Worst time of sequential search (in ordered array and in unordered array)

Example

Sequential search, unordered.

Find an item x in an unordered array I based only on comparisons of x to elements of I.

I[0], I[1], I[2], ..., I[k − 1], I[k], ..., I[n − 1]

OR

Sequential search, ordered.

Find an item x in an ordered array I based only on comparisons of x to elements of I.

I[0] ≤ I[1] ≤ I[2] ≤ ... ≤ I[k − 1] ≤ I[k] ≤ ... ≤ I[n − 1]

Note. A search can be successful if the element x that is searched for is found, of unsuccessful otherwise. In the case of successful search in an ordered array, the sequential search algorithm does not take advantage of the order. In the case of unsuccessful search in an ordered array, the sequential search algorithm does take advantage of the order because it halts after encoutering and element of I that is larger than x.

Notation

size(I) − number of elements to be searched.

T(n) − number of comparisons performed while searching of an entry in an n − element array I.

Worst − case (e.g., successful search for the last element

x = I[n − 1]

or unsuccessful search past the last element

x > I[n − 1]

− works both for unordered and ordered search) running time

T(n) = n

Optimality

For an unordered array, the sequential search is worst − case optimal in the class of algorithms that search by comparisons of keys.

Proof. Suppose that there is an algorithm S that searches by comparisons of keys that is capable of finding an element x in an n – element array performing at less than n comparisons

Here is how you can play an adversary strategy in order to make the algoritm S fail.

1. Give S an array A of n elements that are all greater than 0 and a number 0 to search for.

2. Every time that S compares the given 0 to an element of A, say, to A[i], you answer "Not equal".

3. If S stops and says "0 is not there", you assign 0 to any element A[j] that the algorithm S did not compare 0 to (there must be such an element since there are n elements of A and S made less than n comparisons) and say "Wrong answer, look, A[j]=0".

4. If the algoritm S stops and says "0 is there", you say "Wrong answer, look, all elements of A are greater than 0".

In any case (3 or 4), S will give a wrong answer. **End of proof**.

For an **ordered** array, the sequential search is **NOT** worst– case optimal in that class. For instance, binary search (that searches by comparisons of keys) performs less comparisons in the worst case than sequential search does.