

Agenda

RANDOMIZED ALGORITHMS – INTRODUCTION

- MOTIVATING EXAMPLE: QUICK SORT**
- ALGORITHM AND ANALYSIS**

Algorithm QS

- Input: A set S of numbers
- Output: Elements of S sorted in increasing order
- Steps
 1. Choose an element p from S
 2. Partition S into $S_<$ and $S_>$ s.t.
 1. all elements less than p are in $S_<$ and all elements greater than p are in $S_>$
 3. Sort $S_<$
 4. Sort $S_>$

Algorithm QS - Analysis

- The pivot chosen in Step 1 decides how
 - Step 2 partitions S into sublists
 - i.e. sizes of the sublists are dependent on the input “order” if the pivot is chosen deterministically!
 - What is the best case behavior?
 - Can you ensure that happens in every (recursive) step?
 - What is the worst case behavior?
- Compare QS with the process of “constructing a binary search tree”
 - Compare the *height of the tree* with the expected *depth of recursion in QS*

Algorithm RandQS

- Input: A set S of numbers
- Output: Elements of S sorted in increasing order
- Steps
 1. Choose an element p uniformly at random from S
 2. Partition S into $S_<$ and $S_>$ s.t.
 1. all elements less than p are in $S_<$ and all elements greater than p are in $S_>$
 3. Sort $S_<$
 4. Sort $S_>$

Algorithm RandQS

- What does it achieve (apart from sorting)?
 - Step 1 ensures Step 2 partitions S into sublists uniformly randomly
 - i.e. sizes of the sublists are not dependent on the input “order”
 - This is the best you can hope for! (Why?)
- Compare RandQS with the process of “constructing a binary search tree”
 - Compare the *expected height of the tree* with the expected *depth of recursion in RandQS*

Algorithm RandQS - Analysis

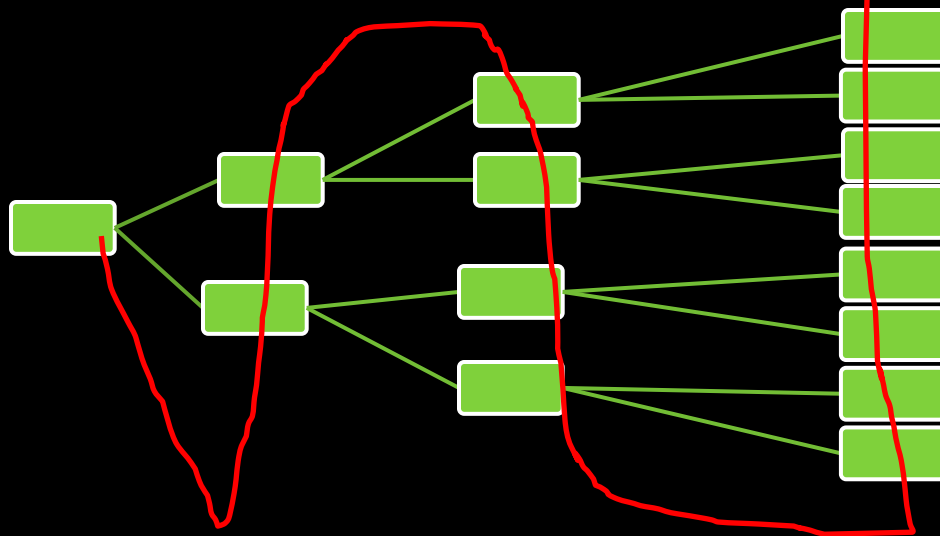
- What is the expected number of comparisons?
 - **Observations:**
 - All comparisons happen in Step 2 (partitioning)
 - No pair of elements are compared more than once.
 - **Notation:**
 - Let $S_{(i)}$ denote the element of rank i in S
 - Define random variable X_{ij} to denote the number of comparisons between $S_{(i)}$ and $S_{(j)}$
 - i.e. X_{ij} is 1 if $S_{(i)}$ and $S_{(j)}$ are compared; 0 otherwise
 - **So, the total number of comparisons is**
 - $\sum_{i=1}^n \sum_{j>i}^n X_{ij}$

Algorithm RandQS - Analysis

- Thus the expected number of comparisons is
 - $E \left[\sum_{i=1}^n \sum_{j>i}^n X_{ij} \right]$
 - By linearity of expectations
 - $E \left[\sum_{i=1}^n \sum_{j>i}^n X_{ij} \right] = \sum_{i=1}^n \sum_{j>i}^n E \left[X_{ij} \right]$
 - By definition of expected values
 - $E \left[X_{ij} \right] = p_{ij} * 1 + (1 - p_{ij}) * 0 = p_{ij}$
 - where p_{ij} is the probability that $S_{(i)}$ and $S_{(j)}$ are compared in an execution.
- Thus the expected number of comparisons is
 - $\sum_{i=1}^n \sum_{j>i}^n p_{ij}$

Algorithm RandQS - Analysis

- $S_{(i)}$ and $S_{(j)}$ are compared if and only if:
 - one of them is an ancestor of the other in the BST
- Let π be the permutation obtained by visiting the nodes of T
 - in increasing order of level numbers
 - and left-to-right within each level



Claim:

One of $S_{(i)}$ and $S_{(j)}$ is an ancestor of the other if and only if:

one of them occurs earlier in π than any element $S_{(m)}$ such that $i < m < j$

Algorithm RandQS - Analysis

Claim:

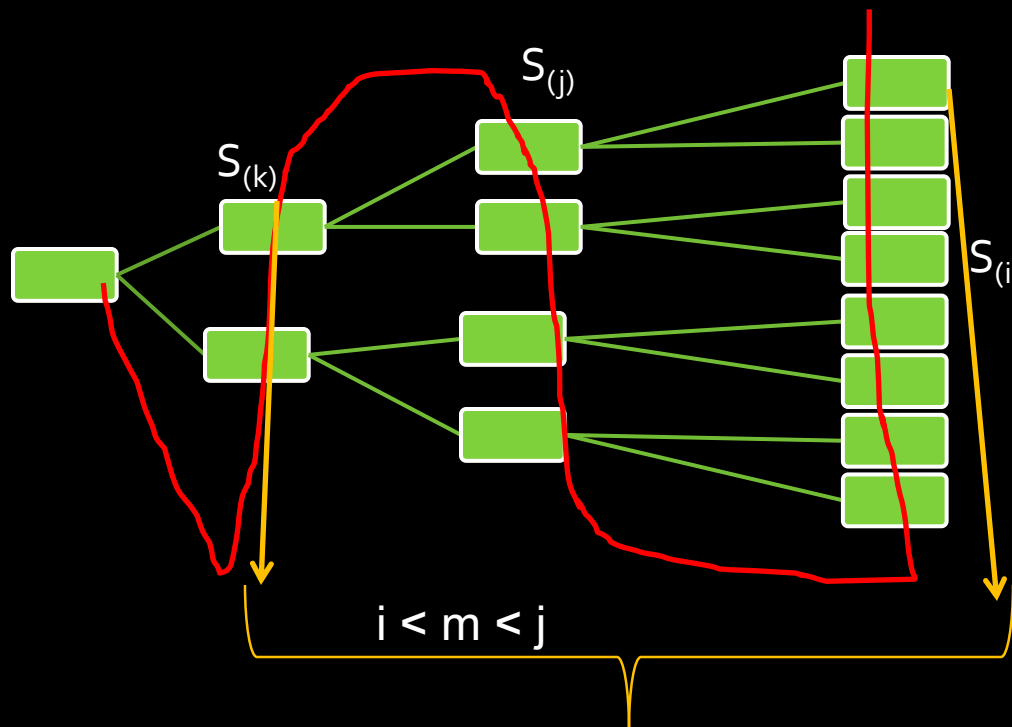
One of $S_{(i)}$ and $S_{(j)}$ is an ancestor of the other only if:

one of them occurs earlier in π than any element $S_{(m)}$ such that $i < m < j$

Proof:

Let $S_{(k)}$ be the earliest in π among elements of rank between i and j .

- Assume $k \neq i$ and $k \neq j$
- Then $S_{(i)}$ and $S_{(j)}$ will belong to the left and right sub-trees - respectively - of $S_{(k)}$ i.e neither is an ancestor of the other.

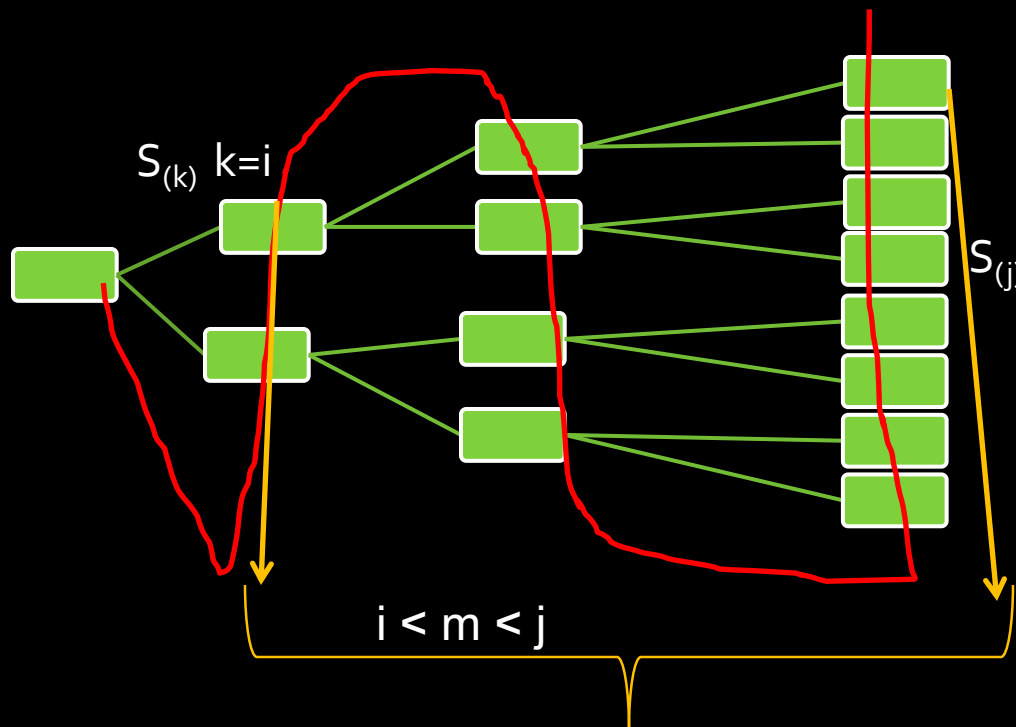


Algorithm RandQS - Analysis

Claim:

One of $S_{(i)}$ and $S_{(j)}$ is an ancestor of the other if:

one of them occurs earlier in π than any element $S_{(m)}$ such that $i < m < j$



Proof:

Let $S_{(k)}$ be the earliest in π among elements of rank between i and j .

Assume $k=i$ or $k=j$

Then this (say $k=i$) must be the ancestor of the other (j).

Algorithm RandQS - Analysis

- All the elements $S_{(i)}, S_{(i+1)}, \dots, S_{(j)}$ are equally likely to appear the earliest in π
 - Probability that one of the two elements (i.e. i or j) occurs the earliest in π is

$$2 / (j-i+1)$$

- Thus, $p_{ij} = 2 / (j-i+1)$

- Then

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

- $= \sum_{i=1}^n \sum_{k>1}^{n-i+1} (2/k)$ // let $k=j-i+1$

- $\leq 2 * \sum_{i=1}^n \sum_{k=1}^n (1/k)$

- $= 2nH_n = O(n*\log n)$

- where H_n is the n -th Harmonic number i.e. $\sum_{i=1}^n 1/k$

- and H_n is $O(\log n)$