# CS 173: Discrete Structures

Eric Shaffer
Office Hour: Wed. 12-1,  2215 SC
shaffer1@illinois.edu

# Announcements

- Mid-term 2 in class Wednesday April 8
  - Discussion sections next week will be review sessions
- Exam will cover through what we get through today
- Trees: Section 10.1
- Structural Induction: Section 4.3

# Average Case analysis

Imagine sequentially searching a list of **n** numbers for some number **x**

Suppose x is in the list and is equally likely to be in any position 1 to n

Suppose checking for x at a given position in the list takes Θ(1) time.

On average, how much time does it take to find x?

#ops

$$\left.\begin{array}{ll} (x = a_1) & 1 \\ a_2 & 2 \\ \vdots & \\ a_n & n \end{array}\right\}$$

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

Avg case

$$= \frac{\frac{n(n+1)}{2}}{n} = \boxed{\frac{n+1}{2}} = \boxed{\Theta(n)}$$

# Average Case analysis

Imagine sequentially searching a list of **n** numbers for some number **x**

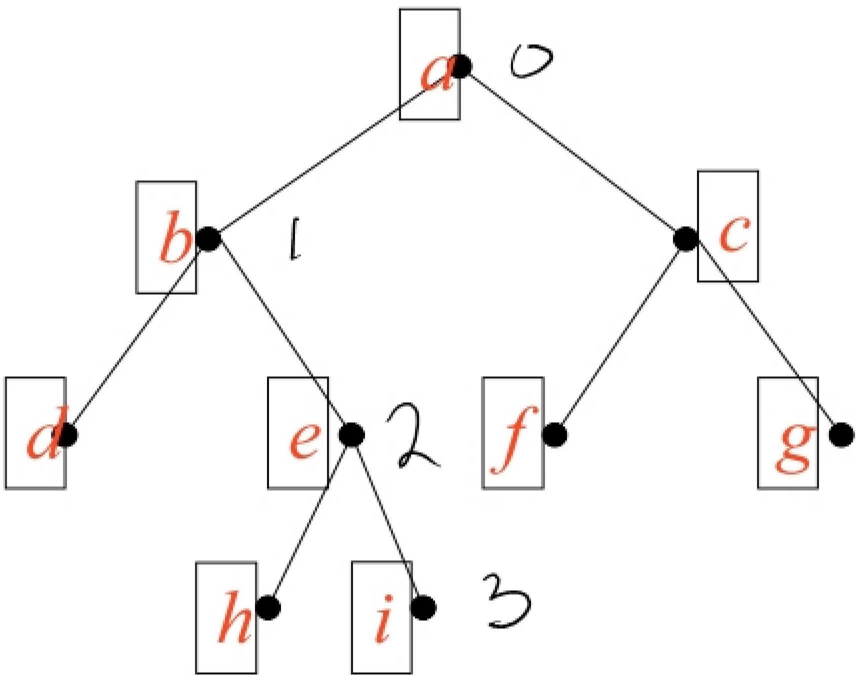
Imagine that the probability of x not being in the list is 0.90

Imagine that the probability of x being in the list is 0.10

How much time does searching take on the average?

$$0.10\left(\frac{n+1}{2}\right) + 0.9(n) = \boxed{\Theta(n)}$$

# Rooted tree terminology




The root is: a

The children of b are: d, e

The parent of g is: c

The sibling of g is: f

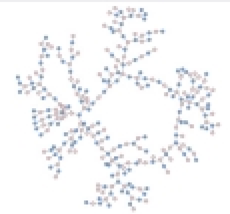
The level of b is: 1

The height is: 3

The ancestors of d are: b, a

Leaves are: d h i f g

Internal vertices are: all others



# Trees

- **A Tree** is a mathematical object
 (just like number or a set or…some other mathematical object)

vertices

- What two sets are present in a tree: $V = \{v_1, v_2, \dots, v_n\}$

edges $E = \{(v_i, v_j), \dots\}$ $n-1$

- What 2 properties are necessary for those sets to be a tree:

① acyclic

② connected

# One more fact about binary trees

- In a binary tree, each vertex has 0, 1, 2 children
- In a full binary tree each vertex has 0, 2 children
- Suppose we have i internal vertices in a fbt T
- Then there are n= $2i+1$ vertices
- How many leaves are there?

$$n - i = 2i + 1 - i = \boxed{i + 1}$$

$$\boxed{L = \frac{n-1}{2} + 1}$$

# Structural Induction

Structural Induction is an inductive proof technique

It is used to prove results about recursively defined sets

Suppose a set $S$ is defined recursively.

An SI proof about a set $S$ has two parts:

1. <u>Base case</u>
   Shows that a property holds for all elements in the base caseof the recursive definition of $S$

2. <u>Recursive step</u>
   Assumes that a property holds for the elements $x_i$ in $S$

   Shows that property holds for a new element produced from the $x_i$ using the recursive definition of the set

# Structural Induction

Suppose S is defined by this recursive definition:

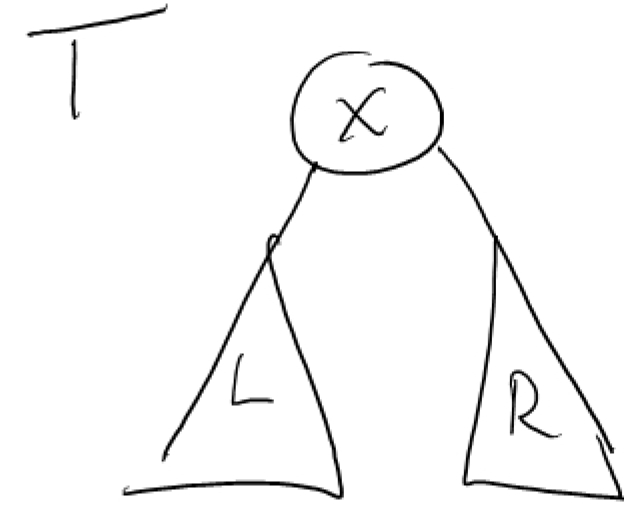
1. The number 2 is in S
2. If x is in S then 2x is in S

Prove that any x in S is a power of 2:

1. 2 is a power of 2 since $2=2^1$
2. Assume that some number y is in S and $y = 2^i$

   By the recursive definition of S, $x=2y$ is in S
   Since $x=2(2^i)=2^{i+1}$ it follows that x is a power of 2

# Structural Induction




We can define full binary trees recursively as well:

1. A single vertex is a full binary tree
2. If L and R are full binary trees, a new full binary tree can be built by making new root vertex x and making the roots of L and R children of x

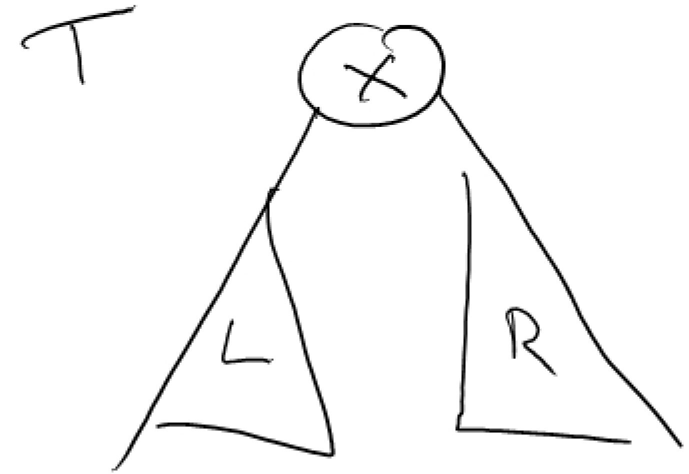
$n_T$ = # vertices in T = $n_L + n_R + 1$

$n_L$ = # vertice in L

$n_R$ = R

$h_T$ = height of T = $\max(h_L, h_R) + 1$

# Binary Trees




Claim: A full binary tree of height h has $\underbrace{n \le 2^{h+1} - 1}$ total vertices

Proof:

Base case is a fbt with a single vertex, h=0 and $1 \le 2^1 - 1$

Recursive step:

Let L be an fbt with $n_L$ vertices and R be an fbt with $n_R$ vertices.

Let L have height $h_L$ and R have height $h_R$

We make a new fbt T with a root having the roots of L and R as children

$h_T = H + 1$

Let $\boxed{H=\max(h_L, h_R)}$ so that T has height $h_T = H+1$

The number of vertices in T will be $\boxed{n = n_L+n_R+1}$

It follows from the inductive hypothesis that:

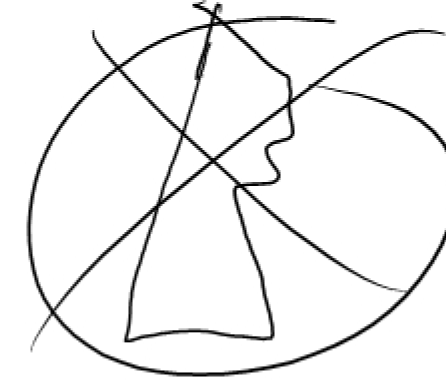
$\boxed{n_L \le 2^{H+1} \text{ and } n_R \le 2^{H+1}}$ so $\underbrace{n = n_L+n_R+1} \le \underbrace{2^{H+1} - 1 + 2^{H+1} - 1 + 1} = 2(2^{H+1}) - 1$

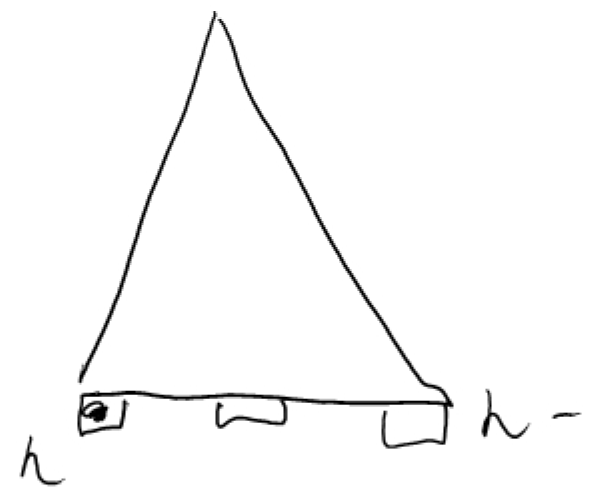
$= 2(2^{h_T}) - 1$

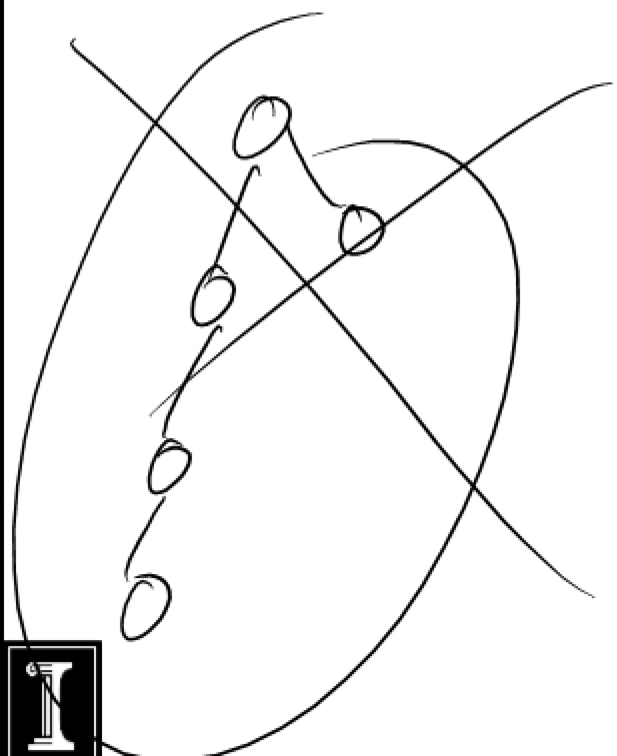
$= \boxed{2^{h_T+1} - 1}$

# Balanced Binary Trees

- In a binary tree of height h
  - What does it mean to be balanced?
  - All leaves at the height h or h-1





# Binary Trees

Theorem: In a full, balanced binary tree T with *L* leaves and height *h*, $L>2^{h-1}$

**Proof**

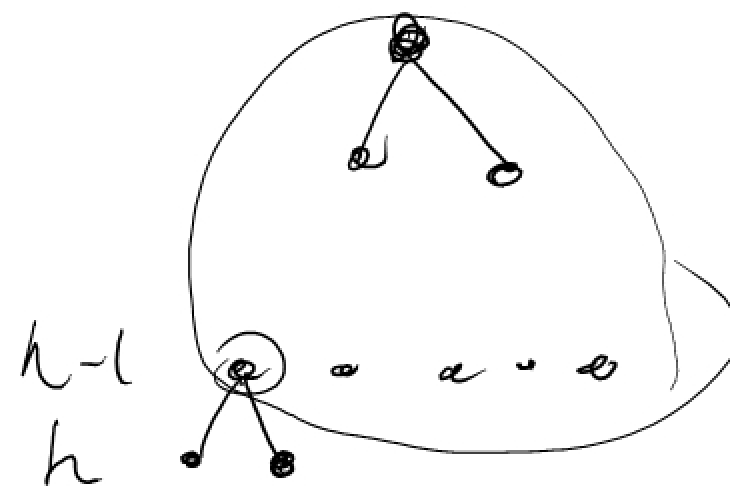
The tree is balanced -> all leaves at level h or h-1

At least two leaves at level h.

$$L = 2^{h-1} - 1 + 2 = \underbrace{2^{h-1} + 1}_{\text{min \# leaves}}$$

$$\boxed{L > 2^{h-1}}$$

A binary tree w/height h has at most $2^h$ leaves



# Binary Trees

Theorem: In a full, balanced binary tree T with $L$ leaves and height $h$, it is true that $h = \text{ceil}(\log_2 L)$

Proof:

We know $\boxed{2^{h-1} < L \le 2^h}$

$$\lg 2^{h-1} < \lg L \le \lg 2^h$$

$$h-1 < \lg L \le h \quad \text{so } h = \lceil \lg L \rceil$$

# Binary Trees

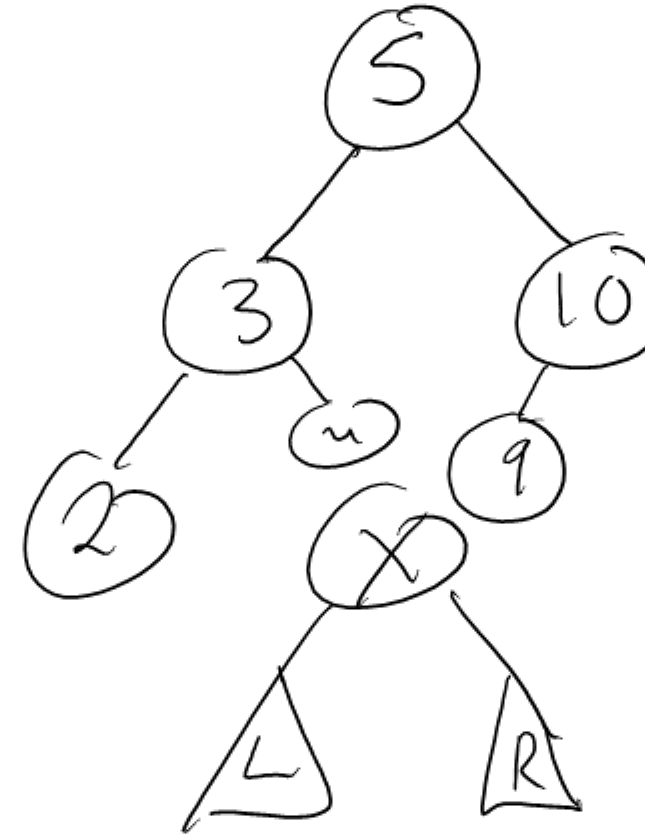
In a full n vertex binary tree with height h, what is the number of leaves in terms of n?

$$\boxed{L = \frac{n-1}{2} + 1}$$

Bound the height in terms of n:

$$h = \left\lceil \lg\left(\frac{n-1}{2} + 1\right) \right\rceil$$

$$\boxed{h = O(\lg n)}$$




$L < x$

$x < R$