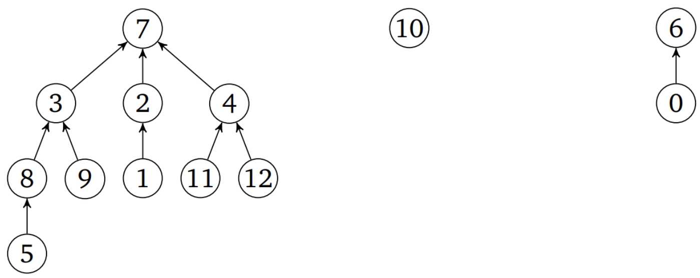
CS251 - Data Structures and Algorithms
Fall 2024

PSO 10, Week 12

Question 1

(Union find)

Consider the following trees, which are a part of a disjoint set.



For the following problems, use both the **union-by-weight** and **path compression**.

(1) Draw the resulting tree(s) after calling find(5). What value does the method return?

(2) Draw the resulting tree(s) of calling union(2,6) on the result of (1).

Question 2

(Minimum spanning tree)

You have found a MST T of a huge Graph G. G has an edge (v_1, v_2) whose weight is 20. After the MST was found the edge weight of (v_1, v_2) is updated. How would you update your MST in the following cases? What's the runtime of your solution? (You may assume that all edges in G have distinct weights before and after we update the weight of the edge (v_1, v_2).)
Note: We are looking for solutions that will update T instead of building a new spanning tree of the updated graph.

1. (v_1, v_2) was in T and the edge weight was updated to 5.
2. (v_1, v_2) was in T and the edge weight was updated to 40.
3. (v_1, v_2) was not in T and the edge weight was updated to 15.
4. (v_1, v_2) was not in T and the edge weight was updated to 25.

Question 3

(More on the MST)

1. An edge is called a **light-edge** crossing a cut $\mathcal{C} := (S, V - S)$, if its weight is the minimum of any edge crossing the cut. Show that:

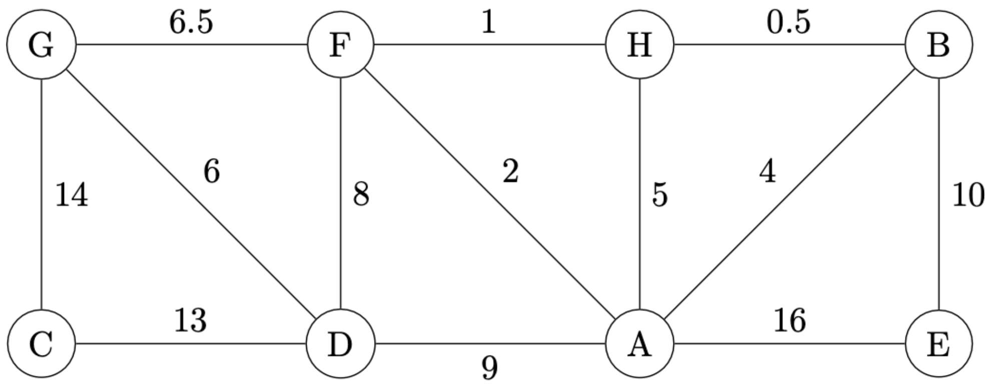
- If an edge (u, v) is contained in some MST (note that MST may not be unique), then it is a light-edge crossing some cut of the graph.
- The converse is not true by giving a simple counter-example of a connected graph such that there exists a cut $\mathcal{C} := (S, V - S)$, in which (u, v) is a light-edge crossing the cut $\mathcal{C}$ but does not form a MST of the graph.

2. Show that a graph has a unique MST, if for every cut of the graph, there is a unique light-edge crossing the cut. Show that the converse is not true by giving a counter-example.

Question 4

(Kruskal's algorithm)

1. Consider the following undirected graph. Assume that the graph is represented in adjacency-list form and that each adjacency-list is given in lexicographic order. List the order that edges are added when we run **Kruskal's algorithm**.



2. Suppose that all edge weights in a graph are integers in the range from 1 to $|V|$. How fast can you make Kruskal's algorithm run?