CS648 : Randomized Algorithms
Semester I, 2011-12, CSE, IIT Kanpur

Lecture 4 : Randomized Quick sort, Random variable, Expected Value

In this lecture, we first introduced randomized quick sort algorithm. After that we introduced some new fundamental concepts of a discrete probability space.

1 Randomized Quick Sort algorithm

Recall the quick sort algorithm you studied in your first course on algorithms. That algorithm was *deterministic* quick sort algorithm. Find below *randomized* quick sort algorithm. The randomization is used only in selecting the pivot element during each recursive call (see the colored statement in **Algorithm 1** below). At this stage, we shall assume that the computation time involved in selecting a random pivot element is $O(1)$. (Towards the end of this course, we shall look more formally on the complexity associated with the generation of random bits used by a randomized algorithm). Note that the output of this algorithm is always correct. However, the number of comparisons performed during this algorthm varies. In the worst case, it will perform $\Theta(n^2)$ comparisons.

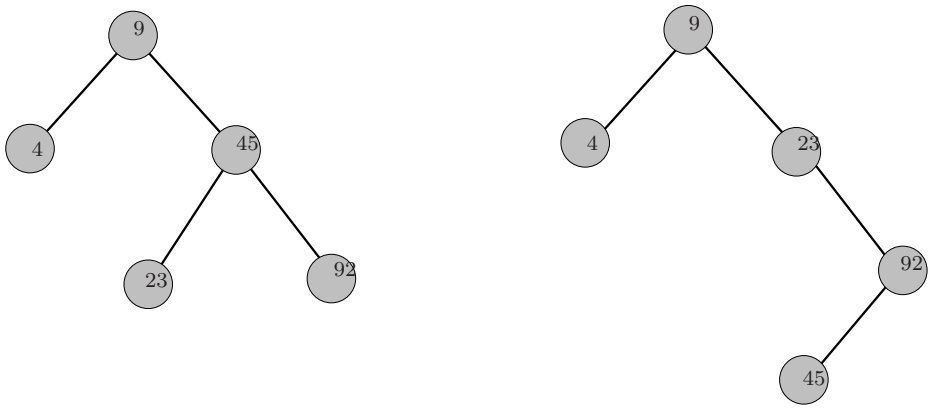
Algorithm 1: RQsort(S) : Randomized quick sort on a set S

```
if |S| ≤ 1 then return S;
Let x be an element selected randomly uniformly from S;
S_<x ← ∅;
S_>x ← ∅;
foreach y ∈ S do
    if y < x then
        add y to S_<x
    else
        add y to S_>x
RQsort(S_<x);
RQsort(S_>x);
return concatenate(S_<x, x, S_>x)
```

Homework: Compare the deterministic quick sort algorithm with the randomized quick sort algorithm described above. What advantage, if any, does the randomized quick sort have over the deterministic quick sort algorithm ?

The randomized quick sort algorithm can also be viewed as a randomized experiment. What will be its sample space ? Well, each execution of randomized quick sort can be represented by the recursion tree associated with it. This recursion tree is a rooted binary tree whose nodes define the pivot elements. So the sample space consists of all rooted binary trees of n nodes.

Homework: Does each elementary event in the sample space associated with randomized quick sort algorithm have same probability. Compute the probability associated with each of the two elementary events of randomized quick sort shown in Figure 1.



(i) (ii)

Figure 1: Two recursion trees associated with execution of randomized quick sort on set $\{4, 9, 23, 45, 92\}$.

2 Three randomized experiments

We now describe three randomized experiments which we shall analyse in this week.

- *Ball-bin experiment:* There are n balls and m bins. The bins are arranged along a line and labeled 1 to m from left to right. The experiment goes like this: each ball selects a bin randomly uniformly and independent of other balls, and falls into that bin.

  The underlying sample space Ω consists of m^n elementary events. Each elementary event is defined by the assignment of any of m bins to each of the n balls. Observe that probability of each elementary event is the same. In other words, the probability distribution is uniform.

- *Randomized quick sort:* The randomized experiment is the execution of randomized quick sort on n elements. Each possible execution of randomized quick sort can be associated with a rooted binary tree. In fact this rooted binary tree is exactly the recursion tree for that execution. As discussed earlier, the sample space consists of all such rooted binary trees on n nodes.

- *Red-blue balls out of bag:* There are r red balls and b blue balls inside a bag. We take out balls uniformly randomly out of the bag and arrange them along a line in the left to right order. Note that the sampling process imposes a distinctness among all the balls (even among those of the same color).

  The underlying sample space has $(r+b)!$ elementary events, each corresponding to some permutation of $r + b$ balls, and each elementary event occurs with the same probability.

3 Random Variable and Expectation

For any randomized experiment or randomized algorithm, we are not only interested in probability of some event. Instead, and more often, we are interested in some quantitative measure defined over the experiment or the algorithm. For example,

- For *ball-bin* problem where we throw n balls uniformly randomly and independently into n bins we may be interested in the number of empty bins
- For *randomized quick sort* on n elements, we are interested in the number of comparisons performed.
- For *red-blue balls* problem, we may be interested in the number of red balls preceding all the blue balls.

We shall now formalize this quantitative measure as follows.

Definition 3.1 *(**Random variable**)*
A random variable X is a function (or mapping) from the sample set Ω to the set of real numbers.

For a given random experiment/algorithm there may be different random variables depending upon which quantitative measure of the outcome we are interested in.

Definition 3.2 *(**Expected value**)*
Let X be a random variable defined over the probability space $(\Omega, \mathbf{P})$. Expected value of a random variable is the "average" value taken by the random variable when the randomized experiment associated with the probability space is performed "very large" number of times. We shall use $\mathbf{E}[X]$ to denote expected value of random variable X. We can express $\mathbf{E}[X]$ formally by the following equation

$$\mathbf{E}[X] = \sum_{\forall \omega \in \Omega} X(\omega)\mathbf{P}[\omega] \tag{1}$$

A random variable induces a partition of the sample space Ω in a very natural manner as follows. Let "$X = \alpha$" denote the event consisting of all elementary events $\omega \in \Omega$ for which $X(\omega) = \alpha$. Observe that X is defined for each elementary event of Ω and by definition the event "$X = \alpha$" is disjoint from "$X = \beta$" for each $\alpha \neq \beta$. Therefore the events "$X = \alpha$" for all values α taken by X define a partition of Ω. This leads to the following alternate formulation for expected value of X.

$$\mathbf{E}[X] = \sum_{\forall \alpha \in X} \alpha \mathbf{P}[X = \alpha] \tag{2}$$

Note that "$\forall \alpha \in X$" in the summation above means over all possible values α taken by the random variable X.

To develop familiarity with the above concepts, let us revisit our examples.

- For *ball-bin* problem, the number of empty bins is a random variable. We shall use X to denote this random variable henceforth.
  Exercise: Calculate $\mathbf{E}[X]$ when there are $n = 3$ balls and $m = 3$ bins.

- For *randomized quick sort* problem, the number of comparisons is a random variable. We shall use Y to denote this random variable henceforth.
  Exercise: Calculate $\mathbf{E}[Y]$ when $n = 5$ elements are to be sorted.

- For *red-blue balls* problem, the number of red balls preceding all the blue balls is a random variable. We shall use Z to denote this random variable henceforth.
  Exercise: Calculate $\mathbf{E}[X]$ when there are $r = 2$ red balls and $b = 4$ blue balls.

While solving the simple exercises given above, you just used the definition of expected value, and/or some simple case analysis. This approach works essentially because the underlying sample space to be analyzed has *few* elements and the different cases to be analyzed are also few. But, ideally one would be interested in getting a general expression for the expected value in terms of the parameters defining the randomized experiment. For example, for the *ball-bin* problem what is the expected number of empty bins in terms of the number of balls n and number of bins m. Thinking for a few minutes will convince you that the above approach of calculating expected value from definition does not seem to work for the general case. This difficulty can be explained as follows.

There are so many elementary events so that it seems infeasible to use Equation 1 here. Since the range of random variables is much smaller than the number of elementary events, we then try to use Equation 2. Here we are required to calculate probability that the random variable takes a given value. For example, for ball-bin problem, we need to know $\mathbf{P}[X = j]$, that is, the probability that there are exactly j empty bins. Likewise, for randomized quick sort, we need to know $\mathbf{P}[Y = \ell]$, that is, the probability that there are exactly ℓ comparisons during randomized quick sort. Similarly, for the red-blue balls problem, we need to know the probability $\mathbf{P}[Z = t]$, that is the probability that there are t red balls preceding all blue balls. If you want to find these probabilities, it will be quite messy (ponder over this claim and proceed only when you get convinced). If you look carefully, you will realize that it is due to the fact that the entire randomized experiment looks too complex when seen in its entirety (ponder over this statement as well). Therefore, for some moments, we abandon this idea of taking the *macroscopic/global*

view of the randomized experiment. We pursue a different approach which takes a *microscopic* view of the experiment. This approach can be expressed in loose words as follows: *Focus on individual objects or entities of the randomized experiment.* As will become clear in next two lectures, this new approach indeed works !