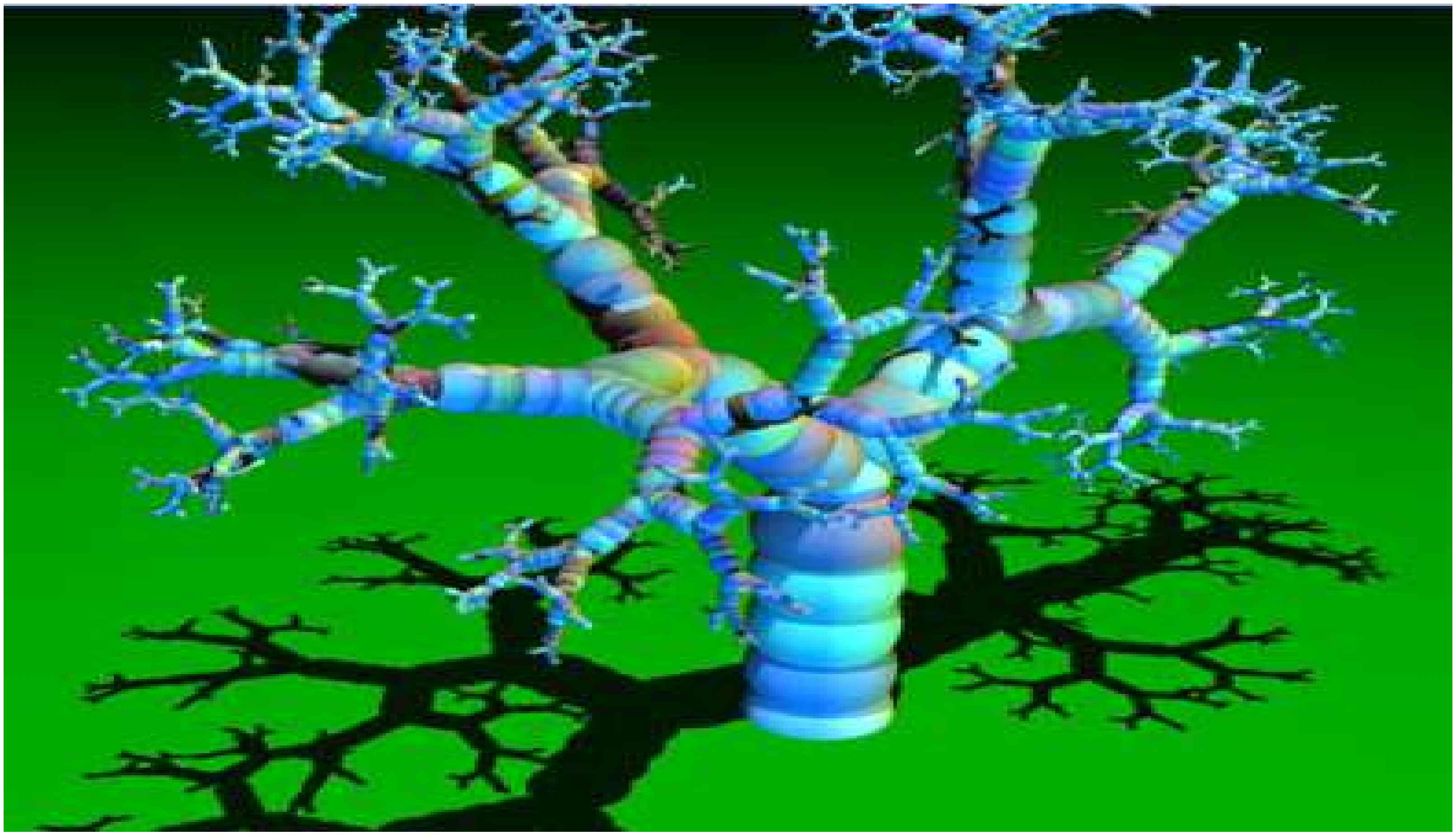
Binary Search

A Lecture in CE Freshman Seminar Series:
Ten Puzzling Problems in Computer Engineering

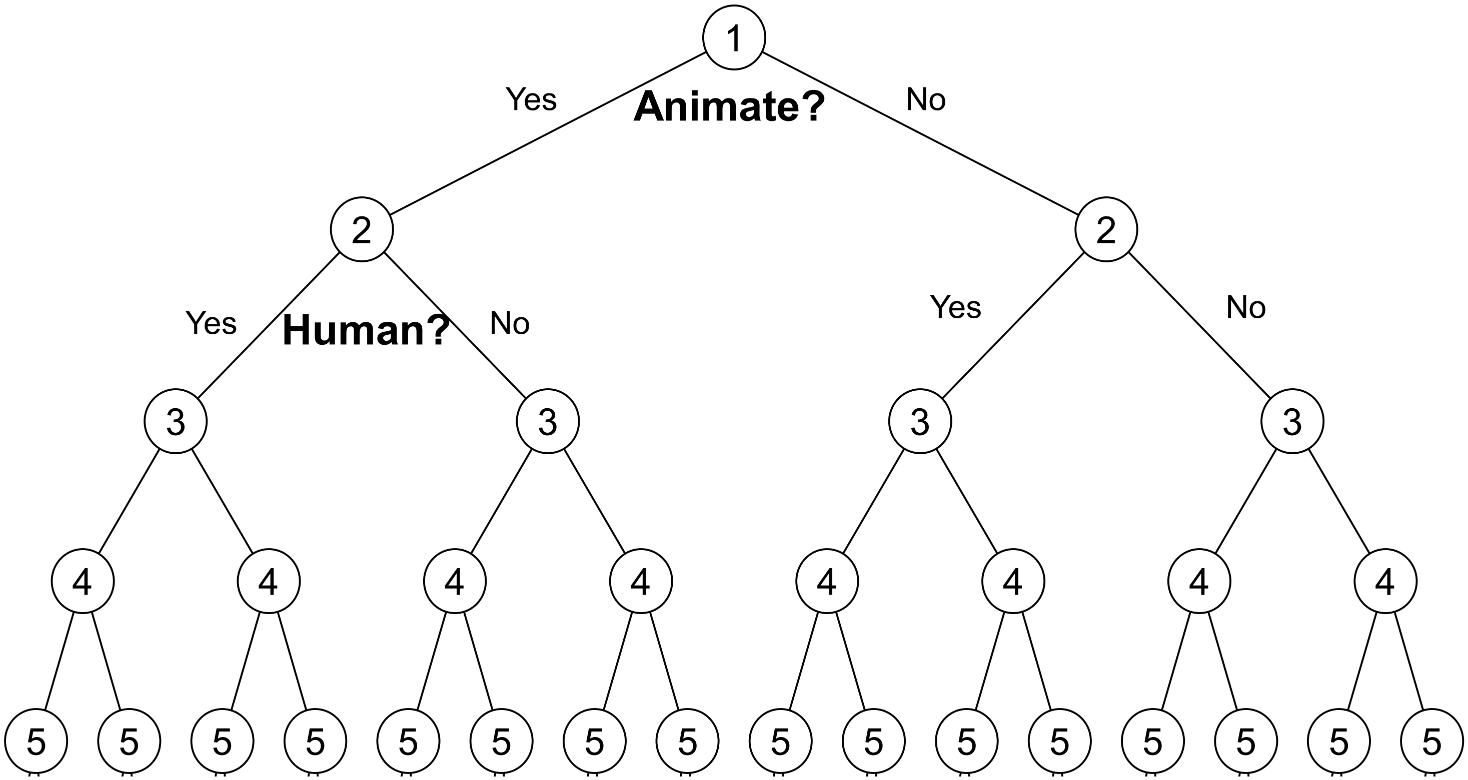


About This Presentation

This presentation belongs to the lecture series entitled "Ten Puzzling Problems in Computer Engineering," devised for a ten-week, one-unit, freshman seminar course by Behrooz Parhami, Professor of Computer Engineering at University of California, Santa Barbara. The material can be used freely in teaching and other educational settings. Unauthorized uses, including any use for financial gain, are prohibited. © Behrooz Parhami

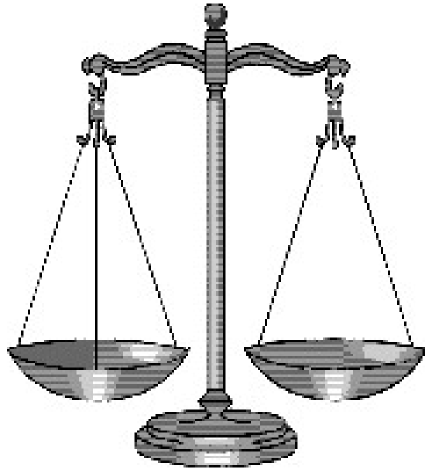
Edition	Released	Revised	Revised	Revised	Revised
First	May 2007	May 2008	May 2009	May 2010	Apr. 2011
		May 2012	May 2015	Apr. 2016	Apr. 2020

Game of 20 Questions as Binary Search



With perfect questioning, one of 2^{20} possible answers can be found with 20 questions

Weighing with a Balance



A large container is known to hold 24 oz of nails.
The hardware store has a balance, but no weights.
Can you measure out 9 oz of nails for a customer?

Divide all nails into two equal piles: 12 oz 12 oz

Divide one pile into two equal piles: 12 oz 6 oz 6 oz

. . . and again: 12 oz (6 oz 3 oz) 3 oz

A chemist has a balance and fixed weights of 1, 2, 4, and 8 grams. Show that she can weigh any amount of material from 1 to 15 grams by placing the weights on one side and the material on the other.

3 = 2 + 1; 5 = 4 + 1; 6 = 4 + 2; 7 = 4 + 2 + 1; 9 = 8 + 1; 10 = 8 + 2; 11 = 8 + 2 + 1

What is the best set of 4 fixed weights in the sense of maximizing the range of measurable weights in increments of 1 gram? (e.g., 1, 4, 7, 12)

Weights of 1, 3, 9, and 27 grams allow us to measure up to 40 grams

Find the Lighter Counterfeit Coin

We have three coins. Two are normal coins; one is a counterfeit coin that weighs less. Identify the counterfeit coin with one weighing on a balance.

Compare coins 1 & 2.
If they weigh the same, coin 3 is counterfeit; otherwise the lighter of the two is counterfeit.

We have nine coins; eight normal coins and a counterfeit coin that weighs less. Identify the counterfeit with 2 weighings.

Generalize: How many weighing with a balance are needed to find a light counterfeit coin among *n* coins?

We need *w* weighing with a balance to find a light counterfeit coin among 3^w coins.
So, the number of required weighings with *n* coins is $w = \lceil \log_3 n \rceil$.

How should we change the procedures above if the counterfeit coin is known to be heavier than normal ones instead of lighter?

12 Coins with 1 Counterfeit: Lighter or Heavier

We have 12 coins. Eleven are normal coins; one is a counterfeit coin that weighs less or more than a normal coin. Identify the counterfeit coin and its relative weight with a minimum number of weighings on a balance.

Hint: First do it for 3 coins, one of which is a counterfeit, using only two weighing,

If A = B, then C contains the counterfeit coin.

Weigh 3 coins from C against 3 good coins. If equal, the lone remaining coin in C is counterfeit and one more weighing is enough to tell if it's lighter or heavier than normal.

If the three C coins are lighter, then . . .

If the three C coins are heavier, then . . .

If A < B or A > B . . .

Another Solution to the 12-Coin Puzzle

We have 12 coins. Eleven are normal coins; one is a counterfeit coin

1 4 6 10	against	5 7 9 12
2 5 4 11	against	6 8 7 10
3 6 5 12	against	4 9 8 11

Each weighing has three possible outcomes:
L -- Left heavier
R -- Right heavier
B -- Balance

L L L	np	B L L		R L L	
L L B	7–	B L B		R L B	
L L R		B L R		R L R	
L B L		B B L		R B L	
L B B		B B B	np	R B B	
L B R		B B R		R B R	
L R L		B R L		R R L	
L R B		B R B		R R B	7+
L R R		B R R		R R R	np

Example: L L B -- Counterfeit coin is among 1, 2, 7, 10 → 7 is lighter

Q1: Complete the table above to show the counterfeit coin in all 27 cases.

Searching in Unsorted and Sorted Lists

How would you find the person or business having the phone number 765-4321 in a standard phone directory?

Because a standard phone directory is sorted by names, rather than by numbers, we have no choice but to scan the entire directory.

On average, half of the entries are examined before either the number is found or the end of the directory is reached. This is an $O(n)$ algorithm.

How would you find the meaning of “scissile” (pronounced ˈsis-əl) in a standard English dictionary?

We do not have to search the entire dictionary. We examine a page in the area where we think “s” words are listed. Then we know whether to search before or after that page. We repeat this process, each time narrowing the search region.

On average, ≈10 pages are examined in a 1000-page dictionary before finding the word or discovering that it is not a valid word. This is an $O(\log n)$ algorithm.

By the way, “scissile” means “easily cut or split”

Searching in an Alphabetized List

Is "tomato paste" an ingredient?

Possible range: [0, 20]
Middle of the range = (0 + 20)/2 = 10

tomato paste > olive or vegetable oil

Possible range: [11, 20]
Middle of the range = (11 + 20)/2 = 15

tomato paste > sliced pitted ripe olives

Possible range: [16, 20]
Middle of the range = (16 + 20)/2 = 18

tomato paste > thinly sliced pepperoni

Possible range: [19, 20]
Middle of the range = (19 + 20)/2 = 19

Tomato paste is indeed an ingredient!

Thompson Family

Prep Date	Serve Date	Meal	All-American Pizza
9/15/2003	9/15/2003	Dinner	Planned

Scaled Amount		Ingredient
2 cups	0	all-purpose flour
4 cups		apple cider
8 slices	2	bacon
1 cup		Big Chief brown sugar
1 cup	4	catsup
1/2 teaspoon		cinnamon
4 cups	6	cranberry juice cocktail
1 teaspoon		crushed red pepper (optional)
2 teaspoon	8	dry mustard
1 package		Fleischmann's® Rapid Rise Yeast
	10	olive or vegetable oil
4 1-pound cans		pork and beans
1 1/2 teaspoons	12	salt
2 cups		shredded Mozzarella cheese
2 2-ounce jars	14	sliced pimientos
1/		sliced pitted ripe olives
1 teaspoon	16	spaghetti sauce seasoning
1/4 cup		Sue Bee Honey
4 o	18	thinly sliced pepperoni or salami
1 6-		tomato paste
1 cup	20	water

A Guessing Game

Interactive search game via Khan Academy

The computer chooses a number

You try to find that number by a sequence of guesses, the fewer, the better

After each guess, the computer provides one of three possible responses:

"Correct!", "Too high!", or "Too low!"

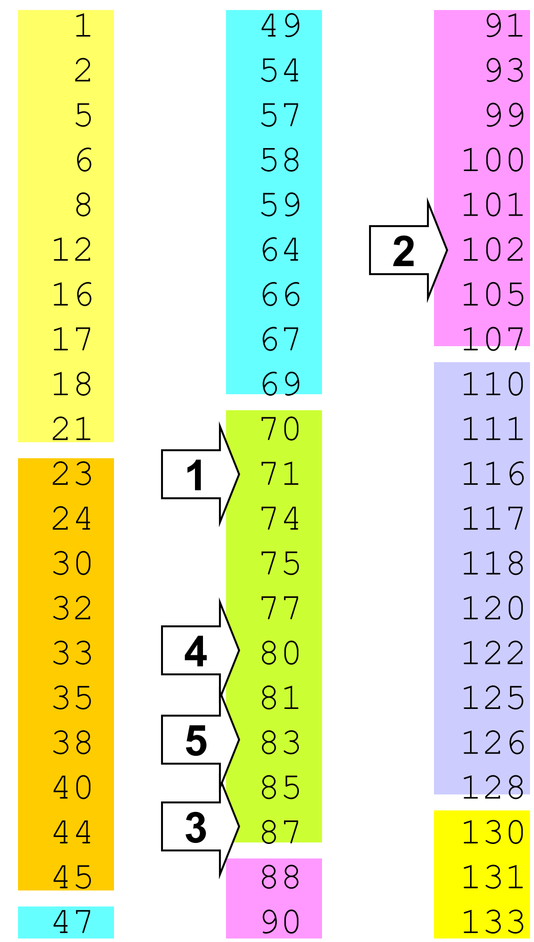
https://www.khanacademy.org/computing/computer-science/algorithms/intro-to-algorithms/a/a-guessing-game

Q2: Play the guessing game above for a number in [1, 300] three times. Record and report the number of questions you asked in the 3 rounds and attach a screenshot of the "winning" screen in one of the rounds.

The Binary Search Algorithm

Is the number 85 in the 63-entry list to the right?

First	Last	Middle	Entry	Outcome
1	63	32	71	>
33	63	48	102	<
33	47	40	87	<
33	39	36	80	>
37	39	38	83	>
37	37	37	85	=



Six probes are needed with a 63-entry list, in the worst case

More generally, a $(2^n - 1)$-entry list requires *n* probes in the worst case

Interpolation Search

Is the number 85 in the 63-entry list to the right?

When looking for an entry *x* in a list, probe it at size(*x* – min)/(max – min)

First probe is at 63(85 – 1)/(133 – 1) ≈ 40

Second probe is at 40(85 – 1)/(87 – 1) ≈ 39

First	Last	Probe	Entry	Outcome
1	63	40	87	<
1	40	39	85	=

Dictionary lookup:
When looking up a word in the dictionary, we instinctively use interpolation search

1		49	91
2		54	93
5		57	99
6		58	100
8		59	101
12		64	102
16		66	105
17		67	107
18		69	110
21		70	111
23		71	116
24		74	117
30		75	118
32		77	120
33		80	122
35		81	125
38		83	126
40	2 →	85	128
44	1 →	87	130
45		88	131
47		90	133

Searching in Dynamic Lists

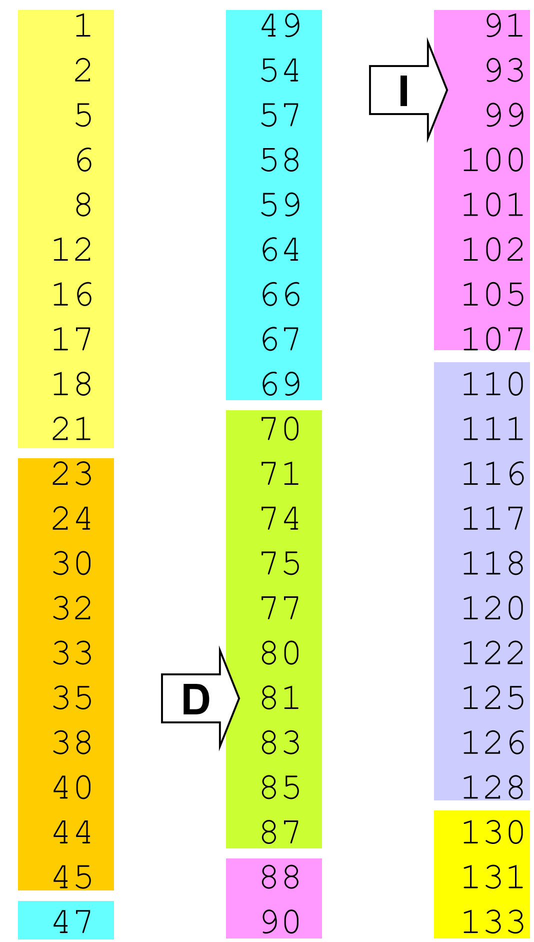
A dynamic list has entries inserted or deleted

If we use a binary search algorithm on a dynamic list, its sorted order must be maintained

Example: Delete 81 from the list
1. Search to find the entry 81
2. Move all entries beyond 81 one notch up

Example: Insert 95 into the list
1. Search for 95, to see where it should go
2. Move all entries beyond there a notch down
3. Put 95 in the vacated location

Addition/deletion takes $O(n)$ steps on average. So, if the number of additions/deletions is comparable to the number of searches, sorting the list does not buy us anything



Examples of Dynamic Lists

Students currently enrolled at UCSB:
This list is dynamic, but does not change often

Customers of a wireless phone company currently having active connections:
This list may change 1000s of times per minute

Even “static” lists may change on occasion …

UCSB graduates, class of 2000:
This list is nearly static, but may change to include missing persons or to make corrections

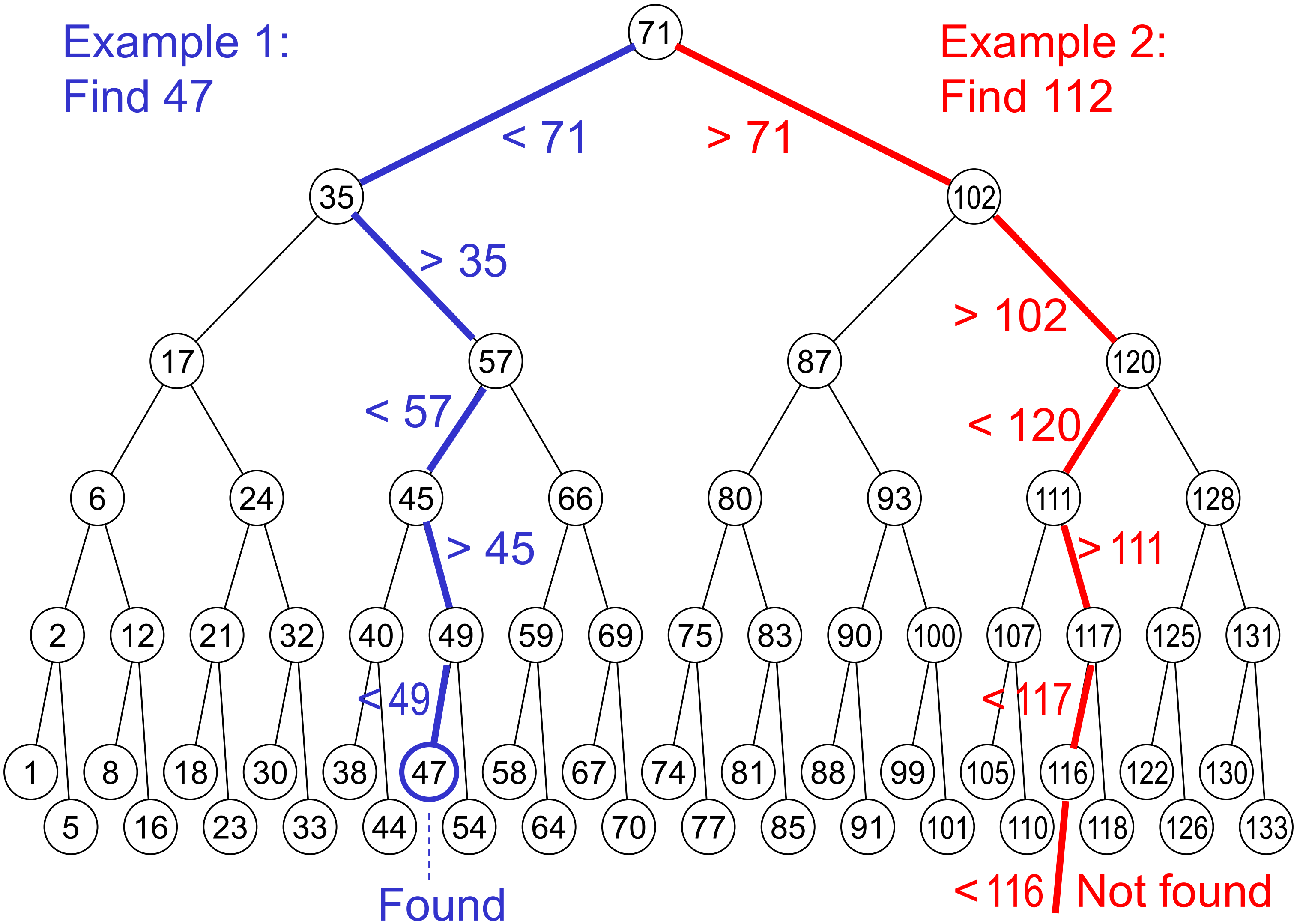
Spell check dictionary for a word processor:
Changes as you add new words

Question: How do we store a rapidly changing dynamic list so that it is easy to search and to update with insertions and deletions?

1	49	91
2	54	93
5	57	99
6	58	100
8	59	101
12	64	102
16	66	105
17	67	107
18	69	110
21	70	111
23	71	116
24	74	117
30	75	118
32	77	120
33	80	122
35	81	125
38	83	126
40	85	128
44	87	130
45	88	131
47	90	133

Stud #s, customer IDs, etc.

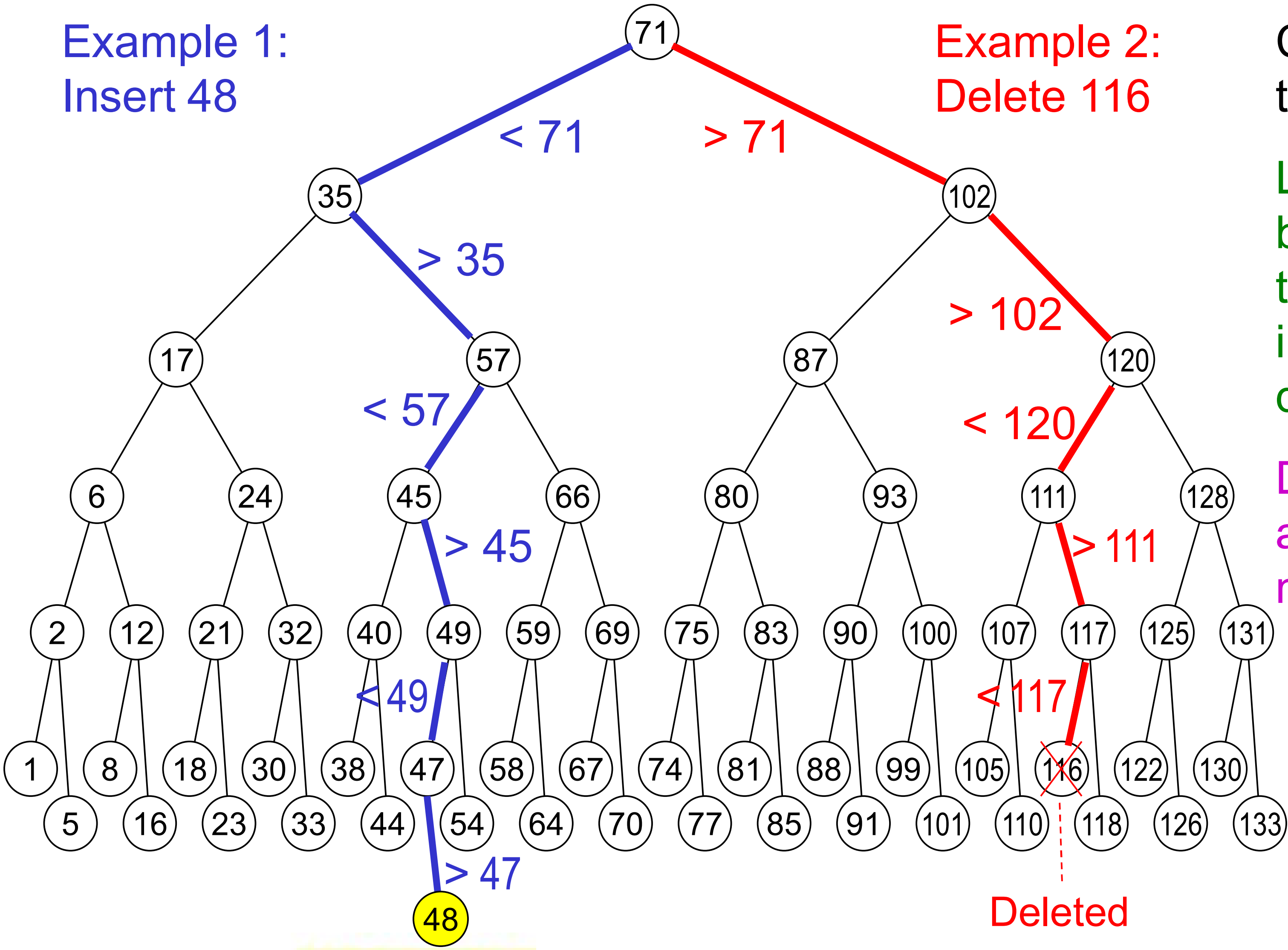
Binary Search Trees



1	49	91
2	54	93
5	57	99
6	58	100
8	59	101
12	64	102
16	66	105
17	67	107
18	69	110
21	70	111
23	71	116
24	74	117
30	75	118
32	77	120
33	80	122
35	81	125
38	83	126
40	85	128
44	87	130
45	88	131
47	90	133

63-item list

Insertions and Deletions in Binary Search Trees

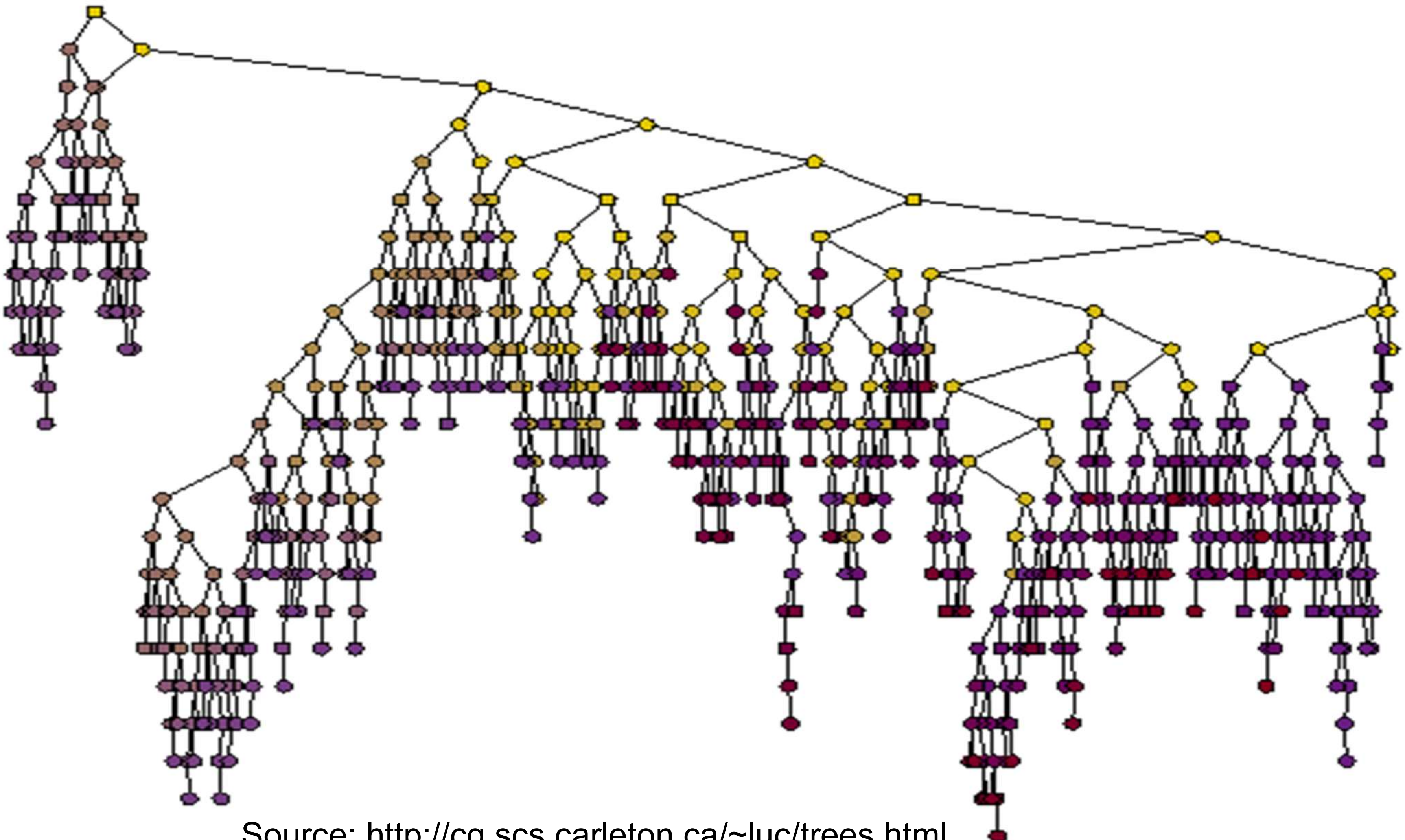


Complications to deal with:

Loss of balance due to repeated insertions and deletions

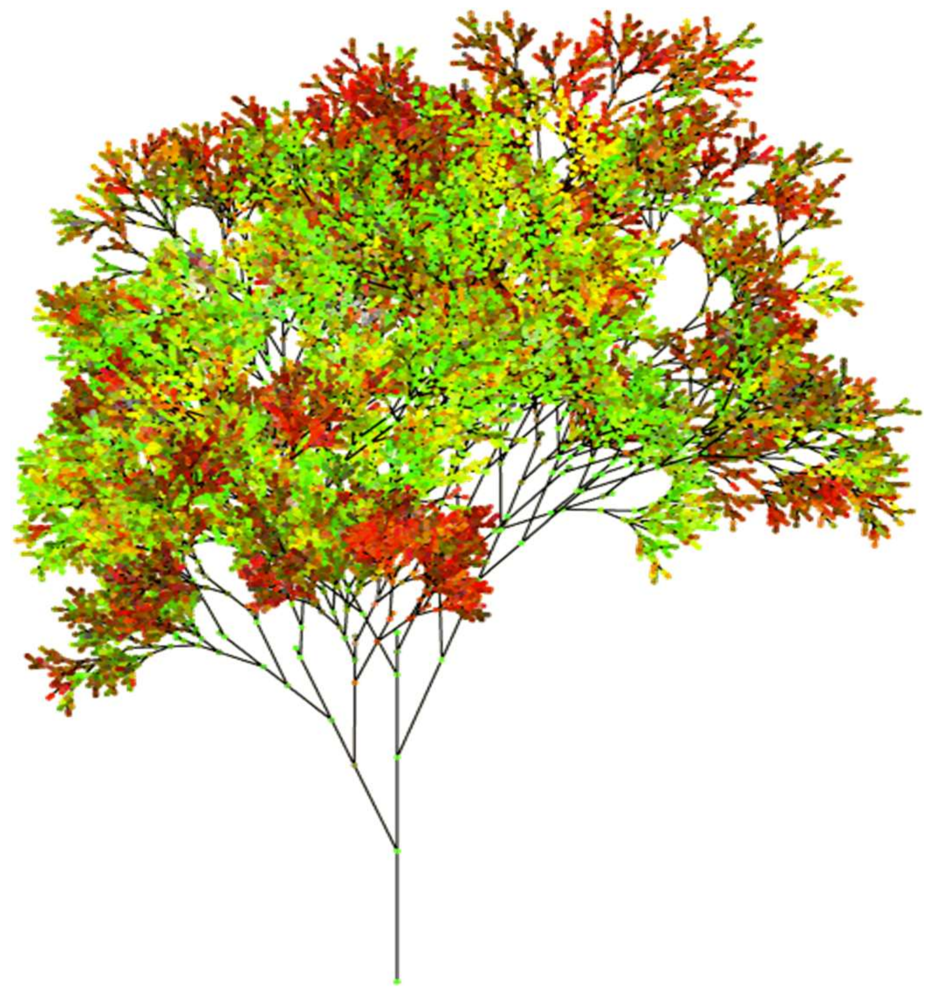
Deletion of an inner or nonleaf node

Example Unbalanced (Random) Binary Tree



Source: http://cg.scs.carleton.ca/~luc/trees.html

Random Binary Trees as Works of Art

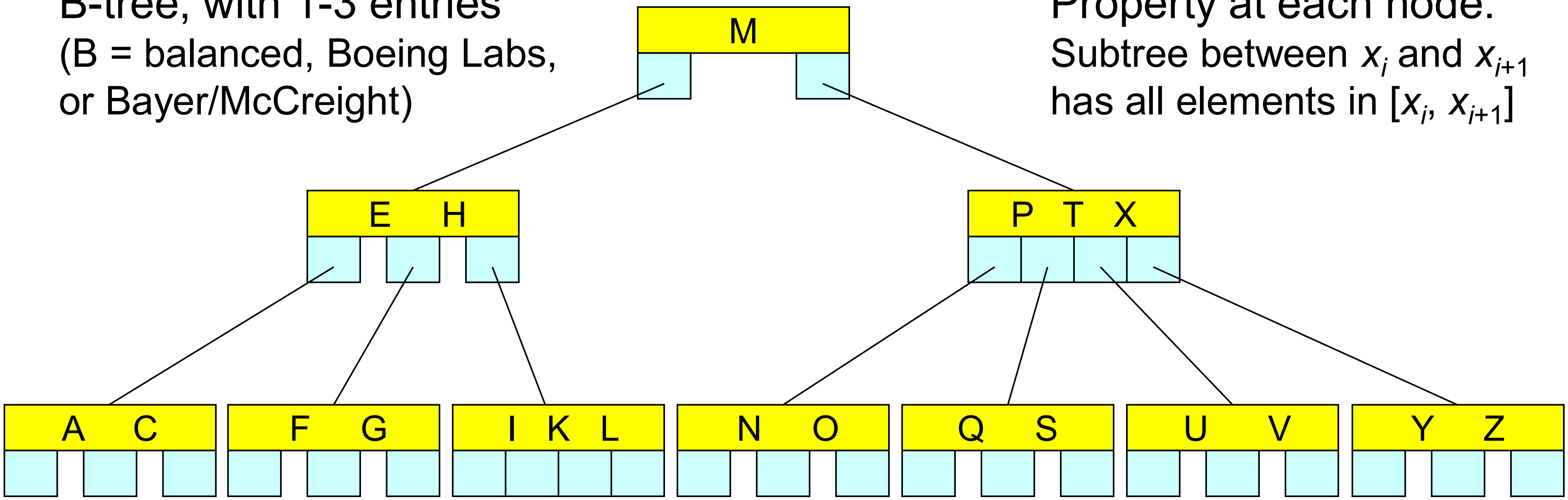


Source: http://cg.scs.carleton.ca/~luc/BRUCE/brucepics.html

Practical Multiway Search Trees

B-tree, with 1-3 entries
(B = balanced, Boeing Labs, or Bayer/McCreight)

Property at each node:
Subtree between x_i and x_{i+1} has all elements in $[x_i, x_{i+1}]$



Find A

Find B

Insert D

Insert J

Delete L

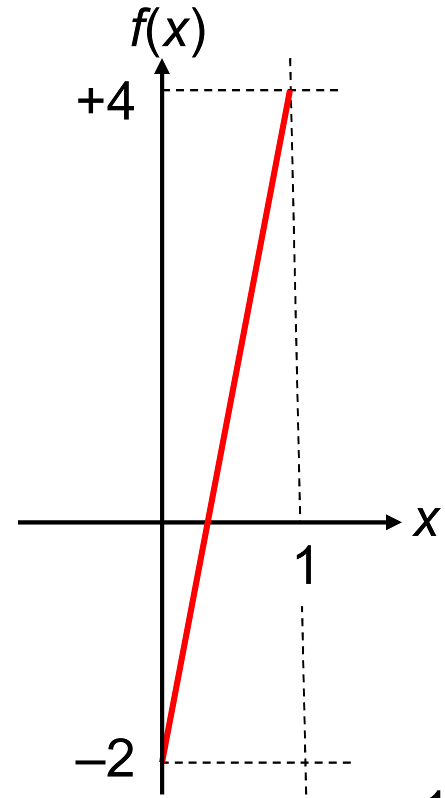
Delete T

Other Applications of Binary Search

Solve the equation $x^4 + 5x - 2 = 0$

$f(x) = x^4 + 5x - 2 = 0$ $\quad f(0) = -2$ $\quad f(1) = 4$

So, there must be a root in [0, 1]

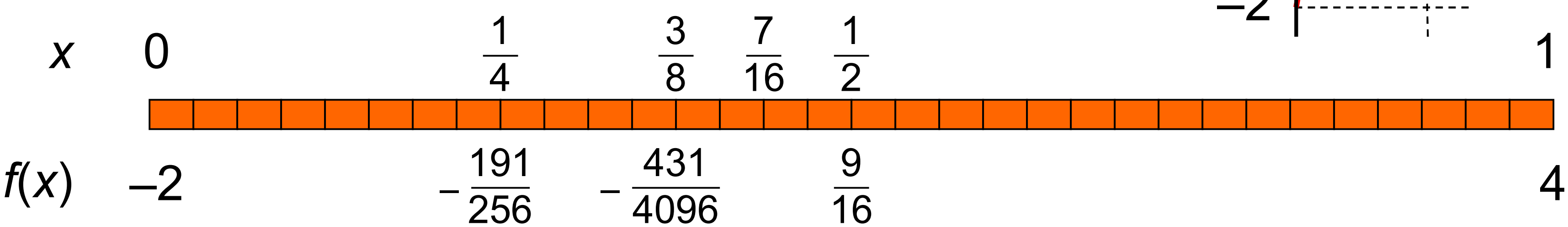


$f(1/2) = +0.5625$ Root in [0, 1/2]

$f(1/4) = -0.7461$ Root in [1/4, 1/2]

$f(3/8) = -0.1052$ Root in [3/8, 1/2]

$f(7/16) = +0.2241$ Root in [3/8, 7/16]



Q3: Continue the root-finding process above until the error is < 0.001.

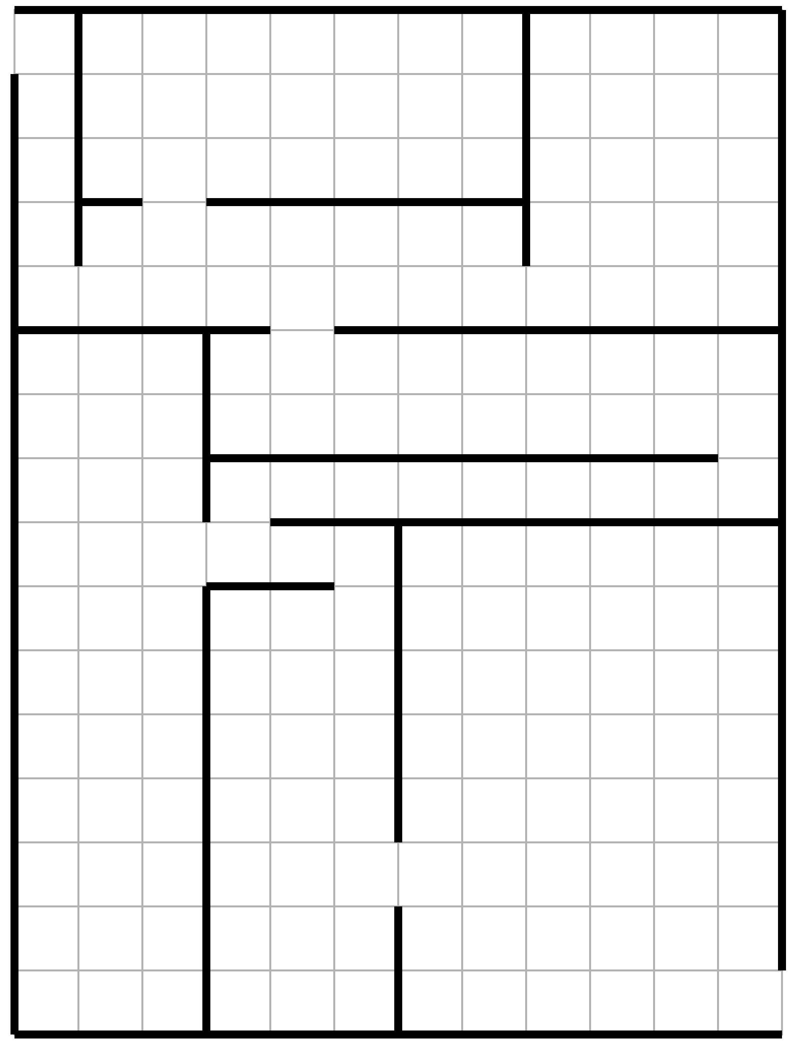
Creating Binary-Tree Mazes

Start with grid and outer walls

Subdivide the area in two parts, with an opening between them

Repeat subdividing process for each of two parts, then for four parts, etc., until no further subdivision is possible

You know you are done when every rectangular area has one side of length 1



Q4: Complete the design of this maze, proceeding until no further subdivision is possible.