Tuesday, 12 March 2024 6:04 PM

Problem Statement

Note: Pair nuts & bolts.

Take a nut n_1 and find its b_1

→ $n_1 = b_1$

→ $n_1 < b_1$

→ $n_1 > b_1$

Given n nuts and n bolts,
$n \cdot n$ nuts + $n \cdot n$ bolts

→ Also, for a given nut, pair is being to find in linearmanner

→ Seek the nuts & bolts.

$n \times n$ of smaller nuts → $n \times n$ smaller bolts

→ We need to compare n bolts with n bolts

w/ N nuts always be smaller than n bolts

→ Divide & conquer.

② Pick a pivot element

partitioning

Time complexity

① Best case

$T(n) = 2T(\frac{n}{2}) + O(n)$ [Best case]

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

Worst case:

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

$T(n) \approx O(n^2)$

pivot → last ele
→ first ele
→ median of three array

Merge Sort

Required extra space to merge

Randomized quick sort

Quick sort is different from other sorting algo because it can be randomized.

What is a randomized algo??

In a normal algo,

In a randomized algo,

Counting Sort → $O(n)$

count the occurrences of each unique ele of the array

arr → 3 2 2 2 3 3 1

① Find the max ele of the array.

max → 3

② Make a new array of size (n+1)

Initialize all values to 0

count → 0 1 2 3 4 5 6 7 8

0 1 2 3 4 5 6 7 8

③ Store frequency of every ele in count array

④ Store cumulative counts.

count → 0 1 3 5 6 6 6 6 7

0 1 2 3 4 5 6 7 8

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

⑤ Find index of each ele in the sorted array after sorting

array → 3 2 2 2 3 3 1

count → 0 1 3 5 6 6 6 7

0 1 2 3 4 5 6 7 8

output → 1 2 2 3 3 3 3

0 1 2 3 4 5 6 7

Time complexity →

Worst case → $O(n + \frac{n}{2} + n)$ → $O(n)$

Best case → $O(n+n)$ ≈ $O(n)$

Avg case → $O(n+n)$ ≈ $O(n)$

K → max ele.

Space complexity

$SC \rightarrow O(max)$

Applications

① Smaller integer with multiple occurrences

② Linear complexity

Ques: Given an array of size (n), and values (n), k < size of array.

→ find the kth smallest ele of the given array.

Eg: arr [] → {7, 10, 4, 3, 20, 15}

k = 3

ans = 7

Eg → {8, 5, 1, 3, 23, 37, 6}

k = 5 ans → 23