Randomized Algorithms

— Randomized QuickSort & k-Smallest Selection

Joseph Chuang-Chieh Lin

Department of Computer Science & Engineering,
National Taiwan Ocean University

Spring 2026

Credits for the resource

- The slides are based on the textbooks:
    - *Rajeev Motwani and Prabhakar Raghavan: Randomized Algorithms. Cambridge University Press. 1995.*

Outline

Outline

Illustration (a binary tree T demonstrating RandQS)

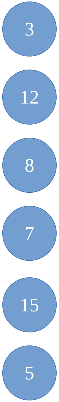


Illustration (a binary tree T demonstrating RandQS)

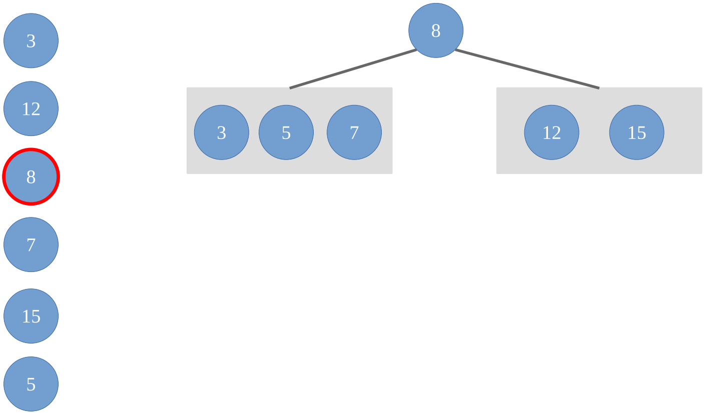


Illustration (a binary tree T demonstrating RandQS)

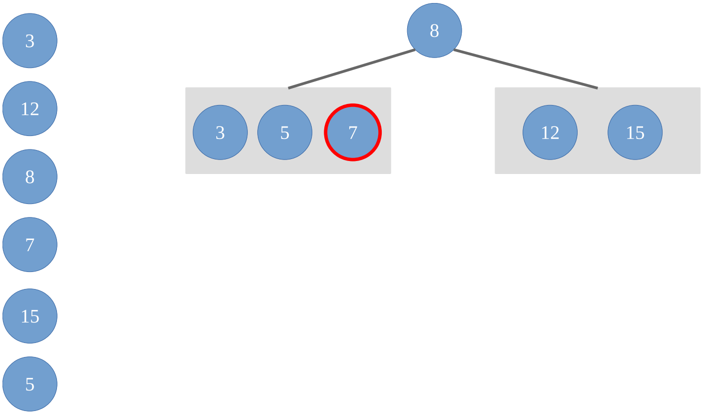


Illustration (a binary tree T demonstrating RandQS)

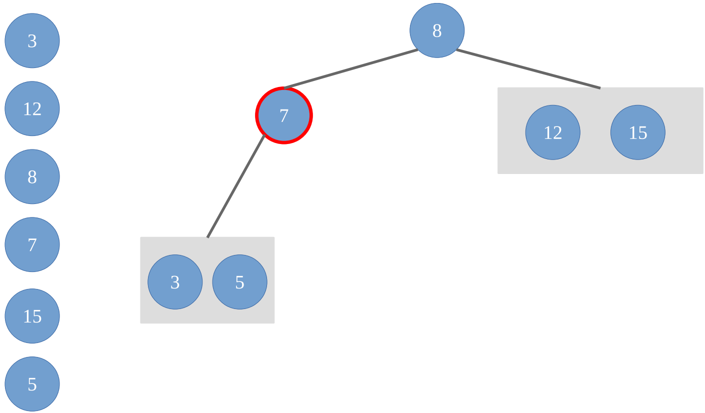


Illustration (a binary tree T demonstrating RandQS)

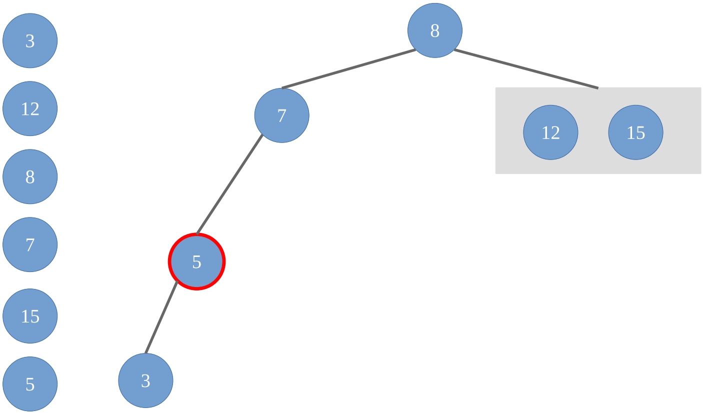
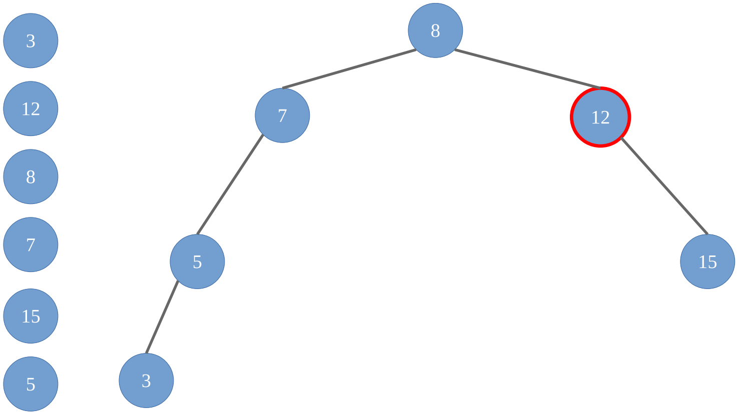


Illustration (a binary tree T demonstrating RandQS)



Algorithm RandQS

Input: A set of (distinct) numbers S
Output: The elements of S sorted in increasing order.

1. Choose an element $y \in S$ uniformly at random;
2. By comparing each element of S with y, compute
   - $S_1 := \{x \in S : x < y\}$;
   - $S_2 := \{x \in S : x > y\}$;
3. Recursively sort S_1 (i.e., run RandQS(S_1)) and S_2 (i.e., run RandQS(S_2)), and output the sorted version of S_1, followed by y, and then the sorted version of S_2.

Analysis (Expected Number of Comparisons)

- Comparisons are performed in Step 2.
- Let $S_{(i)}$ denote the element of rank i (i.e., the ith smallest in S).
- Define X_{ij}:
  - $X_{ij} = 1$ if $S_{(i)}$ and $S_{(j)}$ are compared in an execution.
  - $X_{ij} = 0$ otherwise.

Analysis (Expected Number of Comparisons)

- Comparisons are performed in Step 2.
- Let $S_{(i)}$ denote the element of rank i (i.e., the ith smallest in S).
- Define X_{ij}:
    - $X_{ij} = 1$ if $S_{(i)}$ and $S_{(j)}$ are compared in an execution.
    - $X_{ij} = 0$ otherwise.
- Thus, the total number of comparisons is $\sum_{i=1}^{n} \sum_{j>i} X_{ij}$,

Analysis (Expected Number of Comparisons)

- Comparisons are performed in Step 2.
- Let $S_{(i)}$ denote the element of rank i (i.e., the ith smallest in S).
- Define X_{ij}:
  - $X_{ij} = 1$ if $S_{(i)}$ and $S_{(j)}$ are compared in an execution.
  - $X_{ij} = 0$ otherwise.
- Thus, the total number of comparisons is $\sum_{i=1}^{n} \sum_{j>i} X_{ij}$, and its expected value is

$$\mathbb{E}\left[\sum_{i=1}^{n} \sum_{j>i} X_{ij}\right]$$

Analysis (Expected Number of Comparisons)

- Comparisons are performed in Step 2.
- Let $S_{(i)}$ denote the element of rank i (i.e., the ith smallest in S).
- Define X_{ij}:
  - $X_{ij} = 1$ if $S_{(i)}$ and $S_{(j)}$ are compared in an execution.
  - $X_{ij} = 0$ otherwise.
- Thus, the total number of comparisons is $\sum_{i=1}^{n} \sum_{j>i} X_{ij}$, and its expected value is

$$\mathbb{E}\left[\sum_{i=1}^{n} \sum_{j>i} X_{ij}\right] = \sum_{i=1}^{n} \sum_{j>i} \mathbb{E}[X_{ij}].$$

Analysis (contd.)

- Let p_{ij} denote the probability that $S_{(i)}$ and $S_{(j)}$ are compared in an execution.

Analysis (contd.)

- Let p_{ij} denote the probability that $S_{(i)}$ and $S_{(j)}$ are compared in an execution.
- Since X_{ij} is a Bernoulli random variable, we have

$$\mathbb{E}[X_{ij}] =$$

Analysis (contd.)

- Let p_{ij} denote the probability that $S_{(i)}$ and $S_{(j)}$ are compared in an execution.
- Since X_{ij} is a Bernoulli random variable, we have

$$\mathbb{E}[X_{ij}] = p_{ij} \times 1 + (1 - p_{ij}) \times 0$$

Analysis (contd.)

- Let p_{ij} denote the probability that $S_{(i)}$ and $S_{(j)}$ are compared in an execution.
- Since X_{ij} is a Bernoulli random variable, we have

$$\mathbb{E}[X_{ij}] = p_{ij} \times 1 + (1 - p_{ij}) \times 0 = p_{ij}.$$

Analysis (contd.)

- Let p_{ij} denote the probability that $S_{(i)}$ and $S_{(j)}$ are compared in an execution.
- Since X_{ij} is a Bernoulli random variable, we have

$$\mathbb{E}[X_{ij}] = p_{ij} \times 1 + (1 - p_{ij}) \times 0 = p_{ij}.$$

- Note: $S_{(i)}$ and $S_{(j)}$ are compared in an execution only when one of them is an ancestor of the other in the binary tree T.

Analysis (contd.)

$$\sum_{i=1}^{n} \sum_{j>i} p_{ij} = \sum_{i=1}^{n} \sum_{j>i} \frac{2}{j-i+1}$$

Analysis (contd.)

$$
\begin{aligned}
\sum_{i=1}^{n}\sum_{j>i} p_{ij} &= \sum_{i=1}^{n}\sum_{j>i}\frac{2}{j-i+1} \\
&= \sum_{i=1}^{n}\sum_{j=i+1}^{n}\frac{2}{j-i+1}
\end{aligned}
$$

Analysis (contd.)

$$\begin{aligned}\sum_{i=1}^{n}\sum_{j>i} p_{ij} &= \sum_{i=1}^{n}\sum_{j>i}\frac{2}{j-i+1} \\ &= \sum_{i=1}^{n}\sum_{j=i+1}^{n}\frac{2}{j-i+1} \\ &= \sum_{i=1}^{n}\sum_{k=2}^{n-i+1}\frac{2}{k}\end{aligned}$$

Analysis (contd.)

$$
\begin{aligned}
\sum_{i=1}^{n}\sum_{j>i} p_{ij} &= \sum_{i=1}^{n}\sum_{j>i} \frac{2}{j-i+1} \\
&= \sum_{i=1}^{n}\sum_{j=i+1}^{n} \frac{2}{j-i+1} \\
&= \sum_{i=1}^{n}\sum_{k=2}^{n-i+1} \frac{2}{k} \\
&\le 2\sum_{i=1}^{n}\sum_{k=1}^{n} \frac{1}{k} \qquad .
\end{aligned}
$$

Analysis (contd.)

$$
\begin{aligned}
\sum_{i=1}^{n}\sum_{j>i} p_{ij} &= \sum_{i=1}^{n}\sum_{j>i} \frac{2}{j-i+1} \\
&= \sum_{i=1}^{n}\sum_{j=i+1}^{n} \frac{2}{j-i+1} \\
&= \sum_{i=1}^{n}\sum_{k=2}^{n-i+1} \frac{2}{k} \\
&\leq 2\sum_{i=1}^{n}\sum_{k=1}^{n} \frac{1}{k} .
\end{aligned}
$$

- Note that $H_n = \sum_{k=1}^{n} 1/k$

Analysis (contd.)

$$
\begin{aligned}
\sum_{i=1}^{n}\sum_{j>i} p_{ij} &= \sum_{i=1}^{n}\sum_{j>i}\frac{2}{j-i+1} \\
&= \sum_{i=1}^{n}\sum_{j=i+1}^{n}\frac{2}{j-i+1} \\
&= \sum_{i=1}^{n}\sum_{k=2}^{n-i+1}\frac{2}{k} \\
&\leq 2\sum_{i=1}^{n}\sum_{k=1}^{n}\frac{1}{k} = O(n\log n).
\end{aligned}
$$

- Note that $H_n = \sum_{k=1}^{n} 1/k \approx \Theta(\ln n)$.

Exercise (3%)

Using $O(n)$ Median-of-Medians Algorithm

- **Remark:** The Median-of-Medians algorithm (reference here) by Blum et al. can compute a median of an array of n numbers in a list in $O(n)$ time deterministically.
- Please prove that Algorithm MedianQS (next page) can sort an array of n numbers in $O(n \log n)$ time deterministically.

Algorithm MedianQS

Input: A set of (distinct) numbers S
Output: The elements of S sorted in increasing order.

1. Compute the median y of S using the Median-of-Medians algorithm;
2. By comparing each element of S with y, compute
   - $S_1 := \{x \in S : x < y\}$;
   - $S_2 := \{x \in S : x > y\}$;
3. Recursively sort S_1 (i.e., run MedianQS(S_1)) and S_2 (i.e., run MedianQS(S_2)), and output the sorted version of S_1, followed by y, and then the sorted version of S_2.

Outline

1. Randomized QuickSort
2. Randomized k-Smallest Selection

Algorithm Rand-(k)-Select

Input: A set of n (distinct) numbers S
Output: The k-th smallest element of S.

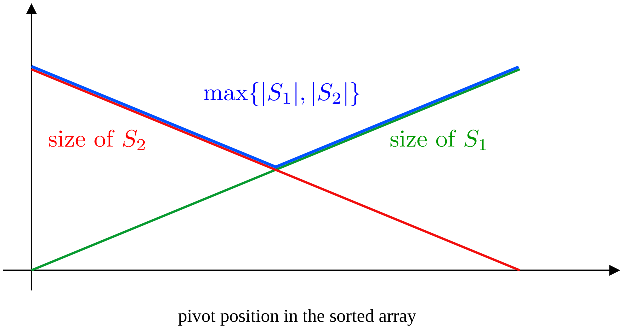
1. Choose an element $y \in S$ uniformly at random;
2. By comparing each element of S with y, compute
   - $S_1 := \{x \in S : x < y\}$;
   - $S_2 := \{x \in S : x > y\}$;
3. If $|S_1| = k - 1$ then return y
4. Else
   - if $|S_1| \geq k$, then recursively run Rand-(k)-Select(S_1).
   - else, recursively run Rand-($k - |S_1| - 1$)-Select(S_2).

Time Complexity Analysis (1/3)

- Let $X := \max\{|S_1|, |S_2|\}/n$.

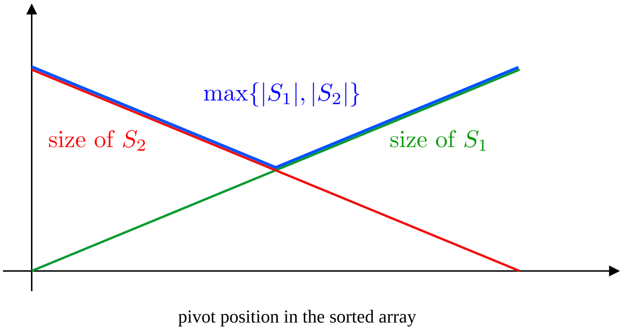
Time Complexity Analysis (1/3)

- Let $X := \max\{|S_1|, |S_2|\}/n$.



Time Complexity Analysis (1/3)

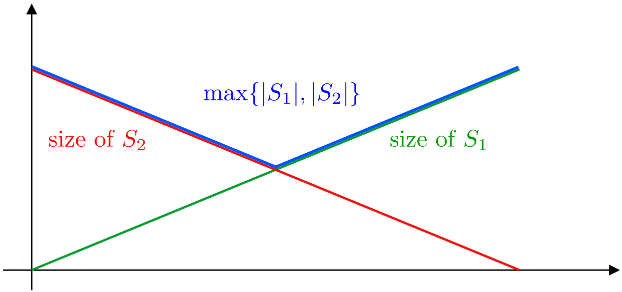
- Let $X := \max\{|S_1|, |S_2|\}/n$.



- What's $\mathbb{E}[X]$?

Time Complexity Analysis (1/3)

- Let $X := \max\{|S_1|, |S_2|\}/n$.



- What's $\mathbb{E}[X]$?
- Prove that $\mathbb{E}[X] \leq \frac{3}{4}$ (Exercise (1%)).

Time Complexity Analysis (2/3)

Note: The recursion only runs in exactly one of S_1 and S_2.

- Let Y_i be the size of the subset of S that the ith recursion proceeds with.

Time Complexity Analysis (2/3)

Note: The recursion only runs in exactly one of S_1 and S_2.

- Let Y_i be the size of the subset of S that the ith recursion proceeds with.
  - $\mathbb{E}[Y_i] =$

Time Complexity Analysis (2/3)

Note: The recursion only runs in exactly one of S_1 and S_2.

- Let Y_i be the size of the subset of S that the ith recursion proceeds with.
  - $\mathbb{E}[Y_i] = \mathbb{E}\left[n \prod_{j=1}^{i} X_j\right] =$

Time Complexity Analysis (2/3)

Note: The recursion only runs in exactly one of S_1 and S_2.

- Let Y_i be the size of the subset of S that the ith recursion proceeds with.
  - $\mathbb{E}[Y_i] = \mathbb{E}\left[n\prod_{j=1}^{i} X_j\right] = \; = n\prod_{j=1}^{i}\mathbb{E}[X_j]$

Time Complexity Analysis (2/3)

Note: The recursion only runs in exactly one of S_1 and S_2.

- Let Y_i be the size of the subset of S that the ith recursion proceeds with.
  - $\mathbb{E}[Y_i] = \mathbb{E}\left[n \prod_{j=1}^{i} X_j\right] = \; = n \prod_{j=1}^{i} \mathbb{E}[X_j] \leq n \left(\frac{3}{4}\right)^i.$

Time Complexity Analysis (3/3)

- Since the "partitioning" step takes $c_1(|S|) + c_2$ for some constants $c_1, c_2 \in \mathbb{R}$, the expected running time of the algorithm is at most

$$\begin{aligned}
\mathbb{E}[\text{Rand-}(k)\text{-Select}(S)] &\leq \sum_{i=0}^{n} \left(c_1 n \left(\frac{3}{4} \right)^i + c_2 \right) \\
&\leq c_1 n \left(\sum_{i=0}^{n} \left(\frac{3}{4} \right)^i \right) + c_2 n \\
&\leq 4c_1 n + c_2 n \\
&= O(n).
\end{aligned}$$

Discussions