CSCI 3104 Fall 2022
Instructors: Prof. Grochow and Nagesh

Problem Set 3

Due Date .... September 19, 2022 8pm MT
Name .... **Tyler Huynh**
Student ID **109603994**
Collaborators .... **N/A**

Contents

Instructions

- The solutions **should be typed**, using proper mathematical notation. We cannot accept hand-written solutions. Here's a short intro to LaTeX.
- You should submit your work through the **class Canvas page** only. Please submit one PDF file, compiled using this LaTeX template.
- You may not need a full page for your solutions; pagebreaks are there to help Gradescope automatically find where each problem is. Even if you do not attempt every problem, please submit this document with no fewer pages than the blank template (or Gradescope has issues with it).
- You are welcome and encouraged to collaborate with your classmates, as well as consult outside resources. You must **cite your sources in this document. Copying from any source is an Honor Code violation. Furthermore, all submissions must be in your own words and reflect your understanding of the material.** If there is any confusion about this policy, it is your responsibility to clarify before the due date.

- Posting to **any** service including, but not limited to Chegg, Reddit, StackExchange, etc., for help on an assignment is a violation of the Honor Code.
- You **must** virtually sign the Honor Code (see Section). Failure to do so will result in your assignment not being graded.

Honor Code (Make Sure to Virtually Sign)

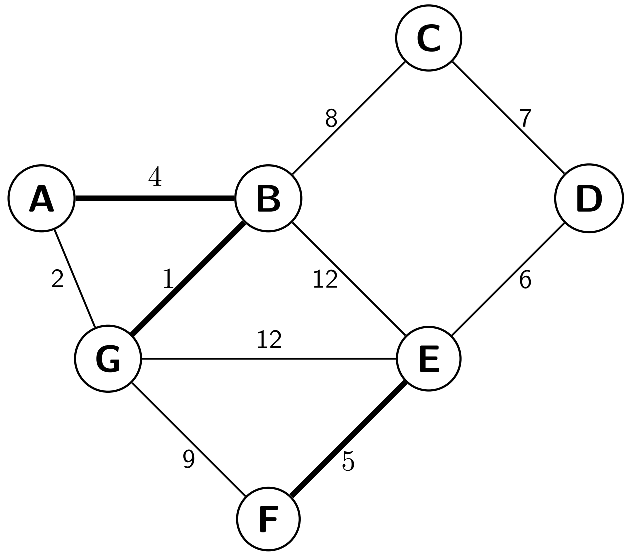
Problem HC.

- My submission is in my own words and reflects my understanding of the material.
- Any collaborations and external sources have been clearly cited in this document.
- I have not posted to external services including, but not limited to Chegg, Reddit, StackExchange, etc.
- I have neither copied nor provided others solutions they can copy.

I agree to the above, Tyler Huynh. $\square$

6 Standard 6 – Safe and Useless Edges

Problem 6. Consider the weighted graph $G(V, E, w)$ below. Let $\mathcal{F} = \{\{A, B\}, \{B, G\}, \{E, F\}\}$ be an intermediate spanning forest (indicated by the thick edges below). Label each edge that is **not** in $\mathcal{F}$ as safe, useless, or undecided. Provide a 1-2 sentence explanation for each such edge.



Answer. Answer

- The edge $\{A, G\}$ is... useless
  This is because ...this edge from {A,G} is useless because by the definition of a useless edge the vertex of A and G both reside in the same spanning forest.

- The edge $\{G, F\}$ is... safe
  This is because ... .this edge is safe because from the vertex G, F it would have the minimum edge weight of 9. Further, it is also an edge that is between two different forest .

- The edge $\{G, E\}$ is... undecided
  This is because ... .this edge is safe because from the vertex G, E it is not the minimum weight edge, but it is also an edge that is between two different forest .

- The edge $\{B, E\}$ is... undecided
  This is because ... this edge is undecided because it exists in the span of two different forest and is not a minimum edge weight.

- The edge $\{B, C\}$ is... safe
  This is because ... this edge is safe because there exists a light edge that belongs to the vertex of C, further it is a minimum edge weight as well.

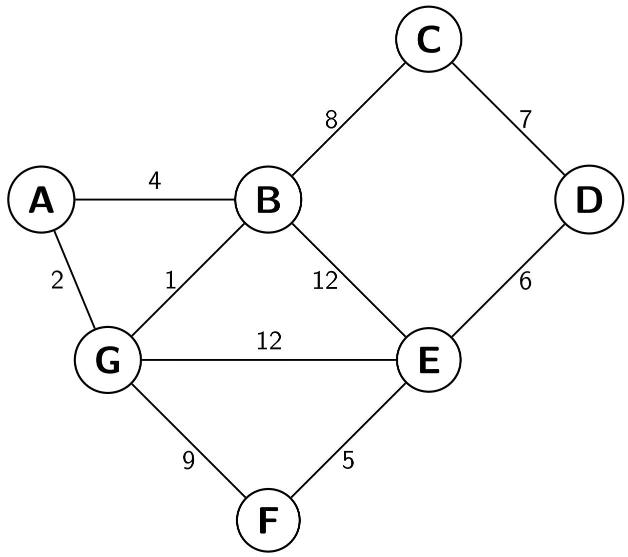
- The edge $\{C, D\}$ is... undecided
  This is because ... This edge would be neither safe or useless since the vertex of c and d do not lie within the spanning forest of {a, b, g} or {e and f}

- The edge $\{D, E\}$ is... safe
  This is because ... This edge would be safe because there exists a light edge that belongs to the vertex of C, further it is a minimum weight edge weight.

□

7 Standard 7: Kruskal's MST Algorithm

Problem 7. Consider the weighted graph $G(V, E, w)$ below. Clearly list the order in which Kruskal's algorithm adds edges to a minimum-weight spanning tree for G. Additionally, clearly articulate the steps that Kruskal's algorithm takes as it selects the first **three** edges.



Proof. Kruskal's algorithm by definition traverses the graph by adding all the vertices with weights to a priority queue organized from least to greatest weights and initialize an empty to a minimum weight spanning tree of G. We will initialize a priority queue containing all the vertices and their respective edge weights:

Vertices:	{G, B}	{A, G}	{A, B}	{E, F}	{E, D}	{C, D}	{B, C}	{G, F}	{G, E}	{B, E}
Edge Weights:	1	2	4	5	6	7	8	9	12	12

Kruskal's Algorithm first three steps:

1st. The first step in this algorithm would be to pop off the shortest edge in the priority queue of vertices {G, B}. Then we will now check if it creates a cycle in our tree, which it does not and add it to our minimum weight spanning tree of G.

2nd. The second step in this algorithm would be to pop off the next shortest edge in the priority queue of vertices {A, G}. We will now check if this set of vertices creates a cycle in our tree, which it does not, thus we will add it to our minimum weight spanning tree of G.

3rd. The third step in this algorithm would be to pop off the next shortest edge in the priority queue of vertices {A, B}. We will now check if this set of vertices creates a cycle in our tree, which it does not, thus we will add it to our minimum weight spanning tree of G.

From the above we will now select the edges such that we will create a minimum edge weight spanning tree of G:

Priority Queue: [{G, B}, {A, G}, {A, B}, {E, F}, {E, D}, {C, D}, {B, C}, {G, F}, {G, E}, {B, E}]

We will first pop off the vertices of {G, B} from our priority queue and check if it creates a cycle, which it does not and add it to our tree of G.

Priority Queue: [{A, G}, {A, B}, {E, F}, {E, D}, {C, D}, {B, C}, {G, F}, {G, E}, {B, E}]
Minimum Weight Spanning Tree G: [{G, B}]

We will now pop off the vertices of {A, G} from our priority queue and check if it creates a cycle, which it does not and add it to our tree of G.

Priority Queue: [{A, B}, {E, F}, {E, D}, {C, D}, {B, C}, {G, F}, {G, E}, {B, E}]
Minimum Weight Spanning Tree G: [{G, B}, {A, G}]

We will now pop off the vertices of {A, B} from our priority queue and check if it creates a cycle, which it does not and add it to our tree of G.

Priority Queue: [{E, F}, {E, D}, {C, D}, {B, C}, {G, F}, {G, E}, {B, E}]
Minimum Weight Spanning Tree G: [{G, B}, {A, G}, {A, B}]

We will now pop off the vertices of {E, F} from our priority queue and check if it creates a cycle, which it does not and add it to our tree of G.

Priority Queue: {E, D}, {C, D}, {B, C}, {G, F}, {G, E}, {B, E}]
Minimum Weight Spanning Tree G: [{G, B}, {A, G}, {A, B}, {E, F}]

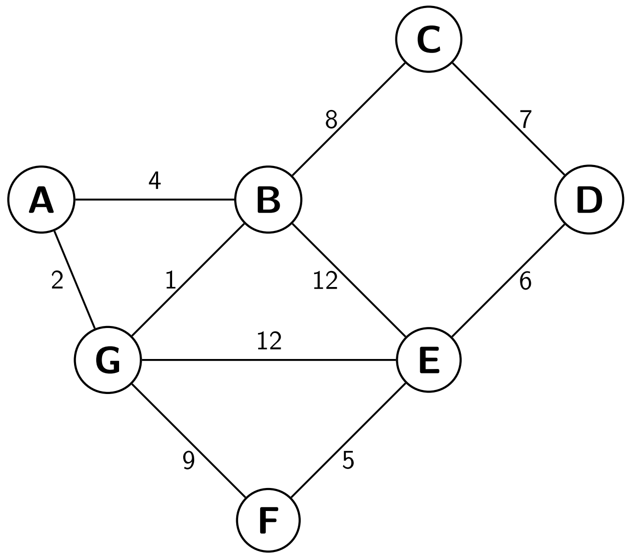
We will now pop off the vertices of {E, D} from our priority queue and check if it creates a cycle, which it does not and add it to our tree of G.

Priority Queue: [{C, D}, {B, C}, {G, F}, {G, E}, {B, E}]
Minimum Weight Spanning Tree G: [{G, B}, {A, G}, {A, B}, {E, F}, {E, D}]

□

8 Standard 8: Prim's MST Algorithm

Problem 8. Consider the weighted graph $G(V, E, w)$ below. Clearly list the order in which Prim's algorithm, **using the source vertex** A, adds edges to a minimum-weight spanning tree for G. Additionally, clearly articulate the steps that Prim's algorithm takes as it selects the first **three** edges.



Proof. □

9 Standard 9: Huffman Coding

Problem 9. Consider the following sequence of numbers:

$$S_n = \begin{cases} 3 & n = 0 \\ 6 & n = 1 \\ S_{n-1} + S_{n-2} & n \geq 2. \end{cases}$$

For an alphabet $\Sigma = \{a, b, c, d, e, f, g, h\}$ with frequencies given by the first $|\Sigma|$ many numbers $S_0, S_1, \ldots, S_{|\Sigma|-1}$, give an optimal Huffman code and its corresponding encoding tree. Specify the frequencies of each letter, and for each stage of the algorithm, the subtrees merged at that stage, and the resulting total frequency for the new merged subtree.

Proof. □