International Journal of Science and Research (IJSR)
ISSN (Online): 2319-7064
Index Copernicus Value (2013): 6.14 | Impact Factor (2013): 4.438

A Search Algorithm "My-Search" To Find Elements

Bishnu Charan Behera

MSc (Mathematics & Computing), ISM Dhanbad, India

Abstract: ***You must have read searching techniques like linear search, binary search, etc which is being taught in data structures to find an element in an array. But this search is a new way different from the above mentioned .you will get details as you proceeds. This is a algorithm which has the same time complexity as that of linear search of "o(n)".but still it is better than "linear search" in terms of execution time. Let a[ ] be the array of some size n. If the element which we want to search is at any position before "n/2" than "my-search and linear-search" both will have execution time, but the magic happens when the search element is after "n/2" position. Suppose the element want to search is at nth position, then using the linear search will find the element after nth iteration, but using "my-search" we can search the element after 1st iteration itself. Elements in (n-i)th position can be found in the (i+1)th iteration i.e., suppose size is 1000 than element in 1000th position can be found in 1st iteration, similarly 999 in 2nd iteration and process goes on like this. When we are dealing with a situation when size is something 10 or 15 its ok. But can you imagine the case when the size is "100000000" or equivalent. If we use this "linear search" technique than the total expenditure you can think off to continue the loop for 100000000 times. But rather if we use "my-search", we get the desired search just after 1 iterations. So, now can you imagine how we can prevent such a big loss through "my-search".***

1. Algorithm

This is an algorithm which i have developed and named as "my –search". It will search a required element from the array. The algorithm is like this:

Algo-my search

```
var:a[ ],n,i,j,c,item;
i=1,j=n,c=0;
while((i<n/2)||(j>=n/2))
{
if(a[i]==item)
{
write"element found";
c++;
exit();
}
else if(a[j]==item)
{
write"element found";
c++;
exit();
}
i++;
j--;
}
If(c==0)
{
write"element not found";
}
```

2. Conclusion

Hence from the above algorithm we concluded that the above algorithm is better than "linear search" algorithm in terms of execution. Using this we can make searching techniques less complex.

3. Future Needs

As i mentioned above this makes the searching techniques less complex, so it can be widely be used in industrial purpose. We know that in industries time matters more than anything which can be resolved by this "my-search". Finding element from thousands of entries is not an easy task and we can do it easily.

Acknowledgment

My profound gratitude goes to my teachers who taught me the computer programming course in the HSC level. Also thanks to teachers of my bachelors and masters programs. Last but not the least to my parents without whom this would not have possible, many thanks to them for being so cooperative to me in writing this.

Authors Profile

Bishnu Charan Behera is currently in the first year of MSc mathematics & computing at ism Dhanbad. He did his BSc in mathematics & computing from institute of mathematics & applications, Bhubaneswar. His areas of research are graph theory, algorithms and numerical methods.

References

[1] Data structures and algorithms, by Springer verlag,1996.isbn:3540760474
[2] Introduction to algorithms, by mit press/triliteral,2001.isbn:0262032937
[3] Data structures and their algorithms, by addison-wesley pub co.,1997,isbn:067339736x
[4] Data structures with c(schaum's outline series)
[5] Algorithms + data structures = programs by prentice halls,1985,isbn:0130224189
[6] International journal of data structures and algorithms
[7] An introduction to analysis of algorithms, by robert sedqewick.isbn-13:978-0321905758