Selection Sort

Implementation:

```
void selection_sort(int arr[],int n)
{
   int i,j,min,temp;
   for(i=0; i<n-1; i++)
   {
      min = i;
      for(j=i+1; j<n; j++)
      {
         if(arr[j]<arr[min])
         {
            min = j;
         }
      }
      if(min!=i)
      {
         temp = arr[i];
         arr[i] = arr[min];
         arr[min] = temp;
      }
   }
}
```

Analysis:

We start analyzing the selection Sort procedure with the time "cost" of each statement and the number of times each statement is executed.

pseudo code of Selection sort:

		Cost	Time
1	for *i* <- 1 to *n-1*	c1	n
2	do *min* <- *i*	c2	n-1
3	for *j* <-i+1 to n		
4	if A[j] < *A*[*min*]	c3	$\frac{n(n+1)}{2}$
5	do *min* <-*j*		
6	if *min*! = *i*	c4	n-1
7	swap(*A*[*i*],A[min]	c5	n-1

Md. Foysal Ahmed
ID: 191-15-12486

For each i= 1,2,3……n, where n is the size of the array. When a for or while loop exits in the usual way (i.e., due to the test in the loop header), the test is executed one time more than the loop body.
The running time of the algorithm is the sum of running times for each statement executed; a statement that takes c_i steps to execute and executes n times will contribute $c_i * n$ times to the total running time.

If the array is in reverse sorted order—that is, in decreasing order the **worst case** occurs. For each value of i, the inner loop will execute for i+1 to n times.
Imagine Initially i=1,

j = 2 and it will run 2 to n = n-1 times
j = 3 and it will run 3 to n = n-2 times
j = 4 and it will run 4 to n = n-3 times
.
.
.
j = n-1 and It will run n-1 to n = 1 times

Therefore the total execution time for inner loop will be $\frac{n(n-1)}{2}$ which is written in line 4.

Therefore, the running time of SELECTION-SORT is,

$$T(n) = c_1 n + c_2(n-1) + +c_3(\tfrac{n(n-1)}{2}) + c_4(\mathrm{n}-1) + c_5(n-1)$$
$$= c_1 n + c_2 n - c_2 + \tfrac{c_3}{2}(n^2 - n) + c_4 n - c_4 + c_5 n - c_5$$
$$= c_1 n + c_2 n - c_2 + \tfrac{c_3}{2} n^2 - \tfrac{c_3}{2} n + c_4 n - c_4 + c_5 n - c_5$$
$$= \tfrac{c_3}{2} n^2 + (c_1 + c_2 - \tfrac{c_3}{2} + c_4 + c_5) n + (-c_2 - c_4 - c_5)$$

This equation can be written as,

$T(n)$ = $an^2 + bn + c$ (where a ,b & c are some constant) **which is an quadratic function.**

In selection sort, the **Best case** occurs if the array is already sorted. Though in the inner loop the if statement will execute for 0 times but the for each value of i , the inner loop will execute for total $\frac{n(n-1)}{2}$ times.
Therefore, the time function will remain same as the worst case which is

$T(n)$ = $an^2 + bn + c$ (where a ,b & c are some constant) **which is an quadratic function.**

Md. Foysal Ahmed
ID: 191-15-12486

Time Complexity:

Worst Case:

5	4	3	2	1

For reversely sorted array elements, if there is n elements, for each value of i, the inner loop will be executed,

(n-1)+(n-2)+(n-3)+……..+3+1+1

$= \frac{(n-1)(n-1+1)}{2}$

$= \frac{n(n-1)}{2}$ which is a polynomial equation

Considering the highest order of n, the complexity of selection sort in worst case is $O(n^2)$

Best Case: The time function we get from the analysis is $T(n) = an^2 + bn + c$

So the complexity of selection sort in best case is $\Omega(n^2)$

Md. Foysal Ahmed
ID: 191-15-12486