Tuijin Jishu/Journal of Propulsion Technology
ISSN: 1001-4055
Vol. 44 No. 3 (2023)

A New Variant of Move-To-Front (SPIN-Move To Front) List Accessing Algorithm

Asst. Prof. Prasanta Kumar Mishra[1]and Asst. Prof. Shiba Prasad Dash[2]
[1]*Nalanda Institute of Technology, Bhubaneswar, Odisha*
[2]*Biju Patnaik University of Technology, Odisha, Rourkela.*

Abstract: List Accessing Problem is a well studied research problem in the context of linear search. There are many existing algorithms are available for list accessing problem. Move-To-Front (MTF), Transpose (TRANS), and Frequency Count (FC) are the three primitive and widely used list accessing algorithms. All other list accessing algorithms are the variants of these three algorithms. In this paper we have made an experimental analysis of MTF and IMTF algorithms and develop "A new variant of MTF (SPIN-MTF) list accessing algorithm". The experimental analysis of the algorithm shows that SPIN-MTF is performing better than MTF and IMTF algorithm.

Keywords*: list accessing algorithm; Data Structure; Linear Search; Experimental Analysis; move-to-front; spin-move to front.*

.

1. Introduction

The basic method for searching in an unordered linear list is called linear search. An unordered linear list is a simple data structure where one can performs insertion, deletion and access operation. In case of insertion and deletion any one can easily handle the list but in case of access operation it is not a easy task. For accessing an element from a list first we have to search that location of the element where the element is present. Linear search is a searching algorithm which obtains its result by traversing a list of data items in a linear way. It will start at the beginning of a list and process through until the requested element is found. In the concept of linear search the list update problem is a popular problem for last few decades. The list update problem has taken a list of distinct element and a request sequence as input and each element of the request sequence corresponding to an operation on an element of the given input list. For accessing an element from the list based on the request sequence is an access operation known as *list accessing problem* (LAP). When an element of the request sequence is served on the list, the requested element is accessed in the list having some access cost using a cost model. The cost model defines the way in which the cost is assigned to an element when it is accessed in the linear unsorted list. After accessing the requested element the list is reorganized, so that this is called *self organizing linear list* to reduce the cost of the element of the list for future request.

A) *List Accessing Cost Model*

After accessing an element in the list, a cost model is used to calculate the total access cost. Cost model which is most widely used for list accessing problems are *full cost model* and *partial cost model*. Full cost model is popularly known as standard cost model (Sleator and Tarjan 1985), in this model the access cost for a requested element is the position of the element from the front of the list. For example, the access cost of the ith element in the list is i. In partial cost model the cost of accessing an element is the number of comparison made with the preceding element in the list before accessing the element. For example, the access cost of i^{th} element in the list is i-1

Because it requires i-1 comparisons before accessing the element i. Immediately after accessing an element, the accessed element can be moved anywhere in the list without paying any cost. This type of exchange is treated as *free exchange*. Any other exchange of two elements in the list costs 1. This type of exchange is treated as *paid exchange.*

B) *List Accessing Algorithms*

List accessing algorithms are classified into two types i.e. *offline* and *online.* In *offline algorithm,* the whole request sequence is completely known beforehand. But in case of online algorithm the request sequence is partially known. Here the request sequence are served one by one on the order of their arrival. There are three widely used list accessing algorithms are Move-To-front (MTF), Transpose (TRANS) and Frequency-Count (FC).

Move-to-Front (MTF): After accessing an element in the list, the accessed element is moved to the front of the list without changing the relative order of other elements in the list.

Transpose (TRANS): After accessing an element of the request sequence, it is exchanged with immediately preceding element in the list.

Frequency Count (FC): There is a counter for each element which counts the frequency of the element of the list according to based on the requests from the request sequence. The list is arranged in decreasing order of frequency count of items in the list.

C) *Look Ahead* (Albers, 1994)

The list accessing algorithm knows not only the present request to be served, but also some future requests this is known as the *look-ahead.*

Weak look-ahead of size k: The algorithm sees the present request and the next k future requests.

Strong look-ahead of size k: The algorithm sees the present request and a sequence of future requests. This request contains k pair wise distinct elements which also differ from the element requested by the present request.

2. Related Work

List update problem is a very useful work in the problem solving technique in many applications. So that the list update problem has been high demand from last four decades. Till date several works have been done for list accessing problem. McCabe (1965) started the initial work in this field. He investigated the problem of maintaining a sequential file and developed two list accessing algorithms Move-To-Front (MTF) and Transpose (TRANS). List accessing problem was studied by many persons like J.L.Bentley, C.C. McGeoch Chung, G.H. Gonet, J.I. Munro, R. Revest (1965-1985) with the assumption that the request sequence is generated by probability distribution. Hester and Hirschberg (1985) have provided an extensive survey of average case analysis of list accessing algorithms with some challenging open problem. Sleator and Tarjon (1985) in their paper studied the Amortized efficiency of MTF, Transpose and Frequency Count for dynamically maintaining a linear list assuming accessing the ith element from the front of the list takes θ(i) times. The concept of strong and weak look-ahead (Albers, 2002) is introduced in the list accessing problem. A detailed study of online algorithms for the list update problem was published (N. *Reingold and J.Westbrook,* 1996). Spyros Angelopoulos , Reza Dorrigiv Alijandro and A Lopez-Ortiz (2006) on their paper “List Update With Locality of Reference” prove that MTF is unique optimal algorithms for list update and this hold for both the standard cost mode. Mohanty and Narayanaswamy (2009) in their paper have done a comprehensive survey of online list accessing algorithms and associated results are mentioned. Rakesh Mohanty, Burle sharma and Sasmita Tripathy(2011) have done their research work on characterization of request sequence for list accessing problem based on several factors such as size of the list, size of the request sequence, ordering of elements and frequency of occurrence of elements in the request sequence. Rakesh Mohanty, Shiba Prasad Dash, Burle Sharma and Sangita Patel (2012) works on some novel results from analysis of MTF list accessing algorithm. Rakesh Mohanty, Sasmita Tripathy (2012) in this paper they have made a comprehensive analysis of MTF algorithm and developed an Improved-MTF (IMTF) offline algorithm.

3. Preliminaries

Till date many list accessing algorithms have been developed. Such algorithms are Move-To-Front (MTF), Transpose (TRANS), Frequency-Count (FC), Improved-MTF (IMTF), Variant of Frequency Count (VFC) etc. In our study we have considered the MTF and IMTF list accessing algorithms.

MTF Algorithm: After accessing an element in the list, the accessed element is moved to the front of the list, without changing the relative order of other elements in the list.

Illustration: - Let the list be L= 1, 2, 3 and the request sequence R= 3, 2, 1, 3, 2. Each time after accessing, the accessed element is moved to the front of the list. So the total access cost for MTF using full cost model is 3+3+3+3+3=15.

IMTF Algorithm: After accessing an element IMTF moves the accessed element to the front of the list if and only if the accessed element is found by looking ahead of the next i-1 elements in the request sequence. If the accessed element is not found within the next i-1 elements in request sequence, then the list configuration remains same (R. Mohanty and S. Tripathy, 2012).

Illustration: - Let the list be L= 1, 2, 3 and the request sequence R= 3, 2, 1, 3, 2. Here the 1st element of the request sequence is 3 which is present in the 3rd position in the list, so the access cost is 3. After accessing, it checks next (3-1) elements in the request sequence, whether the element is present or not. Here it is not present, so the configuration of the list remains same. This process will continue till we reach the end of the request sequence. Here the total access cost is 3+2+1+3+2=11.

4. The Proposed SPIN-MTF List Accessing Algorithm

We have studied the MTF and IMTF algorithms and explore their limitation and we proposed "A new variant of MTF (SPIN-MTF) list accessing algorithm" which is performing well as compare to MTF and IMTF algorithms. The limitation of MTF algorithm is, it performs poorly for distinct element of request sequence and the limitation of IMTF algorithm is, it performs poorly if the algorithm accesses the elements which are present towards the end of the list. Here we used the positional look-ahead concept which is used in IMTF algorithm (R. Mohanty and S. Tripathy, 2012) and full cost model (Sleator and Tarjan, 1985) for calculating the total access cost of the algorithm after accessing all the elements of the request sequence.

SPIN-MTF Statement-"*After accessing an element X_i those position is i from the front of the list, it move to the front if and only if the access element X_i is present in next i-1 elements from the access element of the request sequence. If X_i is not present within the next i-1 elements in the request sequence, the list configuration will be changed i.e. the element of the list will be reverse in periodically, so that it will reduce the cost of the subsequent element which are present towards the end of the list*".

Positional look-ahead: In *positional look-ahead* (R. Mohanty and S. Tripathy, 2012) model the size of look-ahead for each requested element is depend on the position of previous requested element in the list. More specifically, If 'i' is the position of the σ (t) the list, then for answering σ (t+1) request the algorithm already knows the σ(t+2), σ(t+3),,……….σ (t+i-1) elements. Here we assumed that for answering σ (1), the algorithm knows only σ (1) element. According to positional look-ahead models the size of look-ahead can be varies from 1 to k-1. Here k is the size of input list.

Pseudo code of SPIN-MTF Algorithm

```
SPIN-MTF (L, R, C)
                    L: List
                    R: Request Sequence
                    C: Cost (Cost=0)
1.  For i=1,2…..n repeat step 2 to  9
2.  Search the request r[i] element in the list
3.  Cost=Cost+ position
4.  j=position
5.  Search r[i] in next j-1 elements from i^th of request sequence
6.      If r[i] is present
7.            Do  MTF
8.      Else
9.            Spin the list L (Reverse list L)
10. Return Cost
```

Illustration of SPIN-MTF Algorithm: Let the list L= 1, 2, 3 and the request sequence R= 3, 2, 2, 1, 3.

Step-1: The 1st element of the request sequence is 3 and it is present in the 3rd position of the list. So the access cost is 3 using full cost model. Check next (3-1) elements in R. Here it is not present so the list L will be spin i.e. reverse the element of the list L. Now the list configuration is L= 3, 2, 1.
Step-2: The 2nd element of the request sequence is 2 and it is present in the 2nd position of the list. So the access cost is 2 using full cost model. Total access cost is 3+2=5. Now check next (2-1) elements in R. Here it is present so the element will move to the front of the list L (do MTF). Now the list configuration is L= 2, 3, 1.
Step-3: The 3rd element of the request sequence is 2 and it is present in the 1st position of the list. So the access cost is 1 using full cost model. Total access cost is 3+2+1=6. Now check next (2-1) elements in R. Here it is not present so the list L will be spin i.e. reverse the element of the list L. Now the list configuration is L= 1, 3, 2.
Step-4: The 4th element of the request sequence is 1 and it is present in the 1st position of the list. So the access cost is 1 using full cost model. Total access cost is 3+2+1+1=7. Now check next (1-1) elements in R. Here it is not present so the list L will be spin i.e. reverse the element of the list L. Now the list configuration is L= 2, 3, 1.
Step-5: The 5th element of the request sequence is 3 and it is present in the 2nd position of the list. So the access cost is 2 using full cost model. Total access cost is 3+2+1+1+2=9. Now check next (1-1) elements in R. Here it is not present, because it is the last element in R, so the list L will be spin i.e. reverse the element of the list L. Now the list configuration is L= 1, 2, 3.
After accessing all the elements of the request sequence from the list, the total access cost is 3+2+1+1+2=9.

4.2 **Experiments and Results**

We have performed experimental analysis by implementing MTF, IMTF and SPIN-MTF algorithms. For each algorithm, we have generated the list and request sequence for different data set. We calculate the total access cost of each algorithm for different request sequences size and list size.

A) Alphabetical and Numerical Data Set

In this experiment a linear linked list is generated for request sequence by randomly selected numbers and alphabets from the key board. Here the base value corresponds to the maximum list size 2, 8, 10 and 16 respectively. These types of data generate in experiment -1 and calculate the total access cost of MTF, IMTF and SPIN-MTF algorithms.

B) Data Set with Constant List Size

In 2nd experiment we calculate the access cost of MTF, IMTF and SPIN-MTF algorithm with fixed size of input list i.e. 26 and varying the size of request sequence from 30 to 70 no of characters. Request sequence is generated from taking different word of English language where 26 distinct characters are present.

4.3 Implementation Results

In our experiment, we implemented the MTF, IMTF and SPIN-MTF algorithms in C language with Windows 7 operating systems. Here we use the singly linked list data structure. In our source code we declared a structure data type for generating two singly linked lists one for request sequence and the other for the list. In our code we use following user defined functions. The function append( ) generates a Linked list for input list and stores the distinct character of request sequence in the information part of the node of the linked list. REQ() function is used to generate another linear linked list for request sequence and stores all the character of request sequence in the information part of the linked list.
MTF( )- This function is used for calculating the total access cost using MTF algorithm for a given list and a request sequence.
IMTF( )- This function is used for calculating the total access cost using IMTF algorithm for same list and request sequence.
SPIN-MTF( )-This function is used for calculating the total access cost using SPIN-MTF algorithm for same list and request sequence.
We conduct different experiments with changing the list configuration and request sequence of different sizes and calculate the total access cost for MTF, IMTF and SPIN-MTF algorithms. Let L be the size of the list and N be the size of the request sequence. Let C_{MTF} be the total access cost of MTF algorithm, C_{IMTF} be the total access cost of IMTF algorithm and $C_{SPIN\text{-}MTF}$ be the total access cost of SPIN-MTF algorithm.

Experiment-1 (Experimental results of alphabet and numerical data set)

N	L	C_{MTF}	C_{IMTF}	$C_{SPIN-MTF}$
50	16	321	282	179
50	10	120	120	83
50	8	166	152	102
50	2	75	65	50
100	16	595	546	479
100	10	368	340	281
100	8	422	325	272
100	2	148	130	100
200	16	1380	1179	1010
200	10	473	473	339
200	8	644	590	516
200	2	316	277	201

Table 1: Experimental Result of MTF, IMTF and SPIN-MTF Algorithms

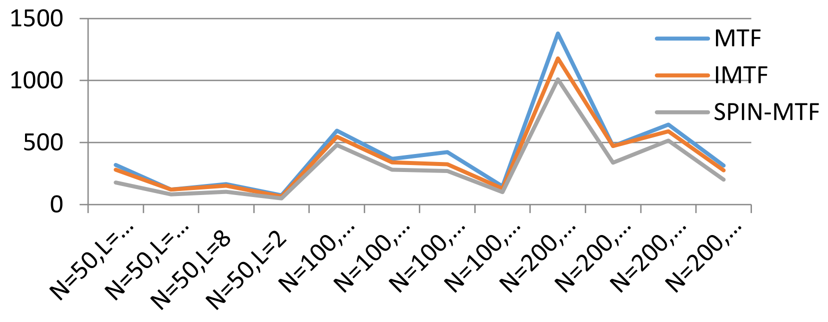

Figure 1: Line chart for Alphabetic and Numeric data set

Table 2: Experimental results of data set with constant size of list and variable size of request sequence (Experiment-2)

N	L	C_{MTF}	C_{IMTF}	$C_{SPIN-MTF}$
30	26	545	399	198
40	26	653	503	497
50	26	471	441	372
60	26	618	506	495
70	26	499	464	357

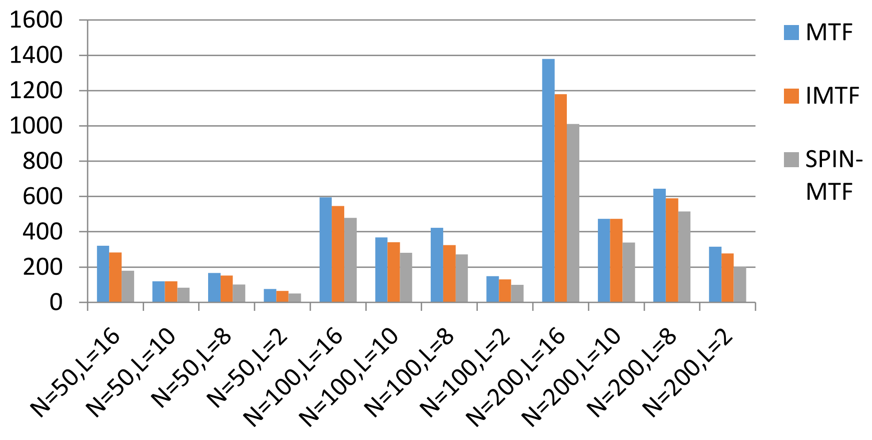

Figure 2 :Bar chart for Alphabetic and Numeric data set

Experiment-2 (Experimental results of alphabet and numerical data set)

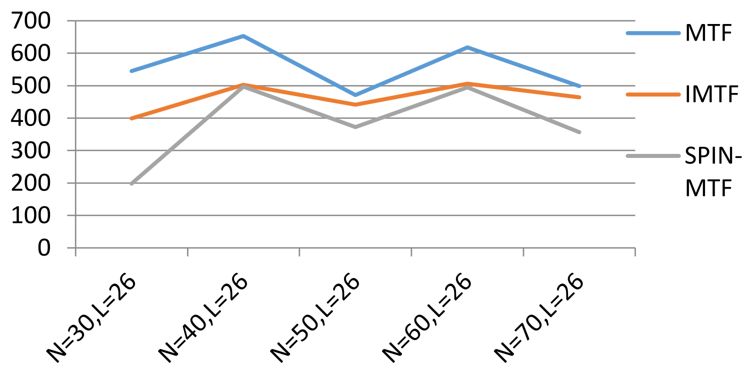

Figure 3: Line chart for constant list size and variable size request sequence

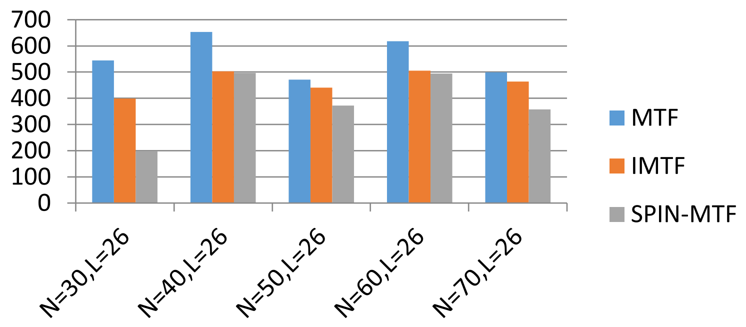

Figure 4: Bar chart for constant list size and variable size request sequence

Conclusion

In this paper, we have presented a new variant of MTF (SPIN-MTF) list accessing algorithm, which is observed to be better than MTF and IMTF algorithms. An experimental analysis of SPIN-MTF over MTF and

IMTF has been done. Our experimental analysis shows that there is significant performance gain of SPIN-MTF over MTF and IMTF algorithms.

Here experimental analysis of SPIN-MTF algorithm has been done and in future theoretical analysis can be done in order to achieve optimal cost.

References

[1] McCabe, J. (1965) 'On serial files with relocatable records', *Operation Research*, Vol. 12, pp.609-618.

[2] Carlisle, D. (2010, April). *graphicx: Enhanced support for graphics.* Retrieved from http://www.c Albers, S. (1994) 'A competitive analysis of the list accessing problem with look ahead', *Springer Lecture Notes in Computer Science*, Vol. 841, pp.201-210.

[3] Mohanty, R. and Narayanaswamy, N.S. (2009) 'Online algorithms for self organizing sequential search – A survey', *Electronic Colloqiumon Computational Complexity(ECCC),* Rep. No. 97.

[4] Mohanty, R., Sharma, B. and Tripathy, S. (2011) 'Characterization of request sequences for list accessing problem and new theoretical results for MTF algorithm', *International Journal Computer Applications (IJCA),* Vol. 22, Rep. No. 8, pp.35-40.

[5] Mohanty, R., Dash, S.P., Sharma, B. and Patel, S. (2012) 'Performance Evaluation of A Proposed Variant of Frequency Count (VFC) List Accessing Algorithm', *IJSAA,* Vol. 2, pp. 234-237.

[6] Mohanty, R., Dash, S.P., Sharma, B. and Patel, S. (2012) 'Some Novel Results From Analysis of Move To Front (MTF) List Accessing Algorithm', *IJSAA,* Vol. 2, pp. 238-242.

[7] Mohanty, R and Tripathy, S. (2012) 'An Improved Move-To-Front (IMTF) Off-line Algorithm for the List Accessing Problem', *International Journal on Advance Computer and Communication,* Vol. 3, Issue.1, pp.19-24.