Week: 08 Date: 3/25/2021

|  |
| --- |
| 15-110 Recitation Week 8 |

**Reminders for Students**

* HW 4 due Monday 3/29 @ Noon EDT
	+ Bonus Points on HW 4 if you fill out the midsemester course and TA surveys - due same time as HW 4, see Piazza for further details
* How was Quiz 3?
* Usual recitation feedback form: <https://forms.gle/WKrrbawKktmRu1xp9>

**Overview**

* Graphs
* Binary Search Trees
* Graphs Search - BFS/DFS
* Tractability
* P vs. NP

|  |
| --- |
| Problems |

# **BINARY SEARCH TREES PRACTICE**

A quick review of Binary Search Trees!

* Each node can have at most 2 children
* Ordering Requirement:
	+ All nodes in the left subtree of a given parent node must have a smaller value than that of the parent node
	+ All nodes in the right subtree of a given parent node must have a greater value than that of the parent node
	+ There should be no duplicate elements in the tree
* NOTE: trees can be sorted numerically OR alphabetically
	+ We refer to alphabetical order as lexicographic order, “A” → “Z”

KAHOOT!! <https://kahoot.it/>

# **BFS/DFS PRACTICE**

Trace a BFS/DFS on the graph below starting at A and looking for H - search in lexicographic order.

G = {

 “A”: [“B”, “C”, “E”, “G”],

 “B”: [“A”, “D”, “E”],

 “C”: [“A”, “G”, “H”],

 “D”: [“B”, “F”],

 “E”: [“A”, “B”, “F”],

 “F”: [“D”, “E”],

 “G”: [“A”, “C”],

 “H”: [“C”]

 }



BFS:

DFS:

# **TRACTABILITY, P VS. NP**

Notes:

**Quick Questions:**

Some intractable problems are in P?

All problems in P are in NP?

If we find a tractable solution to exam scheduling, then P = NP?

The Travelling Salesman Problem is in NP?

# **GRAPHS CODE WRITING**

Write a function subgraph that takes in as input an unweighted graph G and an input list of nodes nodes and returns a new graph which only contains nodes that are elements of nodes and edges that are between two nodes in nodes.

G = {

 “A”: [“B”, “C”, “E”, “G”],

 “B”: [“A”, “D”, “E”],

 “C”: [“A”, “G”, “H”],

 “D”: [“B”, “F”],

 “E”: [“A”, “B”, “F”],

 “F”: [“D”, “E”],

 “G”: [“A”, “C”],

 “H”: [“C”]

 }

For example, subgraph(G, [“A”, “B”, “E”, “G”, “H”]) would return the graph

newG = {

 “A”: [“B”, “E”, “G”],

 “B”: [“A”, “E”],

 “E”: [“A”, “B”],

 “G”: [“A”],

 “H”: []

 }

Visually, the algorithm takes in the graph to the left below and returns the graph to the right below

#

#

#

#

# Code: