Design and Analysis of Algorithm (DAA)

Quicksort

Dr. Dayal Kumar Behera

School of Computer Engineering
KIIT Deemed to be University, Bhubaneswar, India

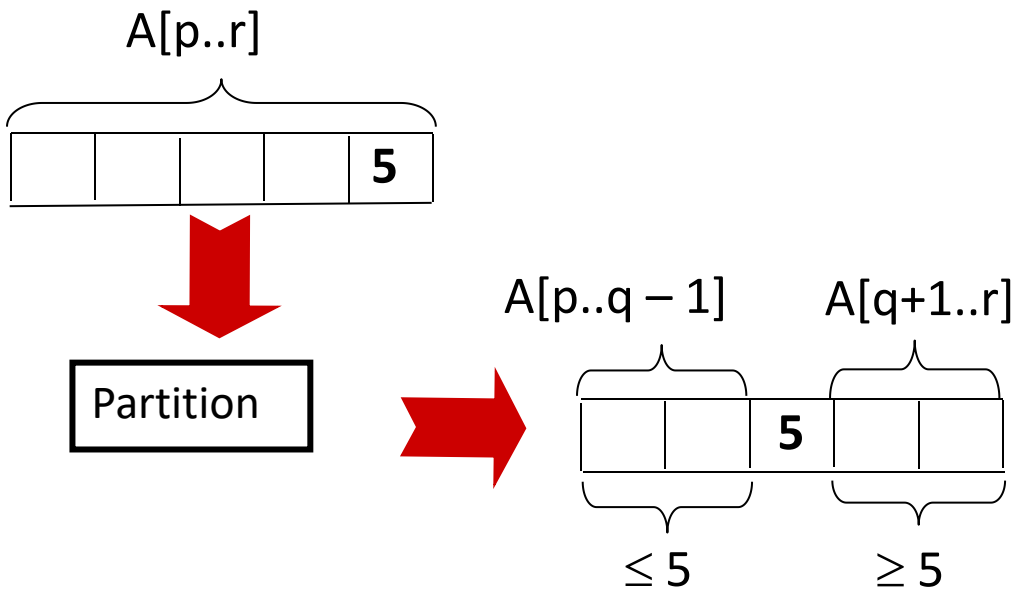
Quicksort

The ***Quicksort*** algorithm is based on divide-and-conquer paradigm. To sort the sub array A[p……r]

- **Divide:** Partition (rearrange) the array A[p…r] into two subarrays A[p … q-1] and A[q+1 … r] such that each element of A[p … q-1] is less than or equal to A[q] (pivot) and each elements of A[q+1 … r] is greater than or equal to A[q] (pivot). Compute the index q as part of this partitioning procedure..

- **Conquer:** Sort the two subarrays A[p … q-1] and A[q+1 … r] by recursive calls to quicksort.
- **Combine:** No work is needed to combine the subarrays (Quicksort operates in-place on the data to be sorted).

Quicksort



Algorithm

```
QUICKSORT(A, p, r)
    if p < r then
        q := PARTITION(A, p, r);
        QUICKSORT(A, p, q – 1);
        QUICKSORT(A, q + 1, r)
```

```
PARTITION(A, p, r)
    x := A[r]; i:= p – 1;
    for j := p to r – 1 do
        if A[j] ≤ x then
            i := i + 1;
            swap A[i] ↔ A[j]
    swap A[i + 1] ↔ A[r];
    return i + 1
```

Analysis of Quicksort

Randomized Quicksort

- Any element in the sub-array can be chosen as pivot.
- Instead of always using A[r] as the pivot, we can randomly chose element from the subarray A[p … r] as pivot.

Randomized Quicksort

```
RANDOMIZED-PARTITION(A, p, r)
1  i = RANDOM(p, r)
2  exchange A[r] with A[i]
3  return PARTITION(A, p, r)
```

```
RANDOMIZED-QUICKSORT(A, p, r)
1  if p < r
2       q = RANDOMIZED-PARTITION(A, p, r)
3       RANDOMIZED-QUICKSORT(A, p, q − 1)
4       RANDOMIZED-QUICKSORT(A, q + 1, r)
```


Each of your
actions will
have an
impact on your
future.


Once you know
who is walking
with you on your path.
you will never
be afraid.

Thank you