,1	3,2	4,3	4,5	
C	4,1	1,3			
D	1,2	2,2	4,4	1,4	99,99

Figure 3: 2D Structure Formation

This table is able to answer any input query performed by the user. User can perform any kind of pattern search over the text by using this 2D structure. The search stages of the work is given as under

B) Single Character Search

If we have to count the number of occurrence of any character over the text, we have find the number of filled columns in the particular character index. Let if we have to find the occurrence of B. We have to scan second row as shown in figure 2.

	1	2	3	4	5
A	2,1	4,2	2,3	1,4	2,4
B	3,1	3,2	4,3	4,5	
C	4,1	1,3			
D	1,2	2,2	4,4	1,4	99,99

Figure 4: Frequency of single character identification

The complexity to search the single character occurrence frequency over the text is $O(1)$.

C) 2-Character Identification

To identify the frequency of the combination of two characters, we can search the follower character value in the 2D index character position. Now number of follower characters represents the number of occurrence of combination character. Let we have to search the occurrence of AB over the string.

	1	2	3	4	5
A	2,1	4,2	2,3	1,4	2,4
B	3,1	3,2	4,3	4,5	
C	4,1	1,3			
D	1,2	2,2	4,4	1,4	99,99

Figure 5: Search of two character positions

Figure 5 is showing, perform the search on index (A) row and find the occurrence of B(2) over the row. As we can see, we got the three occurrence of AB. Let the size of matrix is $N \times M$ then the complexity of the search operation over the matrix for two character combination is $O(M)$ for the worst case.

D) 3 or More Character Identification

Let to perform the occurrence of DBC over the matrix. The first work is search the DB over the matrix as defined in section B). The DB search is shown in blue color in figure6.

	1	2	3	4	5
A	2,1	4,2	2,3	1,4	2,4
B	3,1	3,2	4,3	4,5	
C	4,1	1,3			
D	1,2	2,2	4,4	1,4	99,99

Figure 6: Search Process of 3 or character word

Now to search the third character we move to 2nd row and 2nd column and identify the character at that position. If the character is C, then the string DBC is recognized as shown in brown color. This shows the recognition of the string over the text using 2D design. The complexity of the search process will $O(M+n)$. Here M is the number of columns in the matrix and n is the length of search pattern.

D) Analysis

The analysis of the presented work is been respective to the search complexity. Let we have a string of size N with k number of characters. When the characters are placed in the 2D structure. The sequential search is performed. The complexity of this process is $O(N)$. After this structure design process a matrix is generated of

size $k \times M$. Here k is the number of unique characters and M is the length of maximum frequency character. Let we have to search a pattern of size n . Then the complexity analysis in different cases is shown in table 1.

Table 1: Complexity Analysis

Pattern Size	Complexity
1	$O(1)$
2	$O(M)$
3 or More	$O(M+n)$

IV. RESULTS

In this work we have taken a random string as the input text generated using the fix number of characters. The input character is provided as the search pattern. As the search is performed, all the possible string combinations are generated and the search is performed for each search pattern. The results obtained are here been analyzed in terms of time taken to search the string combinations. The results under different parameters are shown in table 2.

Table 2: Time Based Analysis

Number of Character combination	Text Length	Pattern Length	Number of String Searched	Time Taken (In Sec)
3	25	3	6	.047
4	25	3	6	.046
5	25	3	6	.047
6	25	3	6	.048
3	50	3	6	.062
4	50	3	6	.053
5	50	3	6	.055
6	50	3	6	.055
3	100	3	6	.057
4	100	3	6	.055
5	100	3	6	.056
6	100	3	6	.054

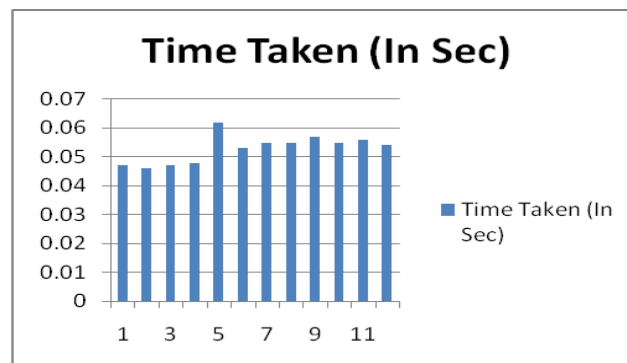


Figure 7: Time Taken Vs. Number of Runs

Here figure 7 is showing the results obtained for different search over the string with different input text.

Table 3: Time Based Analysis

Number of Character combination	Text Length	Pattern Length	Number of String Searched	Time Taken (In Sec)
3	100	3	6	.061
4	100	4	24	.906
5	100	5	120	1.157
6	100	6	720	8.468

Here table 3 is showing is showing the search analysis respective number of string combination searched. The search results are shown in the form of graph presented in figure 8.

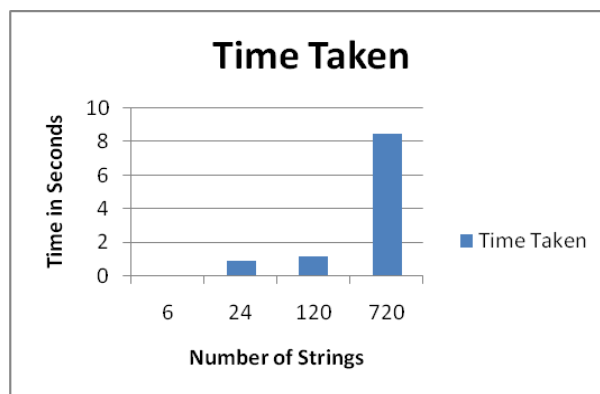


Figure 8 : Time Taken Vs. Number of String Search

IV. CONCLUSION

In this paper, an effective search mechanism is been represented that actually change the structure design of the input text. Once the text is converted to this structure, the optimal search results are obtained from the work. The obtained results show that the presented work is effective enough to provide effective search results.

V. REFERENCES

- [1]. Chih-Chieh Hung, "Exploring Regression for Mining User Moving Patterns in a Mobile Computing System".
- [2]. Karen H. Wang, "Group Mobility and Partition Prediction in Wireless Ad-Hoc Networks".
- [3]. Jiong Yang, "TrajPattern: Mining Sequential Patterns from Imprecise Trajectories of Mobile Objects".
- [4]. Gyozo Gid'ofalvi, "Mining Long, Sharable Patterns in Trajectories of Moving Objects".
- [5]. Chin-Feng Lee, "Exploiting data mining techniques to generate prefetching mechanism and broadcasting strategies in mobile computing environment".
- [6]. Na Yao, "Managing Moving Congestion Patterns in Mobile Networks".
- [7]. Po-Ruey Lei, "Mining Spatial-Temporal Movement Profile of Mobile Users for Social-Aware Applications".
- [8]. Grzegorz Cielniak, "People Recognition by Mobile Robots".
- [9]. Shuai Ma, "Combining Clustering with Moving Sequential Patterns Mining: A Novel and Efficient Technique".
- [10]. Grzegorz Cielniak, "Where is . . . ? Learning and Utilizing Motion Patterns of Persons with Mobile Robots".
- [11]. Yen-Ssu Chou, "Mining User Movement Behavior Patterns in a Mobile Service Environment".