Assignment No: 5

Title: Write a program for analysis of quick sort by using deterministic and randomized variant.

Objective: To analyze time and space complexity of quick sort by using deterministic and randomizedvariant.

Theory:

What is a Randomized Algorithm?

- An algorithm that uses random numbers to decide what to do next anywhere in its logic iscalled Randomized Algorithm..
- For example, in Randomized Quick Sort, we use random number to pick the next pivot (or werandomly shuffle the array).
- Typically, this randomness is used to reduce time complexity or space complexity in otherstandard algorithms.
- Randomized algorithm for a problem is usually simpler and **more efficient** than itsdeterministic counterpart.
- The output or the running time are functions of the input and random bits chosen .

Types of Randomized Algorithms

1. Las Vegas Algorithms

- These algorithms always produce correct or optimum result.
- Time complexity of these algorithms is based on a random value and time complexity isevaluated as expected value.
- For example, Randomized QuickSort always sorts an input array and expected worst case timecomplexity of QuickSort is O(nLogn).
- A Las Vegas algorithm fails with some probability, but we can tell when it fails. In particular,we can run it again until it succeeds, which means that we can eventually succeed with probability 1.
- Alternatively, we can think of a Las Vegas algorithm as an algorithm that runs for anunpredictable amount of time but always succeeds

2. Monte Carlo Algorithms

- Produce correct or optimum result with some probability.
- These algorithms have deterministic running time and it is generally easier to find out worstcase time complexity.
- For example Karger's Algorithm produces minimum cut with probability greater than or equal to$1/n^2$ (n is number of vertices) and has worst case time complexity as O(E).
- A Monte Carlo algorithm fails with some probability, but we can't tell when it fails.
- The polynomial equality-testing algorithm in an example of a Monte Carlo algorithm

Applications of Randomized Algorithms

- Randomized algorithms have huge applications in Cryptography.
- Load Balancing.
- Number-Theoretic Applications: Primality Testing
- Data Structures: Hashing, Sorting, Searching, Order Statistics and ComputationalGeometry.
- Algebraic identities: Polynomial and matrix identity verification. Interactive proof systems.
- Mathematical programming: Faster algorithms for linear programming, Rounding linearprogram solutions to integer program solutions

Analysis of Randomized Quick sort

The running time of quicksort depends mostly on the number of comparisons performed in all calls tothe Randomized-Partition routine. Let X denote the random variable counting the number of comparisons in all calls to Randomized-Partition.

Let zi denote the i-th smallest element of A[1..n].

Thus A[1..n] sorted is <z1, z2, ... , zn >.

Let Zij = {zi, ..., zj} denote the set of elements between zi and zj, including these elements.

Xij = I{ zi is compared to zj}.

Thus, Xij is an indicator random variable for the event that the i-thv smallest and the j-th smallestelements of A are compared in an execution of quicksort.

Number of Comparisons

Since each pair of elements is compared at most once by quicksort, the number X of comparisons isgiven by

$$X = \sum_{i=1}^{n-1} \sum_{j=i+1}^{n} X_{ij}$$

Therefore, the expected number of comparisons is

$$\mathrm{E}[X] = \sum_{i=1}^{n-1} \sum_{j=i+1}^{n} \mathrm{E}[X_{ij}] = \sum_{i=1}^{n-1} \sum_{j=i+1}^{n} \Pr[z_i \text{ is compared to } z_j]$$

Expected Number of Comparisons

Conclusion: In this way we have explored Concept of quick sort by using deterministic and randomized variant.