Chapter 1

Randomized QuickSort

Let us look at the quicksort algorithm

Algorithm 1 QuickSort

```
Input: An array $x$ of $n$ distinct elements
Output: Sorted array $y$
If $n = 1$ or 0 return $x$
$pivot = x[0]$
Create an array $x_1$ containing all elements less than pivot (in the order they appear in $x$)
Create an array $x_2$ containing all elements greater than pivot (in the order they appear in $x$).
Return the array [QuickSort($x_1$), pivot, QuickSort($x_2$)]
```

How many comparisons are done in the above algorithm? The pivot ($x[0]$) chosen at the first call to the quicksort function is compared with all the elements in the array. That is, it is compared with $n-1$ many elements. The array is now split into two parts and quick sort called individually on these arrays. Therefore,

$$\text{Number of comparisons of } x = \\ \text{Number of comparisons of } x_1 + \text{Number of comparisons of } x_2 + n - 1$$

Let us assume the input array x is in descending order. Then the array is split into two parts, one containing $n-1$ elements and the other containing no element. Thus, the total number of comparisons satisfies the following recurrence equation

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

We know that this is $O(n^2)$. Can we do better? The problem is, the pivot we choose does not create arrays x_1 and x_2 of equal size. In the above scenario, this appears in the worst form. One array of $n-1$ elements and another of 0 elements. Let us now assume, there is a "magician" who always picks a pivot which divides the arrays into two equal halves. The recurrence equation for the number of comparisons in this scenario is

$$T(n) = 2T(\lceil \frac{n}{2} \rceil) + n - 1$$

This gives us $O(n \log n)$ many comparisons. It is clear that, if we could simulate the working of a magician we have a faster algorithm. What does the magician do? He always picks the median as the pivot. How fast can you pick the median. It turns out, you can do this in linear time. The median-of-medians algorithm is a non-trivial but beautiful algorithm which can output the median in $O(n)$ steps. This will give us the following recurrence on the number of comparisons: $T(n) = 2T(\lceil \frac{n}{2} \rceil) + O(n)$. This will gives us a deterministic $O(n \log n)$ algorithm.
In this lecture, we will show that a randomized algorithm can give you a similar time. The algorithm is much simpler than the above median finding algorithm. Let us first ask the following question. If I pick a pivot randomly, what is the expected number of elements less than or equal to the pivot? Let X be the number of elements less than or equal to the pivot.

Lemma 1.1. $E[X] = \frac{n+1}{2}$.

Proof. Let $y_1, \ldots, y_n$ be the elements x_1 to x_n sorted in ascending order. Let p_i be the probability that y_i is picked as the pivot. Since any of the y_is can be picked with equal probability we have $p_i = \frac{1}{n}$, for all $i \leq n$. Then

$$\begin{aligned} E[X] = \sum_{i=1}^{n} (i \times p_i) &= \sum_{i=1}^{n} (i \times \frac{1}{n}) \\ &= \frac{1}{n} \sum_{i=1}^{n} i = \frac{1}{n} \frac{n(n+1)}{2} \\ &= \frac{n+1}{2} \end{aligned}$$

□

The above lemma shows that picking a pivot randomly, splits the array into half with high probability. This should help us get an $O(n \log n)$ algorithm for the following randomized quicksort, where the pivot is picked randomly every time.

Algorithm 2 RandQuickSort

```
Input: An array $x$ of $n$ distinct elements
Output: Sorted array $y$
If $n = 1$ or 0 return $x$
Pick a pivot randomly from the elements $x_1$ to $x_n$.
Create an array $x_1$ containing all elements less than pivot (in the order they appear in $x$)
Create an array $x_2$ containing all elements greater than pivot (in the order they appear in $x$).
Return the array [RandQuickSort($x_1$), pivot, RandQuickSort($x_2$)]
```

We now show that the expected number of comparisons made by the above Randomized Quicksort is $O(n \log n)$.

Lemma 1.2. *For any input x, the expected number of comparisons of the Randomized Quicksort given in Algorithm 2 is $O(n \log n)$.*

Proof. Let $y_1, \ldots, y_n$ be the elements sorted in ascending order. Let X be the number of comparisons. Our aim is to find out the expectation of X. Let X_{ij} be the indicator random variable where

$$X_{ij} = \begin{cases} 1, & \text{if } y_i \text{ and } y_j \text{ are compared.} \\ 0, & \text{otherwise.} \end{cases} \tag{1.1}$$

That is X_{ij} is 1 if y_i and y_j are compared and 0 otherwise. Then X the number of comparisons can be got by summing over all X_{ij}s as follows

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

By linearity of expectation we know that the expectation of X is the sum of expectation of X_{ij}s. That is

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

So, let us now try to find the expectation of X_{ij}s. Let us first find the probability that y_i and y_j will be compared. Consider the elements $Y = \{y_i, y_{i+1}, \ldots, y_j\}$. Note the following claim

Claim 1.3. *If any of y_{i+1} to y_{j-1} is selected as pivot before y_i or y_j is selected as pivot, then y_i and y_j will not be compared.*

The reason is, in this case, y_i and y_j will go into separate arrays (in the above algorithm y_i will go into x_1 and y_j will go into x_2). In other words, $X_{ij} = 0$ if an element from $Y \backslash \{y_i, y_j\}$ is selected as pivot before y_i or y_j is picked as pivot. What about the other direction.

Claim 1.4. *If y_i or y_j is selected as pivot before any of the other elements in Y are picked as pivot, then y_i and y_j will be compared.*

To understand the above claim, let p_i be picked before any of the other elements in Y were picked. Then p_i will be compared with all the other elements in the array. Since none of $Y \backslash \{y_i\}$ were picked as pivot before, all the elements in Y are in the same array as y_i. Therefore y_i is compared with all elements in Y, in particular it is compared with y_j.
From the above two claims, it follows that $X_{ij} = 1$ if and only if either y_i or y_j is selected as pivot before any other element from $Y \backslash \{y_i, y_j\}$ is selected as pivot. The question to ask now is, what is the probability for a $k \leq n$, y_k is selected as pivot before any other element from $Y \backslash \{y_k\}$ is selected as pivot. That is among Y, y_k is selected as pivot the first. Note that for any two different y_k and $y_{k'}$ in Y, this probability is the same. That is

$$\begin{aligned}\texttt{Prob}\ [y_1 \text{ is selected as pivot first }] &= \texttt{Prob}\ [y_2 \text{ is selected as pivot first }] \\ &= \cdots = \texttt{Prob}\ [y_k \text{ is selected as pivot first}] = \ldots \\ &= \texttt{Prob}\ [y_n \text{ is selected as pivot first }]\end{aligned}$$

Since they all have equal probability and because the probability that atleast one of them will be selected first in Y has probability 1, we have for all $k \leq n$,

$$\texttt{Prob}\ [y_k \text{ is selected as pivot first}] = \frac{1}{|Y|} = \frac{1}{j - i + 1}$$

We can now compute the probability that $X_{ij} = 1$.

$$\begin{aligned}\texttt{Prob}\ [X_{ij} = 1] &= \texttt{Prob}\ [y_i \text{ is selected as pivot first}] + \texttt{Prob}\ [y_j \text{ is selected as pivot first}] \\ &= \frac{2}{j - i + 1}\end{aligned}$$

For the indicator random variable X_{ij}, the expectation $E[X_{ij}]$ is given by

$$\begin{aligned}E[X_{ij}] &= 1 \times \texttt{Prob}\ [X_{ij} = 1] + 0 \times \texttt{Prob}\ [X_{ij} = 0] \\ &= \frac{2}{j - i + 1}\end{aligned}$$

Finally, we calculate the expected number of comparisons, $E[X]$.

$$\begin{aligned}
E[X] &= \sum_{i=1}^{n-1} \sum_{j=i+1}^{n} E[X_{ij}] = \sum_{i=1}^{n-1} \sum_{j=i+1}^{n} \frac{2}{j-i+1} \\
&= \sum_{i=1}^{n-1} \sum_{k=2}^{n-i+1} \frac{2}{k} = 2\sum_{i=1}^{n-1} \sum_{k=2}^{n-i+1} \frac{1}{k} \quad \text{(rename } j-i+1 \text{ to } k\text{)} \\
&\leq 2\sum_{i=1}^{n-1} \sum_{k=1}^{n} \frac{1}{k} \\
&\leq 2\sum_{i=1}^{n-1} c \log n \quad (\because \text{ there exists a constant } c > 1 \text{ such that } \sum_{k=1}^{n} \frac{1}{k} \leq c \log n) \\
&\leq 2c(n-1)\log n
\end{aligned}$$

In other words, the expected number of comparisons ($E[X]$) is $O(n \log n)$. □