Quick sort

It is an algorithm of Divide & Conquer type.

Divide: Rearrange the elements and split arrays into two sub-arrays and an element in between search that each element in left sub array is less than or equal to the average element and each element in the right sub- array is larger than the middle element.

Conquer: Recursively, sort two sub arrays.

Combine: Combine the already sorted array.

Algorithm:

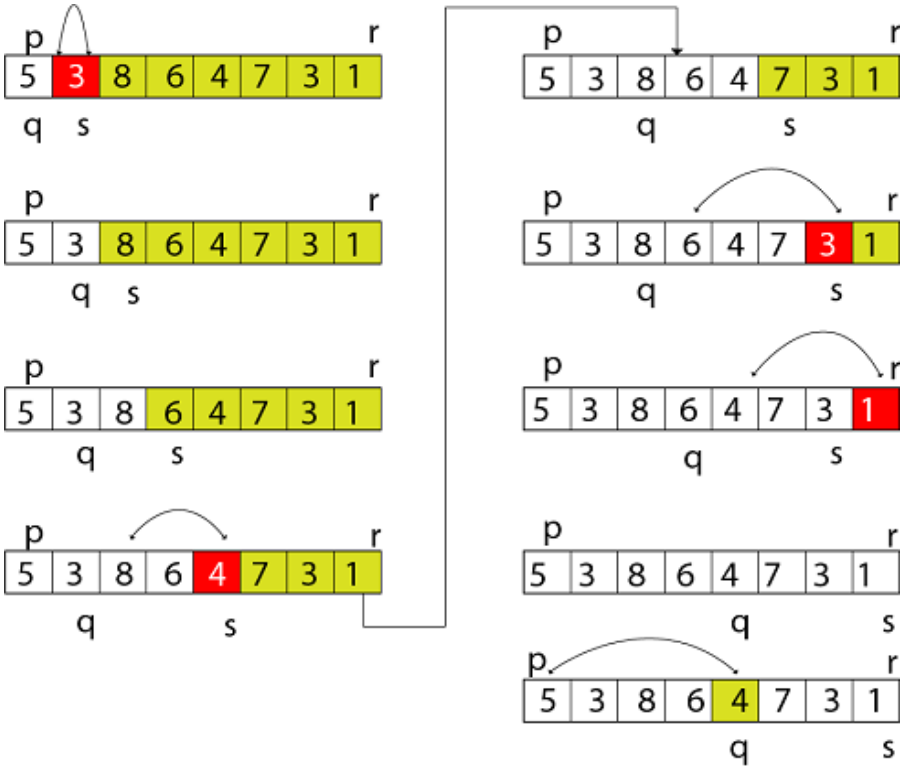
```
QUICKSORT (array A, int m, int n)
 1 if (n > m)
 2 then
 3 i ← a random index from [m,n]
 4 swap A [i] with A[m]
 5 o ← PARTITION (A, m, n)
 6 QUICKSORT (A, m, o - 1)
 7 QUICKSORT (A, o + 1, n)
```

Partition Algorithm:

Partition algorithm rearranges the sub arrays in a place.

```
PARTITION (array A, int m, int n)
 1 x ← A[m]
 2 o ← m
 3 for p ← m + 1 to n
 4 do if (A[p] < x)
 5 then o ← o + 1
 6 swap A[o] with A[p]
 7 swap A[m] with A[o]
 8 return o
```

Figure: shows the execution trace partition algorithm



Example of Quick Sort:

```
44  33  11  55  77  90  40  60  99  22  88
```

Let **44** be the **Pivot** element and scanning done from right to left

Comparing **44** to the right-side elements, and if right-side elements are **smaller** than **44**, then swap it. As **22** is smaller than **44** so swap them.

```
22    33    11    55    77    90    40    60    99    44    88
```

Now comparing **44** to the left side element and the element must be **greater** than 44 then swap them. As **55** are greater than **44** so swap them.

```
22    33    11    44    77    90    40    60    99    55    88
```

Recursively, repeating steps 1 & steps 2 until we get two lists one left from pivot element **44** & one right from pivot element.

22	33	11	**40**	77	90	**44**	60	99	55	88

Swap with 77:

22	33	11	40	**44**	90	**77**	60	99	55	88

Now, the element on the right side and left side are greater than and smaller than **44** respectively.

Now we get two sorted lists:

22 33 11 40 **44** 90 77 66 99 55 88

Sublist1 **Sublist2**

And these sublists are sorted under the same process as above done.

These two sorted sublists side by side.

22 33 11 40 **44** **90** 77 60 99 55 88

11 33 **22** 40 **44** 88 77 60 99 55 **90**

11 **22** 33 40 **44** 88 77 60 **90** 55 **99**

First sorted list

88 77 60 **55** **90** 99

Sublist3 **Sublist4**

55 77 60 **88** 90 99

Sorted

55 **77** **60**

55 60 77

Sorted

Merging Sublists:

11	**22**	**33**	**40**	**44**	**55**	**60**	**77**	**88**	**90**	**99**

SORTED LISTS

Worst Case Analysis: It is the case when items are already in sorted form and we try to sort them again. This will takes lots of time and space.

Equation:

$$T(n) = T(1) + T(n-1) + n$$

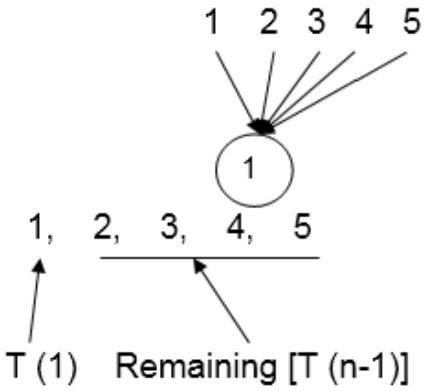
T (1) is time taken by pivot element.

T (n-1) is time taken by remaining element except for pivot element.

N: the number of comparisons required to identify the exact position of itself (every element)

If we compare first element pivot with other, then there will be 5 comparisons.

It means there will be n comparisons if there are n items.



Relational Formula for Worst Case:

T (n) =T (1) + T (n-1) + n..... (1)

T (n-1) =T (1) +T (n-1-1) + (n-1)

By putting (n-1) in place of n in equation1

Put T (n-1) in equation1

T(n)=T(1)+T(1)+(T(n-2)+(n-1)+n.....(ii)

T (n) =2T (1) +T (n-2) + (n-1) +n

T (n-2) =T (1) +T (n-3) + (n-2)

By putting (n-2) in place of n in equation1

Put T (n-2) in equation (ii)

T(n)=2T(1)+T(1)+T(n-3)+(n-2)+(n-1)+n

T(n)=3T(1)+T(n-3)+)+(n-2)+(n-1)+n

T (n-3) =T (1) +T (n-4) +n-3

By putting (n-3) in place of n in equation1

T(n)=3T(1)+T(1)+T(n-4)+(n-3)+(n-2)+(n-1)+n

=4T(1)+T(n-4))+(n-3)+(n-2)+(n-1)+n.....(iii)

> **Note: for making T (n-4) as T (1) we will put (n-1) in place of '4' and if We put (n-1) in place of 4 then we have to put (n-2) in place of 3 and (n-3) In place of 2 and so on.**

T(n)=(n-1) T(1) + T(n-(n-1))+(n-(n-2))+(n-(n-3))+(n-(n-4))+n

T (n) = (n-1) T (1) + T (1) + 2 + 3 + 4+.....n

T (n) = (n-1) T (1) +T (1) +2+3+4+.....+n+1-1

[Adding 1 and subtracting 1 for making AP series]

T (n) = (n-1) T (1) +T (1) +1+2+3+4+..... + n-1

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

Stopping Condition: T (1) =0

Because at last there is only one element left and no comparison is required.

T (n) = (n-1) (0) +0+ $\frac{n(n+1)}{2}$ -1

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

$T(n) = O(n^2)$

Avoid all the terms expect higher terms n^2

Worst Case Complexity of Quick Sort is T (n) =O (n^2)

Randomized Quick Sort [Average Case]:

Generally, we assume the first element of the list as the pivot element. In an average Case, the number of chances to get a pivot element is equal to the number of items.

Let total time taken =T (n)

For eg: In a given list

p 1, p 2, p 3, p 4.....pn

If p 1 is the pivot list then we have 2 lists.

I.e. T (0) and T (n-1)

If p2 is the pivot list then we have 2 lists.

I.e. T (1) and T (n-2)

p 1, p 2, p 3, p 4.....pn

If p3 is the pivot list then we have 2 lists.

I.e. T (2) and T (n-3)

p 1, p 2, p 3, p 4.....p n

So in general if we take the **Kth** element to be the pivot element.

Then,

$$T(n) = \sum_{K=1}^{n} T(K-1) + T(n-k)$$

Pivot element will do n comparison and we are doing average case so,

$$T(n) = n+1 + \frac{1}{n}\left(\sum_{K=1}^{n} T(K-1) + T(n-k)\right)$$

N comparisons — Average of n elements

So Relational Formula for Randomized Quick Sort is:

$$T(n) = n+1 + \frac{1}{n}\left(\sum_{K=1}^{n} T(K-1) + T(n-k)\right)$$

$$= n+1 + \frac{1}{n}\left(T(0)+T(1)+T(2)+\ldots T(n-1)+T(n-2)+T(n-3)+\ldots T(0)\right)$$

$$= n+1 + \frac{1}{n} x2 \left(T(0)+T(1)+T(2)+\ldots T(n-2)+T(n-1)\right)$$

$n\,T(n) = n(n+1) + 2\,(T(0)+T(1)+T(2)+\ldots T(n-1)$.....eq 1

Put n=n-1 in eq 1

$(n-1)\,T(n-1) = (n-1)\,n + 2\,(T(0)+T(1)+T(2)+\ldots T(n-2)$.....eq2

From eq1 and eq 2

$n\,T(n) - (n-1)\,T(n-1) = n(n+1)-n(n-1)+2\,(T(0)+T(1)+T(2)+?T(n-2)+T(n-1))-2(T(0)+T(1)+T(2)+\ldots T(n-2))$

$n\,T(n) - (n-1)\,T(n-1) = n[n+1-n+1]+2T(n-1)$

$n\,T(n) = [2+(n-1)]T(n-1)+2n$

$n\,T(n) = n+1\;T(n-1)+2n$

$$\frac{n}{n+1}T(n) = \frac{2n}{n+1} + T(n-1)$$ [Divide by n+1]

$$\frac{1}{n+1}T(n) = \frac{2}{n+1} + \frac{T(n-1)}{n}$$ [Divide by n] .....eq 3

Put n=n-1 in eq 3

$$\frac{1}{n}T(n-1) = \frac{2}{n} + \frac{T(n-2)}{n-1}$$.....eq4

Put 4 eq in 3 eq

$$\frac{T(n)}{n+1} = \frac{2}{n+1} + \frac{2}{n} + \frac{T(n-2)}{n-1}$$.....eq5

Put n=n-2 in eq 3

$$\frac{T(n-2)}{n-1} = \frac{2}{n-1} + \frac{2}{n} + \frac{T(n-3)}{n-2}$$.....eq6

Put 6 eq in 5 eq

$$\frac{T(n)}{n+1} = \frac{2}{n+1} + \frac{2}{n} + \frac{2}{n-1} + \frac{T(n-3)}{n-2} \ldots\ldots\ldots \text{eq7}$$

Put n=n-3 in eq 3

$$\frac{T(n-3)}{n-2} = \frac{2}{n-2} + \frac{T(n-4)}{n-3} \ldots\ldots\ldots\ldots \text{eq8}$$

Put 8 eq in 7 eq

$$\frac{T(n)}{n+1} = \frac{2}{n+1} + \frac{2}{n} + \frac{2}{n-1} + \frac{2}{n-2} + \frac{T(n-4)}{n-3} \ldots\ldots\ldots\ldots \text{eq9}$$

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

3 terms of $\frac{2}{n+1} + \frac{2}{n} + \frac{2}{n-1} = T(n-3)$

4 terms of $\frac{2}{n+1} + \frac{2}{n} + \frac{2}{n-1} + \frac{2}{n-2} = T(n-4)$

From 3eq, 5eq, 7eq, 9 eq we get

$$\frac{T(n)}{n+1} = \frac{2}{n+1} + \frac{2}{n} + \frac{2}{n-1} + \ldots\ldots\ldots + \frac{2}{3} + \frac{T(n-(n-1))}{n-(n-2)} \ldots\ldots\ldots \text{eq10}$$

From 3 eq $\frac{T(n)}{n+1} = \frac{2}{n+1} + \frac{T(n-1)}{n}$

Put n=1

$$\frac{T(1)}{2} = \frac{2}{2} + \frac{T(0)}{1}$$

$$\frac{T(1)}{2} = 1$$

From 10 eq

$$\frac{T(n)}{n+1} = \frac{2}{n+1} + \frac{2}{n} + \frac{2}{n-1} + \ldots\ldots\ldots\ldots + \frac{2}{3} + \frac{T(1)}{2}$$

$$= \frac{2}{n+1} + \frac{2}{n} + \frac{2}{n-1} + \ldots\ldots\ldots\ldots + \frac{2}{3} + 1$$

Multiply and divide the last term by 2

$=\frac{2}{n+1}+\frac{2}{n}+\frac{2}{n-1}+\ldots\ldots\ldots\ldots\ldots+\frac{2}{3}+2\times\frac{1}{2}$

$=2\left[\frac{1}{2}+\frac{1}{3}+\frac{1}{4}\ldots\ldots+\frac{1}{n}+\frac{1}{n+1}\right]$

$=2\sum_{2\le k\le n+1}^{n}\frac{1}{k}=2\int_{2}^{n+1}\frac{1}{k}$

Multiply & divide k by n

$=2\int_{2}^{n+1}\frac{1}{\frac{kn}{n}}$

Put $\frac{k}{n}=x$ and $\frac{1}{n}=dx$

$\frac{T(n)}{n+1}=2\int_{2}^{n+1}\frac{1}{x}\,dx$

$=2\log x\int_{2}^{n+1}$

NOTE: $\int\frac{1}{x}dx=\log x$

$=2[\log(n+1)-\log 2]$

$T(n)=2(n+1)[\log(n+1)-\log 2]$

Ignoring Constant we get

$T(n)=n\log n$

$T(n)=O(n\log n)$

Is the average case complexity of quick sort for sorting n elements.

3. Quick Sort [Best Case]: In any sorting, best case is the only case in which we don't make any comparison between elements that is only done when we have only one element to sort.

Method Name	Equation	Stopping Condition	Complexities
1.Quick Sort[Worst Case]	$T(n)=T(n-1)+T(0)+n$	$T(1)=0$	$T(n)=n^2$
2.Quick Sort[Average Case]	$T(n)=n+1+\frac{1}{n}(\sum_{K=1}^{n}T(k-1)+T(n-k))$		$T(n)=n\log n$

← Prev

Next →