Data Structures and Algorithms
Probability in Computer Science

CS 225
Brad Solomon

October 23, 2022

Department of Computer Science

Learning Objectives

Finish disjoint set analysis (one final proof)

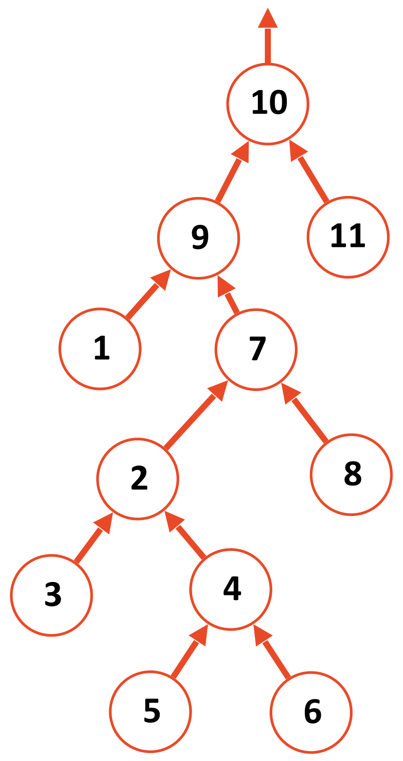
Formalize the concept of randomized algorithms

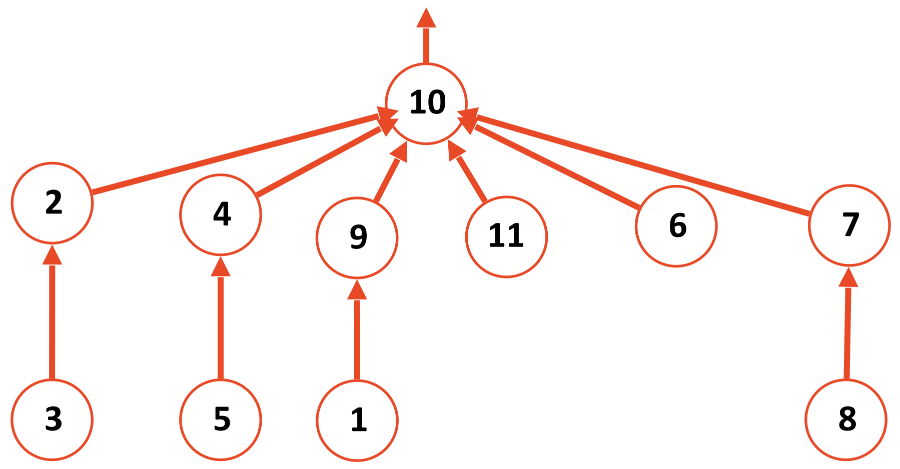
Review fundamentals of probability in computing

Distinguish the three main types of ‘random’ in computer science

Disjoint Sets w/ Path Compression

How do we observe how the efficiency of a set changes due to PC?

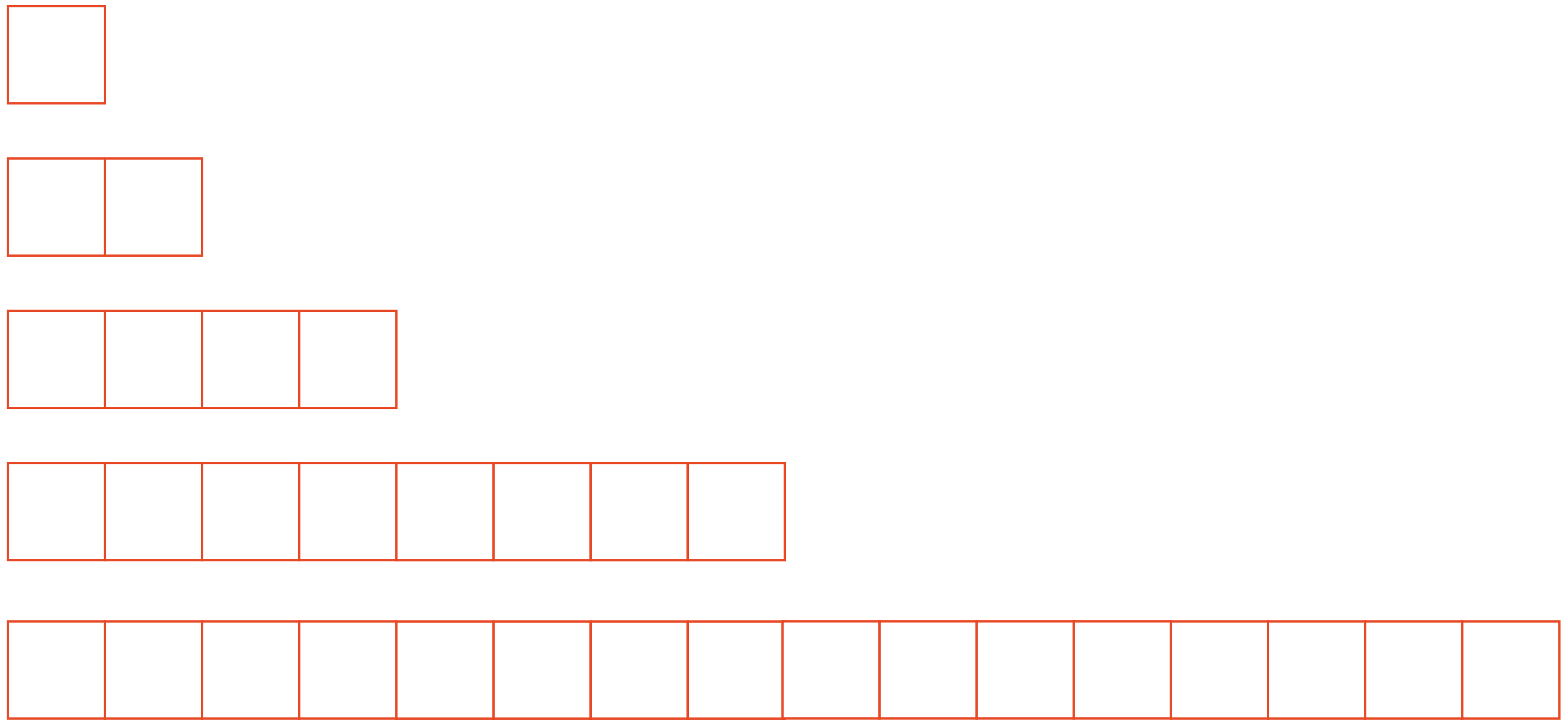




Amortized Time Review

We have **n items**. We make **n insert()** calls.

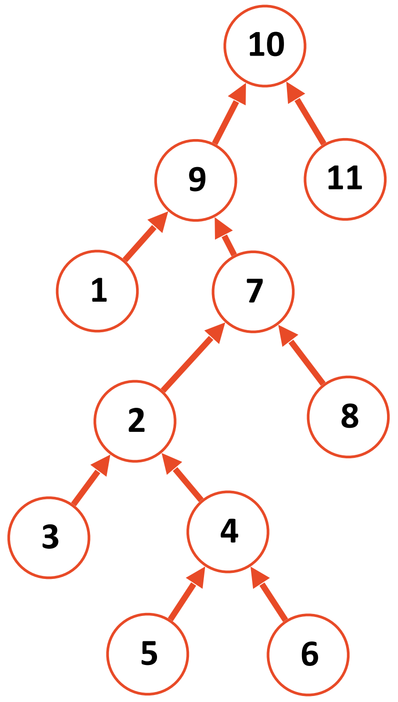
We are interested in the **worst case work** possible **over n calls**.



Amortized Time (Rank w/ Path Compression)

We have **n items** in an Uptree. We make **m find()** calls.

We are interested in the **worst case work** possible **over m calls**.



Key Properties of UpTree by rank w/ PC

The parent of a node is always higher rank than the node.

The min(nodes) in a set with a root of rank r has $\geq 2^r$ nodes.

For any integer r, there are at most $\frac{n}{2^r}$ nodes of rank r.

Amortized Time (Rank w/ Path Compression)

Put every non-root node in a bucket by rank!

Structure buckets to store ranks $[r,\ 2^r - 1]$

Ranks	Bucket
0	0
1	1
2 - 3	2
4 - 15	3
16 – 65535	4
65536 – 2^{65536}-1	5

Iterated Logarithm Function (log^*n)

log^*n is piecewise defined as

0 if $n \leq 1$

otherwise

$1 + log^*(\log n)$

Amortized Time (Rank w/ Path Compression)

Let $|B_r|$ be the size of the bucket with min rank r.

What is $max(|B_r|)$?

Ranks	Bucket
0	0
1	1
2 - 3	2
4 - 15	3
16 – 65535	4
65536 – 2^{65536}-1	5

Amortized Time (Rank w/ Path Compression)

The work of **find(x)** are the steps taken on the path from a node x to the root (or immediate child of the root) of the UpTree containing x

We can split this into two cases:

Case 1: We take a step from one bucket to another bucket.

Case 2: We take a step from one item to another inside the same bucket.

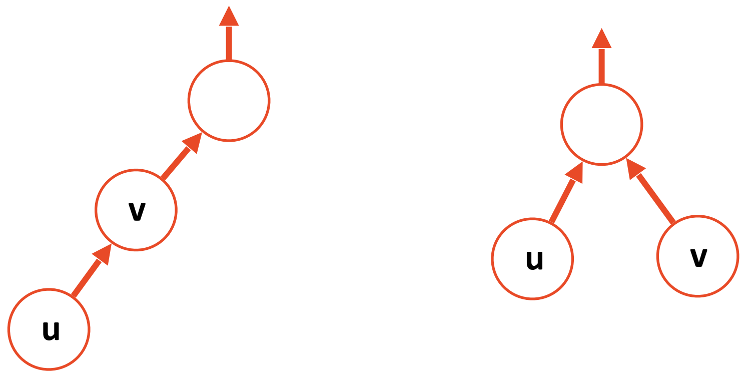
Amortized Time (Rank w/ Path Compression)

Case 2: We take a step from one item to another *inside* the same bucket.

Let's call this the step from **u** to **v**.

Every time we do this, we do path compression:

We set parent(u) a little closer to root



How many total times can I do this for each **u** in $|B_r|$?

How many nodes are in $|B_r|$?

Final Result

For **n** calls to makeSets() [**n items**] and **m** find() calls the total work is:

Randomized Algorithms

A **randomized algorithm** is one which uses a source of randomness somewhere in its implementation.

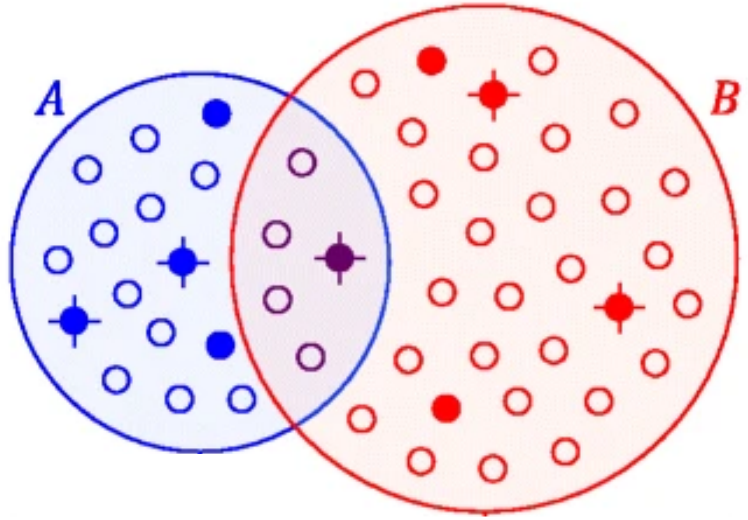
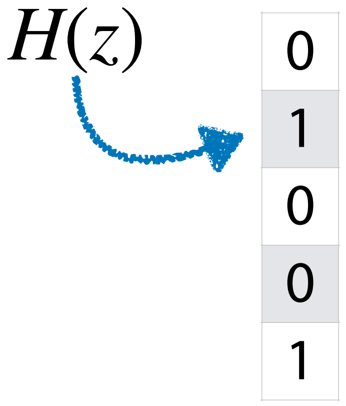
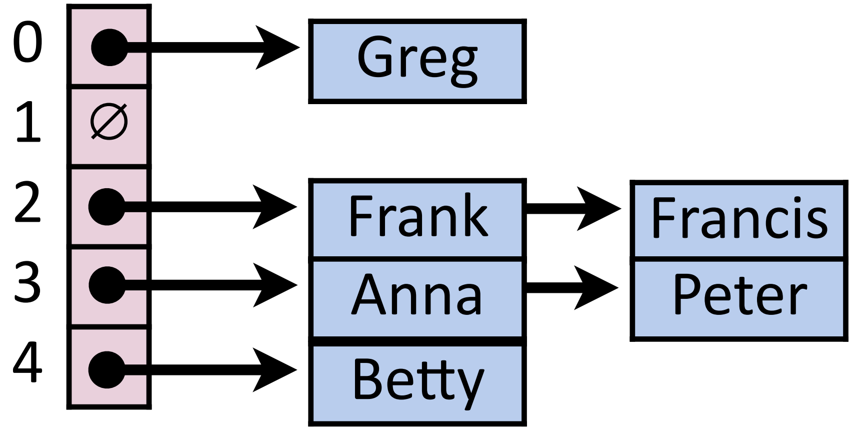


Figure from Ondov et al 2016





$H(x)$	0	2	1	0	0	4	0	2	0	6
$H(y)$	1	0	2	3	1	0	3	4	0	1
$H(z)$	2	1	0	2	0	1	0	0	7	2

A faulty list

Imagine you have a list ADT implementation ***except***…

Every time you called **insert**, it would fail 50% of the time.

Quick Primes with Fermat's Primality Test

If p is prime and a is not divisible by p, then $a^{p-1} \equiv 1 \pmod{p}$

But… ***sometimes*** if n is composite and $a^{n-1} \equiv 1 \pmod{n}$

Fundamentals of Probability

Imagine you roll a pair of six-sided dice.

The **sample space** Ω is the set of all possible outcomes.

An **event** $E \subseteq \Omega$ is any subset.

Fundamentals of Probability

Imagine you roll a pair of six-sided dice. What is the expected value?

A **random variable** is a function from events to numeric values.

The **expectation** of a (discrete) random variable is:

$$E[X] = \sum_{x \in \Omega} Pr\{X = x\} \cdot x$$

Fundamentals of Probability

Imagine you roll a pair of six-sided dice. What is the expected value?

Linearity of Expectation: For any two random variables X and Y,

$$E[X + Y] = E[X] + E[Y]$$

Fundamentals of Probability

Imagine you roll a pair of six-sided dice. What is the expected value?

Linearity of Expectation: For any two random variables X and Y,

$$E[X+Y] = E[X] + E[Y]$$

$$= \sum_x \sum_y Pr\{X=x, Y=y\}(x+y)$$

$$= \sum_x x \sum_y Pr\{X=x, Y=y\} + \sum_y y \sum_x Pr\{X=x, Y=y\}$$

$$= \sum_x x \cdot Pr\{X=x\} + \sum_y y \cdot Pr\{Y=y\}$$

Fundamentals of Probability

Imagine you roll a pair of six-sided dice. What is the expected value?

Linearity of Expectation: For any two random variables X and Y,

$$E[X+Y] = E[X] + E[Y]$$

Randomization in Algorithms

1. Assume input data is random to estimate average-case performance

2. Use randomness inside algorithm to estimate expected running time

3. Use randomness inside algorithm to approximate solution in fixed time

Average-Case Analysis: BST

Let $S(n)$ be the average **total internal path length** over all BSTs that can be constructed by uniform random insertion of n objects

Claim: $S(n)$ is $O(n\ log\ n)$

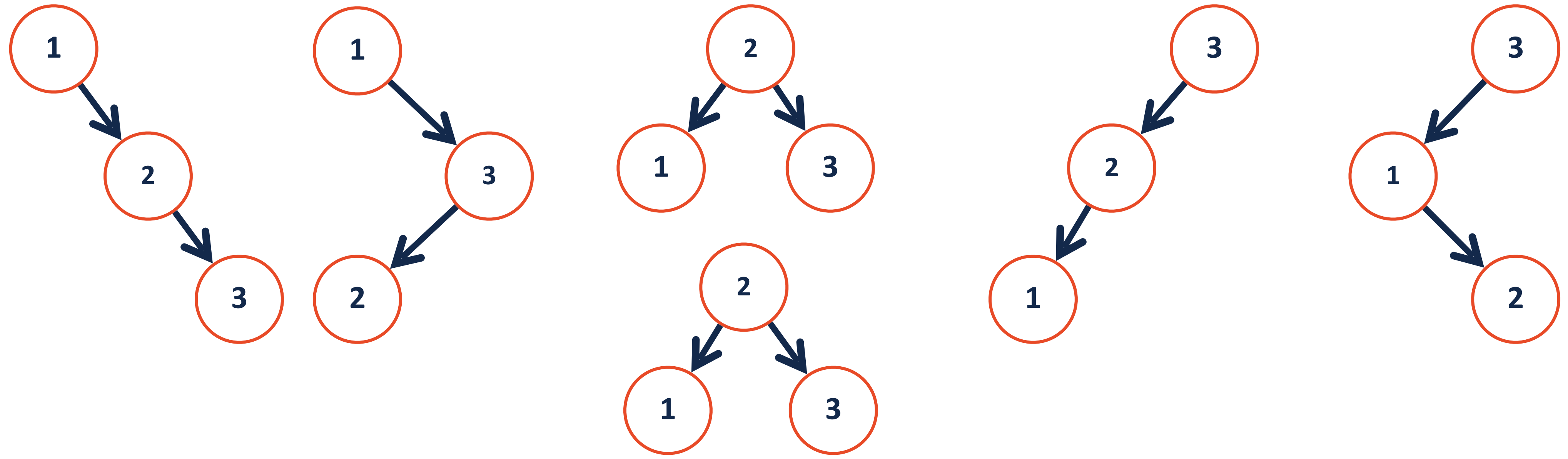
N=0:

N=1:

Average-Case Analysis: BST

Let $S(n)$ be the average **total internal path length** over all BSTs that can be constructed by uniform random insertion of n objects

N=3:



Average-Case Analysis: BST

Let $S(n)$ be the average **total internal path length** over all BSTs that can be constructed by uniform random insertion of n objects

Let $0 \leq i \leq n - 1$ be the number of nodes in the left subtree.

Then for a fixed i, $S(n) = (n - 1) + S(i) + S(n - i - 1)$

Average-Case Analysis: BST

Let $S(n)$ be the **average** total internal path length **over all BSTs** that can be constructed by uniform random insertion of n objects

$$S(n) = (n-1) + \frac{1}{n}\sum_{i=0}^{n-1} S(i) + S(n-i-1)$$

Average-Case Analysis: BST

$$S(n) = (n - 1) + \frac{2}{n} \sum_{i=1}^{n-1} S(i)$$

$$S(n) = (n - 1) + \frac{2}{n} \sum_{i=1}^{n-1} (ci \ ln \ i)$$

$$S(n) \leq (n - 1) + \frac{2}{n} \int_{1}^{n} (cx \ ln \ x) dx$$

$$S(n) \leq (n - 1) + \frac{2}{n} \left(\frac{cn^2}{2} \ ln \ n - \frac{cn^2}{4} + \frac{c}{4} \right) \approx cn \ ln \ n$$

Average-Case Analysis: BST

Let $S(n)$ be the average **total internal path length** over all BSTs that can be constructed by uniform random insertion of n objects

Since $S(n)$ is $O(n \ log \ n)$, if we assume we are randomly choosing a node to insert, find, or delete* then each operation takes:

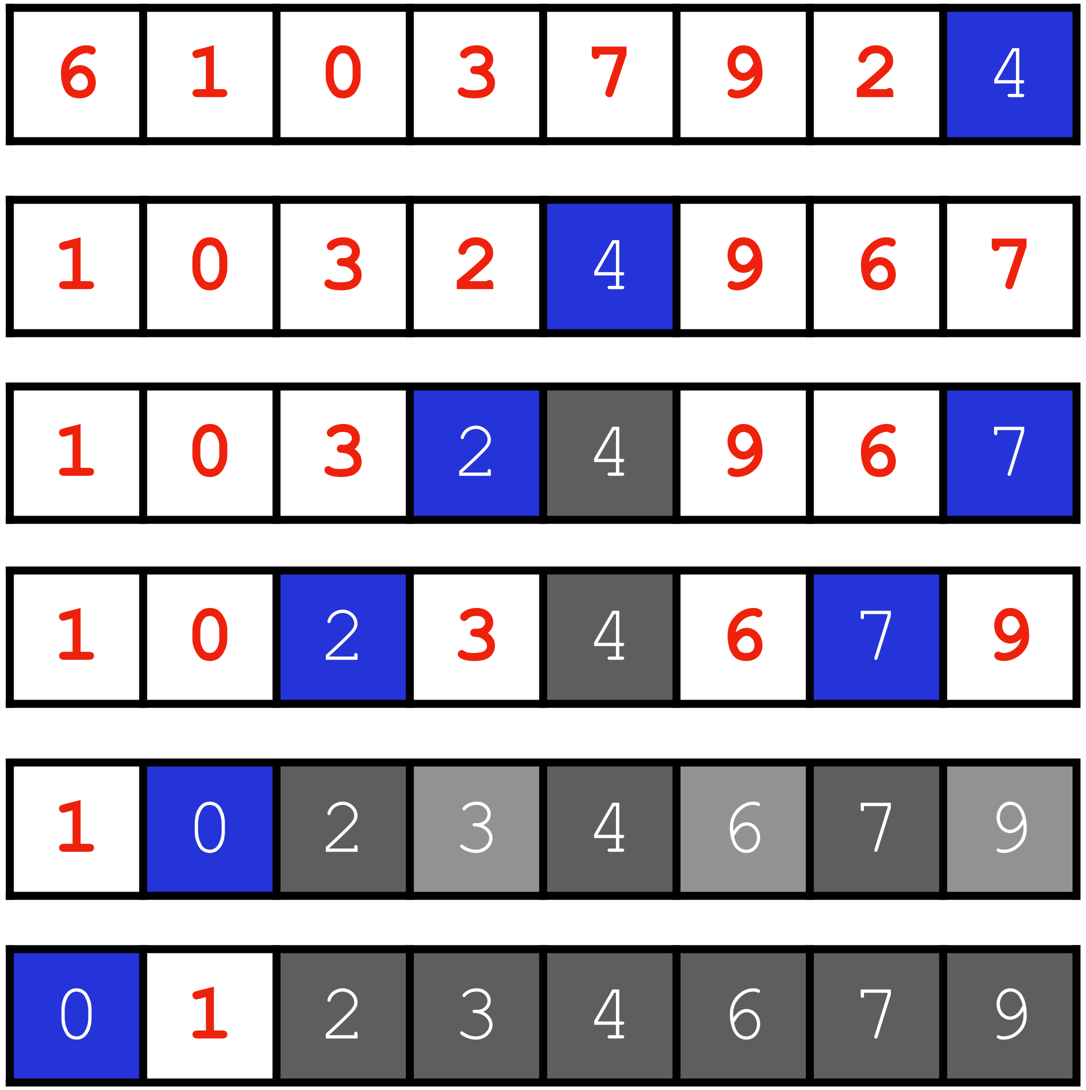
Average-Case Analysis: BST

Summary: All operations are on average $O(log\ n)$

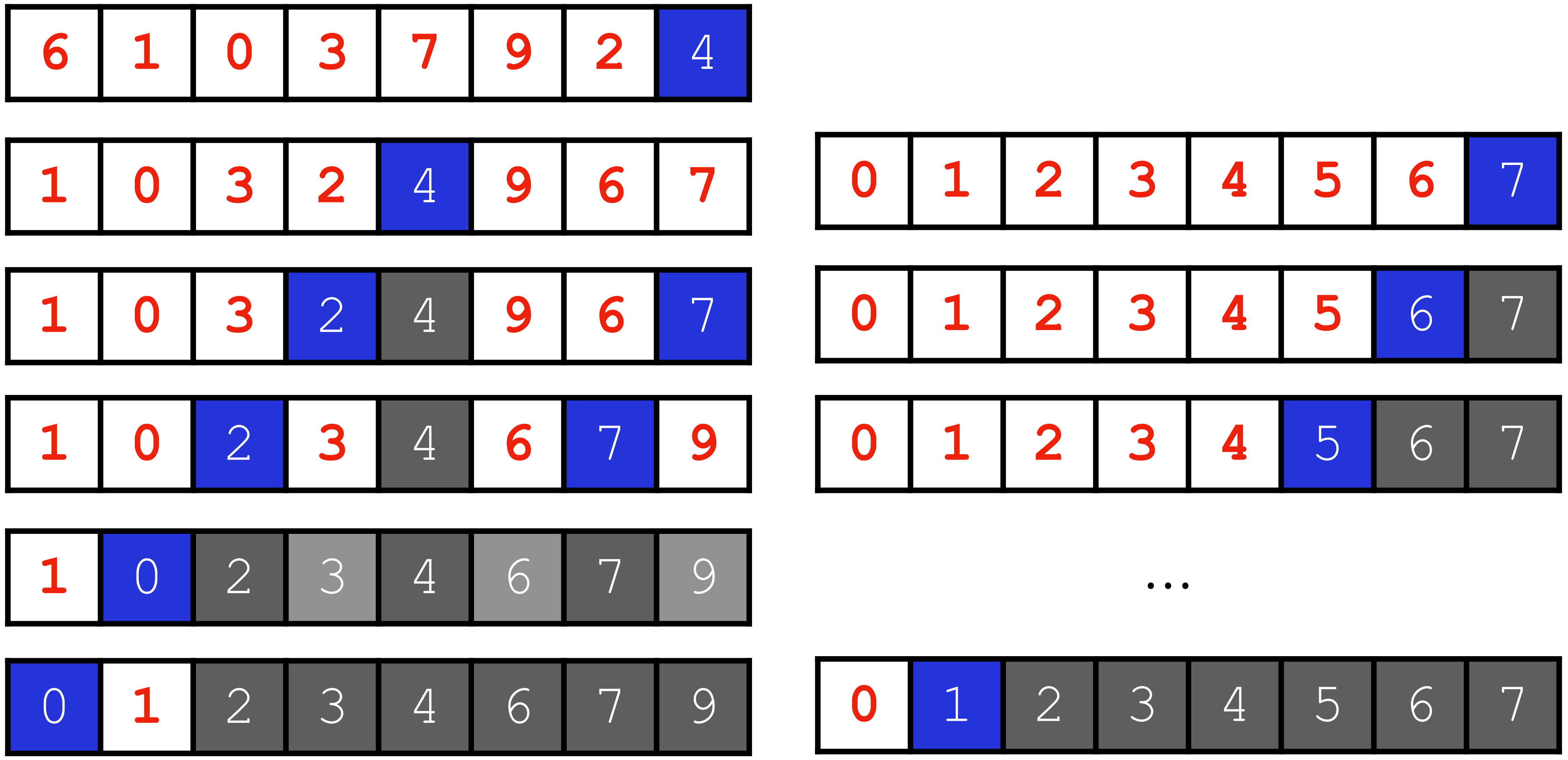
Randomness:

Assumptions:

Expectation Analysis: Randomized Quicksort



Expectation Analysis: Randomized Quicksort



Expectation Analysis: Randomized Quicksort

In **randomized quicksort**, the selection of the pivot is random.

Claim: The expected time is $O(n \ log \ n)$ ***for any input!***

Let X be the total comparisons and X_{ij} be an **indicator variable**:

$$X_{ij} = \begin{cases} 1 & \text{if } i\text{th object compared to } j\text{th} \\ 0 & \text{if } i\text{th object not compared to } j\text{th} \end{cases}$$

Then…