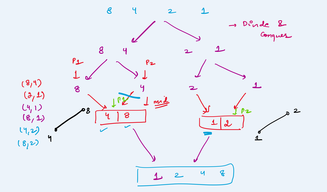


Quick Sort

Q Problem statement

Nuts → N
Bolts → B

→ Nut fits into a bolt
→ If two nuts N_1 & N_2 are not comparable

→ Similar for bolts as well

→ If you take a nut n_1 & bolt B_1

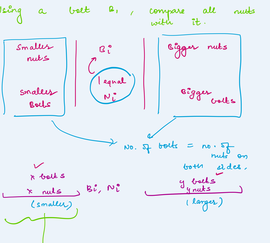
→ $N_1 = B_1$
→ $N_2 < B_2$
→ $N_1 > B_2$

Given N nuts & B bolts,

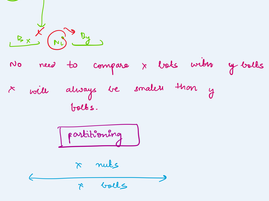
→ Also for every nut, there is a corresponding bolt present.

→ Sort the nuts & bolts.

Using a bolt B_1, compare all nuts with it.

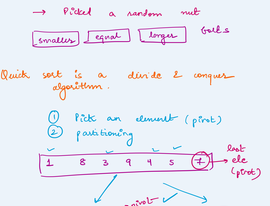


no. of bolts = no. of nuts on both sides.



→ No need to compare x bolts with y bolts

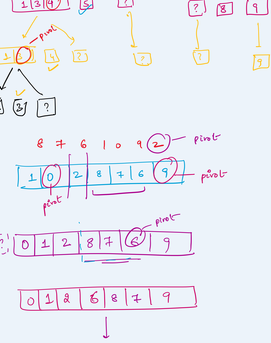
→ x will always be smaller than y bolts.

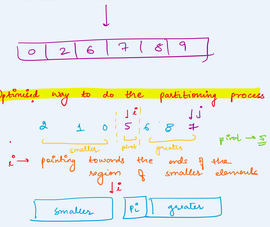


→ Picked a random nut

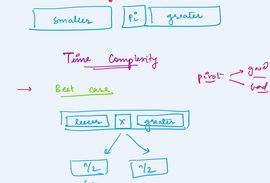
Quick sort is a divide & conquer algorithm.

1. Pick an element (pivot)
2. partitioning

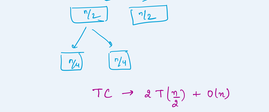




Optimized way to do the partitioning process

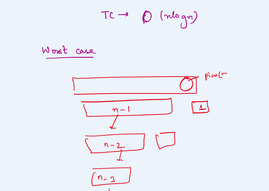


i → pushing towards the ends of the region of smaller elements



Time Complexity

→ Best case

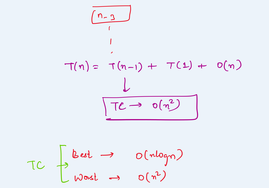


pivot → $O(n^2)$
→ Good

$$TC \rightarrow 2T\left(\frac{n}{2}\right) + O(n)$$

$$TC \rightarrow O(n \log n)$$

Worst case



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

$$TC \rightarrow O(n^2)$$

TC → Best → $O(n \log n)$
→ Worst → $O(n^2)$

Space Complexity → $O(1)$

Quick sort	Merge Sort
→ Worst → $O(n^2)$	Worst → $O(n \log n)$
→ SC → $O(1)$	SC → $O(n)$

For next class

→ Randomized quick sort

→ Count Sort