COURSE HANDOUT

Course Code	ACSC13
Course Name	Design and Analysis of Algorithms
Class / Semester	IV SEM
Section	A-SECTION
Name of the Department	CSE-CYBER SECURITY
Employee ID	IARE11023
Employee Name	Dr K RAJENDRA PRASAD
Topic Covered	Quick Sort
Course Outcome/s	Analysis of quicksort for unsorted elements
Handout Number	18
Date	19 April, 2023

Content about topic covered: Quicksort

In Quicksort, the division in to two sub arrays is made so that the sorted sub arrays need not to be merged later. This is accomplished by rearranging the elements in a[1:n] such that a[i] ≤ a[j] for all i between 1 and m and for all j between m+1 and n for some m, 1≤m≤n.

Thus the elements in a[1:m] and a[m+1:n] can be independently sorted. No merge is needed.

The rearrangement of the elements is accomplished by picking some element of a[ ], say t = a[s], and then re ordering the other elements so that all the elements appearing before t in a[1:n] are less than or equal to t and all elements appearing after t are greater than or equal to t. This rearranging is called partitioning.

(1)	(2)	(3)	(4)	(5)	(6)	(7)	(8)	(9)	(10)	i	p
65	70	75	80	85	60	55	50	45	$+\infty$	2	9
65	45	75	80	85	60	55	50	70	$+\infty$	3	8
65	45	50	80	85	60	55	75	70	$+\infty$	4	7
65	45	50	55	85	60	80	75	70	$+\infty$	5	6
65	45	50	55	60	85	80	75	70	$+\infty$	6	5
60	45	50	55	65	85	80	75	70	$+\infty$		

```
Algorithm Partition(a, m, p)
// Within a[m], a[m + 1], ..., a[p − 1] the elements are
// rearranged in such a manner that if initially t = a[m],
// then after completion a[q] = t for some q between m
// and p − 1, a[k] ≤ t for m ≤ k < q, and a[k] ≥ t
// for q < k < p. q is returned. Set a[p] = ∞.
{
    v := a[m]; i := m; j := p;
    repeat
    {
        repeat
            i := i + 1;
        until (a[i] ≥ v);

        repeat
            j := j − 1;
        until (a[j] ≤ v);

        if (i < j) then Interchange(a, i, j);

    } until (i ≥ j);

    a[m] := a[j]; a[j] := v; return j;
}
```

```
Algorithm Interchange(a, i, j)
// Exchange a[i] with a[j].
{
    p := a[i];
    a[i] := a[j]; a[j] := p;
}
```

```
Algorithm QuickSort(p, q)
// Sorts the elements a[p], ..., a[q] which reside in the global
// array a[1 : n] into ascending order; a[n + 1] is considered to
// be defined and must be ≥ all the elements in a[1 : n].
{
    if (p < q) then  // If there are more than one element
    {
        // divide P into two subproblems.
            j := Partition(a, p, q + 1);
                 // j is the position of the partitioning element.
        // Solve the subproblems.
            QuickSort(p, j − 1);
            QuickSort(j + 1, q);
        // There is no need for combining solutions.
    }
}
```

Analysis of Quicksort:

Assume that the partitioning element v has an equal probability of being the ith smallest element, $1 \leq i \leq p\text{-}m$ in a[m:p-1].
The average time complexity of Quicksort $C_A(n)$ is described as

$$C_A(n) = n + 1 + \frac{1}{n}\sum_{1\leq k\leq n}[C_A(k-1) + C_A(n-k)] \quad \ldots \quad (1)$$

N+1 is the number of element comparisons required by partition on its first call.

$$C_A(0) = C_A(1) = 0$$

Multiplying both sides of eq. (1) by n,

$$n\,C_A(n) = n(n+1) + 2[C_A(0) + C_A(1) + \cdots + C_A(n-1)] \quad \ldots \quad (2)$$

n by 'n-1' in eq.(2)

$$(n-1)\,C_A(n-1) = (n-1)n + 2[C_A(0) + C_A(1) + \cdots + C_A(n-2)] \ldots \quad (3)$$

Subtracting eq. (3) from eq. (2)

$$n\,C_A(n) - (n-1)\,C_A(n-1) = 2n + 2C_A(n-1)$$

$$n\,C_A(n) = 2n + 2C_A(n-1) + (n-1)\,C_A(n-1)$$

$$n\,C_A(n) = 2n + (n+1)\,C_A(n-1)$$

Dividing both sides by n(n+1)

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

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

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

$$\Rightarrow \frac{C_A(n)}{(n+1)} = \frac{C_A(n-3)}{n-2} + \frac{2}{n-1} + \frac{2}{n} + \frac{2}{n+1}$$

$$\vdots$$

$$\Rightarrow \frac{C_A(n)}{(n+1)} = \frac{C_A(1)}{2} + \frac{2}{3} + \frac{2}{4} + \cdots + \frac{2}{n+1}$$

$$\Rightarrow \frac{C_A(n)}{(n+1)} = \frac{C_A(1)}{2} + 2\sum_{3\leq k\leq n+1}\frac{1}{k}$$

$$\Rightarrow \frac{\mathrm{C_A(n)}}{(\mathrm{n}+1)} = 2 \sum_{3 \le k \le n+1} \frac{1}{k}$$

Since $C_A(1) = 0$

$$\sum_{3 \le k \le n+1} \frac{1}{k} \le \int_2^{n+1} \frac{1}{x} dx = \log(n+1) - \log 2$$

$$\frac{\mathrm{C_A(n)}}{(\mathrm{n}+1)} \le 2\,[\log(n+1) - \log 2]$$

$$\mathrm{C_A(n)} \le 2\,(n+1)[\log(n+1) - \log 2]$$

$$\mathrm{C_A(n)} = O(n \log n).$$