

## Self Organizing List | Set 1 (Introduction)

The worst case search time for a sorted linked list is  $O(n)$ . With a Balanced Binary Search Tree, we can skip almost half of the nodes after one comparison with root. For a sorted array, we have random access and we can apply Binary Search on arrays.

One idea to make search faster for Linked Lists is [Skip List](#). Another idea (which is discussed in this post) is to *place more frequently accessed items closer to head*.. There can be two possibilities. offline (we know the complete search sequence in advance) and online (we don't know the search sequence).

In case of offline, we can put the nodes according to decreasing frequencies of search (The element having maximum search count is put first). For many practical applications, it may be difficult to obtain search sequence in advance. A [Self Organizing list](#) reorders its nodes based on searches which are done. The idea is to use locality of reference (In a typical database, 80% of the access are to 20% of the items). Following are different strategies used by Self Organizing Lists.

- 1) Move-to-Front Method:** Any node searched is moved to the front. This strategy is easy to implement, but it may over-reward infrequently accessed items as it always move the item to front.
- 2) Count Method:** Each node stores count of the number of times it was searched. Nodes are ordered by decreasing count. This strategy requires extra space for storing count.
- 3) Transpose Method:** Any node searched is swapped with the preceding node. Unlike Move-to-front, this method does not adapt quickly to changing access patterns.

### Competitive Analysis:

The worst case time complexity of all methods is  $O(n)$ . In worst case, the searched element is

Google™ Custom Search



GeeksforGeeks



53,527 people like [GeeksforGeeks](#).



[Interview Experiences](#)

[Advanced Data Structures](#)

[Dynamic Programming](#)

[Greedy Algorithms](#)

[Backtracking](#)

[Pattern Searching](#)

[Divide & Conquer](#)

[Mathematical Algorithms](#)

[Recursion](#)

[Geometric Algorithms](#)

always the last element in list. For [average case analysis](#), we need probability distribution of search sequences which is not available many times.

For online strategies and algorithms like above, we have a totally different way of analyzing them called *competitive analysis* where performance of an online algorithm is compared to the performance of an optimal offline algorithm (that can view the sequence of requests in advance). Competitive analysis is used in many practical algorithms like caching, disk paging, high performance computers. The best thing about competitive analysis is, we don't need to assume anything about probability distribution of input. The Move-to-front method is 4-competitive, means it never does more than a factor of 4 operations than offline algorithm (See [the MIT video lecture](#) for proof).

We will soon be discussing implementation and proof of the analysis given in the video lecture.

#### References:

[http://en.wikipedia.org/wiki/Self-organizing\\_list](http://en.wikipedia.org/wiki/Self-organizing_list)

[MIT Video Lecture](#)

[http://www.eecs.yorku.ca/course\\_archive/2003-04/F/2011/2011A/DatStr\\_071\\_SOLists.pdf](http://www.eecs.yorku.ca/course_archive/2003-04/F/2011/2011A/DatStr_071_SOLists.pdf)

[http://en.wikipedia.org/wiki/Competitive\\_analysis\\_\(online\\_algorithm\)](http://en.wikipedia.org/wiki/Competitive_analysis_(online_algorithm))

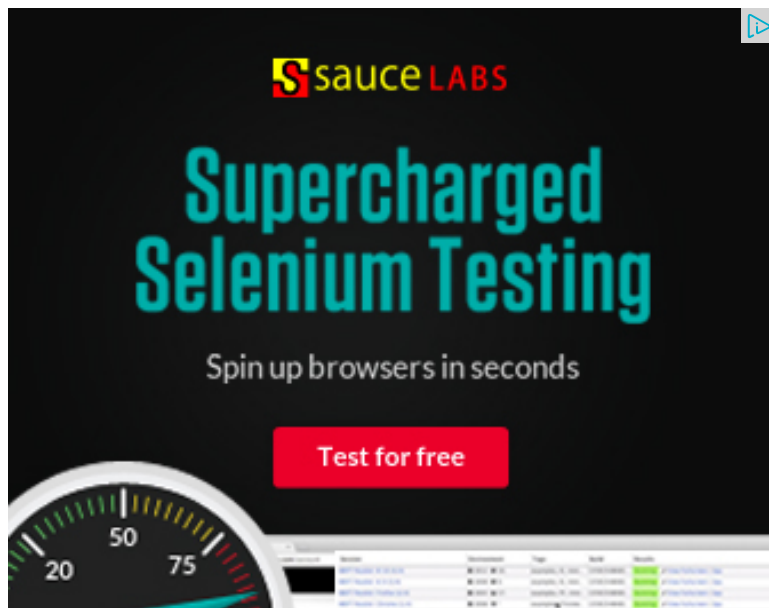
This article is compiled by **Abhay Rathi**. Please write comments if you find anything incorrect, or you want to share more information about the topic discussed above



## Popular Posts

[All permutations of a given string](#)

[Memory Layout of C Programs](#)



Memory Layout of C Programs

Understanding "extern" keyword in C

Median of two sorted arrays

Tree traversal without recursion and without stack!

Structure Member Alignment, Padding and Data Packing

Intersection point of two Linked Lists

Lowest Common Ancestor in a BST.

Check if a binary tree is BST or not

Sorted Linked List to Balanced BST

## Related Tpoics:

- Given a linked list, reverse alternate nodes and append at the end
- Pairwise swap elements of a given linked list by changing links
- Merge a linked list into another linked list at alternate positions
- QuickSort on Singly Linked List
- Delete N nodes after M nodes of a linked list
- Design a stack with operations on middle element
- Swap Kth node from beginning with Kth node from end in a Linked List
- QuickSort on Doubly Linked List



18

Tweet

2

1

Writing code in comment? Please use [ideone.com](https://ideone.com) and share the link here.

1 Comment

GeeksforGeeks

Sort by Newest ▼



Join the discussion...



**Prerit Sharma** · 7 months ago

```
#include<iostream>
```

```
using namespace std;
```

```
class SelfOrganisedList{
```

```
private:
```



```
struct node{  
  
    int data;  
  
    struct node *next;  
  
};  
  
node *head;
```

[see more](#)

^ | v • Reply • Share ›

 [Subscribe](#)

 [Add Disqus to your site](#)

 705



[Subscribe](#)

## Recent Comments

Abhi You live US or India?

[Google \(Mountain View\) interview](#) · 47 minutes ago

**Aman** Hi, Why arent we checking for conditions...

[Write a C program to Delete a Tree.](#) · 1 hour ago

kzs please provide solution for the problem...

[Backtracking | Set 2 \(Rat in a Maze\)](#) · 1 hour ago

**Sanjay Agarwal** bool

tree::Root\_to\_leaf\_path\_given\_sum(tree...


Root to leaf path sum equal to a given number · 1 hour ago

**GOPI GOPINATH** @admin Highlight this sentence "We can easily...

[Count trailing zeroes in factorial of a number](#) · 1 hour ago

**newCoder3006** If the array contains negative numbers also. We...


[Find subarray with given sum](#) · 2 hours ago

AdChoices 

[▶ C++ Code](#)

[▶ Java Source Code](#)


[▶ Programming C++](#)

AdChoices 

► [Binary Tree C++](#)

► [Java to C++](#)

► [C++ Array](#)

AdChoices 

► [C++ Array](#)

► [C++ Empty Queue](#)

► [C++ Data String](#)

@geeksforgeeks, **Some rights reserved**

**Contact Us!**

Powered by **WordPress** & **MooTools**, customized by geeksforgeeks team