Lecture 8

1 Introduction to Randomized Algorithms

1.1 Random Quick Sort

```
Algorithm RandQS:

Input: A set $S = \{x_1, \dots, x_n\}$ of pairwise distinct real numbers.
Output: The elements of $S$ in sorted order.

  1. If $|S| \leq 1$, then return $S$.
  2. Choose some $p \in S$ as a pivot uniformly at random.
  3. By comparing every other element of $S$ to $p$, divide these elements into two sets
     as follows:
       (a) $S_1 := \{x \in S : x < p\}$.
       (b) $S_2 := \{x \in S : x > p\}$.
  4. Return: RandQS($S_1$)  $p$  RandQS($S_2$).
```

Theorem 1.1. *The expected running time of RandQS (over the choice of pivots) is $\Theta(n \ln n)$.*

In the proof of Theorem 1.1 we will make use of the following simple claim.

Claim 1.2. *Let $y_1, \dots, y_n$ be a sorting of S, that is, $\{y_1, \dots, y_n\} = \{x_1, \dots, x_n\}$ and $y_1 < y_2 < \dots < y_n$. Then, for every $1 \leq i < j \leq n$, the numbers y_i and y_j are compared at some point during the running of the algorithm if and only if y_i or y_j is the first element of $\{y_i, y_{i+1}, \dots, y_j\}$ to be chosen as pivot.*

Proof. Observe that y_i and y_j are compared if and only if, at some point where both are still in the same set, one of them is chosen as a pivot. Let $i \leq k \leq j$ be the unique integer such that y_k is the first element of $\{y_i, y_{i+1}, \dots, y_j\}$ to be chosen as pivot. Note that such a k must exist as y_i and y_j are in the same set in the beginning but not at the end. If $k \in \{i, j\}$, then y_i and y_j are still in the same subset which is to be sorted. Since one of them is currently the pivot and the other is not, they will be compared. On the other hand, if $i < k < j$, then both y_i and y_j will be compared to y_k. Since $y_i < y_k$ and $y_j > y_k$, they will be sent to separate sets and thus will never be compared. □

Proof of Theorem 1.1. Let X denote the total number of comparisons the algorithm performs and observe that it suffices to prove that $\mathbb{E}(X) = \Theta(n \ln n)$. We will in fact prove that $\mathbb{E}(X) = 2n \ln n + \Theta(n)$.

For every $1 \leq i < j \leq n$, let $X_{ij} = 1$ if y_i and y_j were compared at some point during the running of the algorithm, and $X_{ij} = 0$ otherwise. Observe that, for every $1 \leq i < j \leq n$, the numbers y_i and y_j are compared at most once. It follows that $X = \sum_{1 \leq i < j \leq n} X_{ij} = \sum_{i=1}^{n-1} \sum_{j=i+1}^{n} X_{ij}$ and thus

$$\mathbb{E}(X) = \mathbb{E}\left(\sum_{i=1}^{n-1} \sum_{j=i+1}^{n} X_{ij}\right) = \sum_{i=1}^{n-1} \sum_{j=i+1}^{n} \mathbb{E}(X_{ij})$$

holds by the linearity of expectation.

Now, it is an immediate corollary of Claim 1.2 that $\mathbb{E}(X_{ij}) = \mathbb{P}\left(X_{ij} = 1\right) = 2/(j - i + 1)$ holds for every $1 \leq i < j \leq n$. Therefore

$$\begin{aligned}\mathbb{E}(X) &= \sum_{i=1}^{n-1} \sum_{j=i+1}^{n} \mathbb{E}(X_{ij}) = \sum_{i=1}^{n-1} \sum_{j=i+1}^{n} \frac{2}{j - i + 1} = 2 \sum_{i=1}^{n-1} \sum_{k=2}^{n-i+1} \frac{1}{k} = 2 \sum_{k=2}^{n} \sum_{i=1}^{n-k+1} \frac{1}{k} \\ &= 2 \sum_{k=2}^{n} \frac{n + 1 - k}{k} = 2(n+1) \sum_{k=2}^{n} \frac{1}{k} - 2(n-1) = (2n+2) \sum_{k=1}^{n} \frac{1}{k} - 4n.\end{aligned}$$

Recalling that $\ln n \leq \sum_{k=1}^{n} \frac{1}{k} \leq \ln n + 1$, we conclude that $\mathbb{E}(X) = 2n \ln n + \Theta(n)$ as claimed. □

Next, we will prove an even stronger result.

Theorem 1.3. *For every set $S = \{x_1, \ldots, x_n\}$ of pairwise distinct real numbers, the running time of RandQS(S) is $\Theta(n \ln n)$ with probability at least $1 - 1/n$.*

Proof. With each execution of the algorithm we associate a binary tree as follows. The root of the tree corresponds to S. If some vertex of the tree corresponds to some set $S' \subseteq S$ and the pivot chosen for S' was p, then its left child is $S'_1 := \{x \in S' : x < p\}$ (provided that $S'_1 \neq \emptyset$) and its right child is $S'_2 := \{x \in S' : x > p\}$ (provided that $S'_2 \neq \emptyset$). Since partitioning S' into S'_1 and S'_2 requires linear time (i.e. $O(|S'|)$), the total work done at each level of the tree takes time $O(n)$. Therefore, in order to prove the theorem, it suffices to prove that, with probability at least $1 - 1/n$, the depth of the tree is $O(\ln n)$.

We will first prove that the probability that the distance of some leaf of the tree to the root is at least $24 \ln n$ is at most n^{-2}. Let P be a path from some leaf of the tree to the root. A vertex of P corresponding to some $S' \subseteq S$ is called *good* if $\max\{|S'_1|, |S'_2|\} \leq 2|S'|/3$; otherwise it is called *bad*.

Claim 1.4. *For sufficiently large n, at most $3 \ln n$ of the vertices of P are good.*

Proof. Let $v_1, v_2, \dots, v_t$ be the good vertices of P, appearing on P in this order (that is, if $i < j$, then v_i is closer to the root than v_j). For every $1 \leq i \leq t$, let s_i denote the size of the subset of S corresponding to v_i. For every $1 \leq i \leq t-1$, since v_i is a good vertex, it follows that $s_{i+1} \leq 2s_i/3$. Hence

$$1 \leq s_t \leq \left(\frac{2}{3}\right)^{t-1} n \implies t \leq \log_{3/2} n + 1 = \frac{\ln n}{\ln(3/2)} + 1 \leq 3 \ln n,$$

where the last inequality holds for sufficiently large n. □

Let P' consist of the first (say, starting from the root) $24 \ln n$ vertices of P if $|P| > 24 \ln n$, and let $P' = P$ otherwise. Let X be a random variable counting the number of bad vertices in P'. For every vertex $u \in P'$, let $X_u = 1$ if u is bad and $X_u = 0$ otherwise. Observe that $X = \sum_{u \in P'} X_u$, that the X_u's are independent, and that $\mathbb{P}(X_u = 1) \leq 2/3$. In particular, $\mathbb{E}(X) \leq 2|P'|/3 \leq 16 \ln n$. Using Chernoff's bound we obtain

$$\begin{aligned}\mathbb{P}(|P| \geq 24 \ln n) = \mathbb{P}(|P'| = 24 \ln n) &\leq \mathbb{P}(X \geq 21 \ln n) \leq \mathbb{P}(X \geq \mathbb{E}(X) + 5 \ln n) \\ &\leq e^{-2\frac{(5 \ln n)^2}{24 \ln n}} \leq 1/n^2,\end{aligned}$$

where the first inequality holds by Claim 1.4.

Finally, since clearly the tree has at most n leaves, a union bound argument shows that the probability that there exists a path from root to leaf of length at least $24 \ln n$ is at most

$$n \cdot \mathbb{P}(|P| \geq 24 \ln n) \leq n \cdot n^{-2} = 1/n$$

as claimed. □

2 Various types of randomized algorithms

Las Vegas Algorithms: A Las Vegas algorithm is a randomized algorithm whose output is always correct. However, the running time of a Las Vegas algorithm is a random variable. The standard definition of a Las Vegas algorithm includes the restriction that the expected running time should be finite. A classical example of a Las Vegas algorithm is Random Quick Sort.

Monte Carlo Algorithms: A Monte Carlo algorithm is a randomized algorithm whose output may be incorrect with a certain (typically small) probability. We further divide this class of algorithms (for decision problems) according to their allowed type of error.

One-sided error: Whenever the algorithm outputs true (respectively, false) this is always correct, but when it outputs false (respectively, true) there is some positive probability that this answer is incorrect. Both of the algorithms we presented in Lecture 7 are of this type.

Two-sided error: Regardless of what the algorithm outputs, there is a positive probability that this output is incorrect.