CSCI 3104 Summer 2021
Instructor: Michael Levet

Problem Set 5

06.18.21

Due Date 06.18.21
Name ..... **Brian E. Peterson**
Student ID **109067890**
Collaborators ..... **N/A**

Contents

1 Instructions

- The solutions **must 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 2). Failure to do so will result in your assignment not being graded.

2 Honor Code (Make Sure to Virtually Sign the Honor Pledge)

Problem 1. On my honor, my submission reflects the following:

- 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.

In the specified region below, clearly indicate that you have upheld the Honor Code. Then type your name.

Honor Pledge. On my honor, my submission reflects the following:

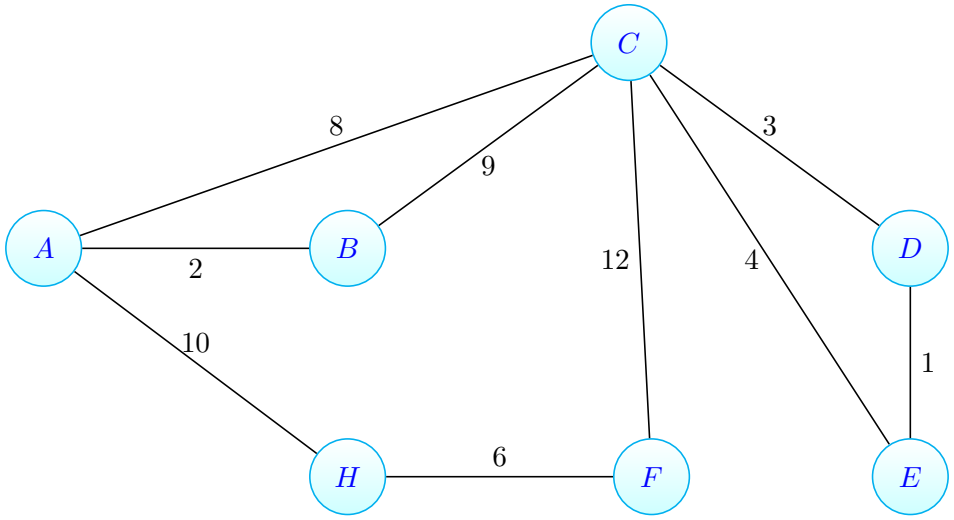
- My submission is in my own words and reflects my understanding of the material.
- Any collaborations and external sources have been clearly cited in this document.
- I have not posted to external services including, but not limited to Chegg, Reddit, StackExchange, etc.
- I have neither copied nor provided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

- Brian E. Peterson □

3 Standard 6- Safe and Useless Edges

3.1 Problem 2

Problem 2. Consider the following graph $G(V, E, w)$.



Consider the intermediate spanning forest $\mathcal{F}$ that contains no edges of G. Clearly identify the safe, useless, and undecided edges. Justify your reasoning. [**Hint:** You may find Corollary 61 on page 42 of the typed lecture notes to be helpful.]

Answer. (D, E) - safe; from either D or E, (D, E) is the minimum weight edge and these are single-vertex components so they have only one endpoint in their respective components

(A, B) - safe; from either A or B, (A, B) is the minimum weight edge and again these are single-vertex components

(C, D) - safe; from either C or D, (C, D) is the minimum weight edge and has only one endpoint in its component

(C, E) - useless; from either C or E, (C, E) is not the minimum weight edge. After forming a tree between C, D, and E, (C, E) is useless because it would form a cycle.

(F, H) - safe; from either F or H, (F, H) is the minimum weight edge and these are single-vertex components

(A, C) - safe; this is the minimum-weight edge connecting (B, A) and (C, D, E) components and has only one endpoint in either component

(B, C) - useless; B and C already form a common component in (B, A, C, D, E) so (B, C) would create a cycle

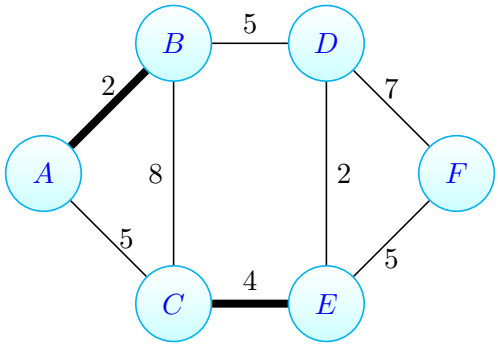
(A, H) - safe; there are two edges connecting (B, A, C, D, E) and (F, H), but this is the minimum of the two and only contains one endpoint in each of the two components

(F, C) - useless; (F, H, A, B, C, D, E) are one component now so there's no reason to connect them. Doing so would create a cycle

□

3.2 Problem 3

Problem 3. Consider the following graph $G(V, E, w)$. Suppose we have the intermediate spanning forest $\mathcal{F}$ (indicated using thick edges) consisting of the edges $\{A, B\}$ and $\{C, E\}$. Clearly identify the safe, useless, and undecided edges. Justify your reasoning. [**Hint:** You may find Corollary 61 on page 42 of the typed lecture notes to be helpful.]



Answer. (D, E) - safe; this is the minimum weight edge connecting (C, E) and D, which are separate components

(A, C) - safe; (A, C) is the minimum weight edge connecting (A, B) and (C, E) and has only one endpoint in each component.

(E, F) - safe; this is the minimum weight edge connecting (C, E, D) to F, which are separate

(B, D) - useless; this would connect B to D, but B and D are already part of the component (B, A, C, E, F, D). This would create a cycle

□

3.3 Problem 4

Problem 4. Let $G(V, E, w)$ be a weighted graph, such that no two edges have the same weight. Let C be a cycle of G, and let e be the edge of C that has the largest weight. Our goal is to show that e does not belong to any minimum-weight spanning tree of G. The proof is outlined below. Your job is to justify each step.

(a) Let T be a spanning tree of G that contains e. Show that removing e disconnects T. That is, $T - e$ has two components.

Answer. Any spanning tree has $n - 1$ edges. Removing any edge from T will therefore reduce this to $n - 2$ edges, resulting in two components forming.

□

(b) Let T_1 and T_2 be the two components of $T - e$. Show that there is an edge $e' \neq e$ that has one endpoint in T_1 and the other endpoint in T_2.

Answer. There does exist a hypothetical edge, $e\prime$ that can connect T_1 and T_2 which is not e. This is because spanning trees are trees, i.e. have n-1 edges connecting each of, in this case, the n vertices in the cycle, C. That is, there is a break somewhere in the cycle, on one side of which the spanning tree starts, and the other on which it ends. Thus, there always exists one "opposite" neighbor one edge away from the break in the cycle. This is $e\prime$. □

(c) Conclude that replacing e with e' in T results in a spanning tree. That is $(T - e) \cup e'$ is also a spanning tree.

Answer. Removing e will either widen this break in the cycle, or add a second one, depending on e's placement with respect to the start and end of the spanning tree. Adding $e\prime$ will shift the break to where e was before, i.e. there will again be $n - 1$ edges, and since T was previously a spanning tree, shifting this break will only create a different spanning tree. □

(d) Explain why $w((T - e) \cup e') < w(T)$. Deduce that e does not belong to any minimum-weight spanning tree of G.

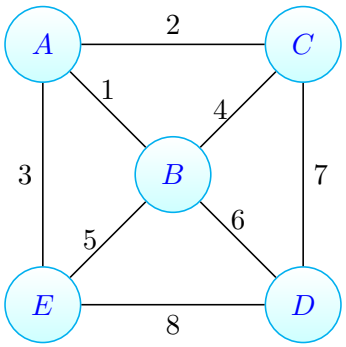
Answer. e is the largest weighted edge in the graph, so replacing it with any other $e\prime$ will result in a lower total weight for the spanning tree. □

4 Standards 7-8: Kruskal's and Prim's Algorithms

Problem 5. This problem will be broken into two parts. Part (a) will count for Standard 7: Kruskal's Algorithm, while part (b) will count for Standard 8: Prim's Algorithm.

The goal of this problem is to construct an example of a weighted graph $G(V, E, w)$ with distinct edge weights such that there exists a source vertex s, where Prim's algorithm (starting at s) adds the edges in a different order than Kruskal's algorithm. Note that as the edge weights are distinct, both algorithms will return the same minimum-weight spanning tree. The only difference is that the edges should be added in different orders.

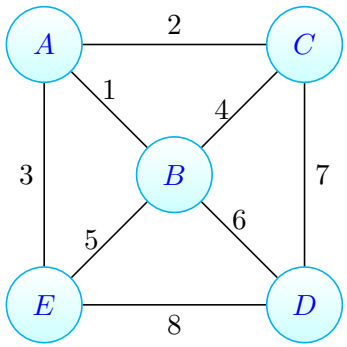
On the graph below, replace the 0's with your desired edge weights. **Do so here.**



Your work for the algorithms will be the next two pages.

4.1 Standard 7: Kruskal's Algorithm

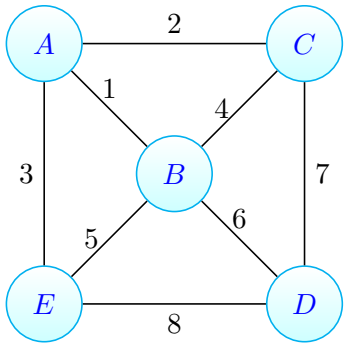
(a) Adjust the figure below to display your edge weights (**which must be the same as the edge weights you assigned at the start of the problem**), replacing 0 with digits of your choice. Then use Kruskal's algorithm *as defined in the lecture notes* to compute the MST. For your answer you must give the order in which Kruskal's algorithm adds the edges to the MST. **For each edge added to the MST, clearly indicate both the edge and its weight.** You do not need to specify the state of the algorithm during its execution (i.e. we do not need to see the state of the disjoint sets data structure in your answer).



Answer. Add (A, B), of weight 1. Add (A, C), of weight 2. Add (A, E), of weight 3. Skip (B, C). Skip (B, E). Add (B, D), of weight 6. Done. Total weight: 12. □

4.2 Standard 8: Prim's Algorithm

(b) Adjust the figure below to display your edge weights (**which must be the same as the edge weights you assigned at the start of the problem**), replacing 0 with digits of your choice. Then use Prim's algorithm *as defined in the lecture notes* to compute the MST. For your answer you must (i) clearly specify the source vertex, and (ii) give the order in which Prim's algorithm adds the edges to the MST. **For each edge added to the MST, clearly indicate both the edge and its weight.** You do not need to specify the state of the algorithm during its execution (i.e. we do not need to see the state of the disjoint sets data structure in your answer).



Answer. Let the source vertex be E. Then, add (A, E) with weight 3. Add (A, B) with weight 1. Add (A, C) with weight 2. Add (B, D) with weight 6. Total weight: 12. □