Name: $\boxed{\text{Daniel Kim}}$
ID: $\boxed{102353420}$

CSCI 3104, Algorithms **Profs. Chen & Grochow**
Problem Set 7 – Due Thur Mar 12 11:55pm **Spring 2020, CU-Boulder**

Advice 1: For every problem in this class, you must justify your answer: show how you arrived at it and why it is correct. If there are assumptions you need to make along the way, state those clearly.

Advice 2: Informal reasoning is typically insufficient for full credit. Instead, write a logical argument, in the style of a mathematical proof.

Instructions for submitting your solutions:

- All submissions must be typed.
- You should submit your work through the **class Canvas page** only.
- 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 allot at least as many pages per problem (or subproblem) as are allotted in this template.

Quicklinks: 1 2 3 3a 3b 3c 3d

Name: Daniel Kim
ID: 102353420

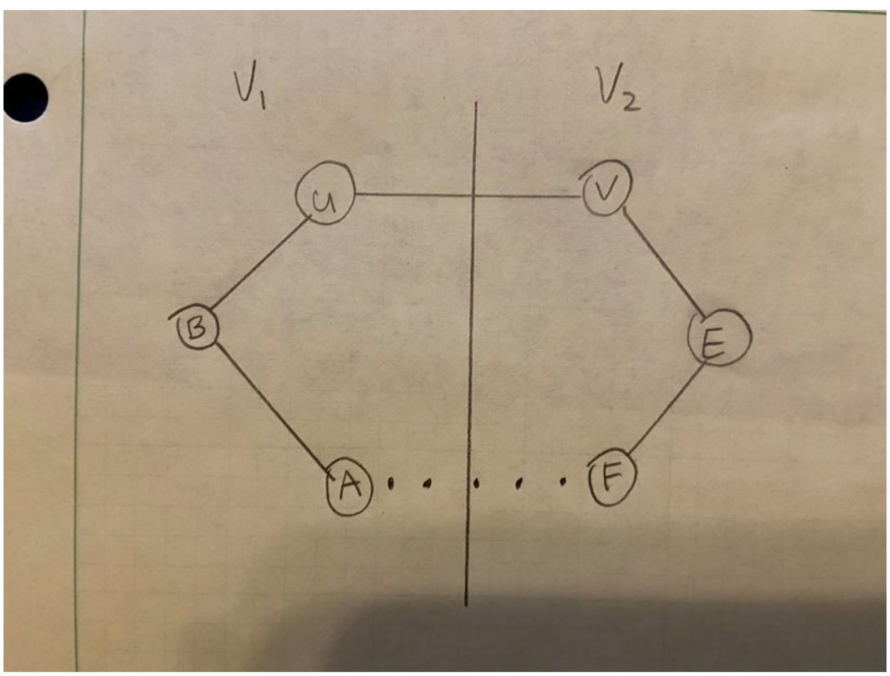
CSCI 3104, Algorithms
Problem Set 7 – Due Thur Mar 12 11:55pm

Profs. Chen & Grochow
Spring 2020, CU-Boulder

1. *Let $G(V, E, w)$ be a weighted graph. The* **Cycle Property** *provides conditions for when an edge is* ***not*** *included in any minimum spanning tree (MST) of G. The* ***Cycle Property*** *is stated as follows. Let C be a cycle in G, and let edge $e = (u, v)$ be a maximum-cost edge on C. Then the edge e does not belong to any MST of G.*

   *Use an exchange argument to show why this property holds. You may assume that all edge costs are distinct. [**Note:** You may freely use any of the tree properties covered in Michael's Tree Notes on Canvas.]*

   Let's assume that the e is an edge in a MST tree T, and if we remove e from T, we get $T - (e)$ which is not a tree. Vertices u and v also disconnects because e was removed. From the problem above, e is the maximum-cost edge on C, and C is a cycle. So $T - (e)$ is divided into two part component which are V_1 and V_2 just like the graph below. In addition, there must be existing vertex that can connect V_1 and V_2. Let say that vertex A and F connects by an edge, so it should be $w(u, v) < w(A, F)$, because e is the maximum-cost edge. If we move (A, F) to $T - (e)$, we get a new spanning tree that weighs smaller than T. Therefore, T is not a MST of G, and the edge e does not belong to any MST of G

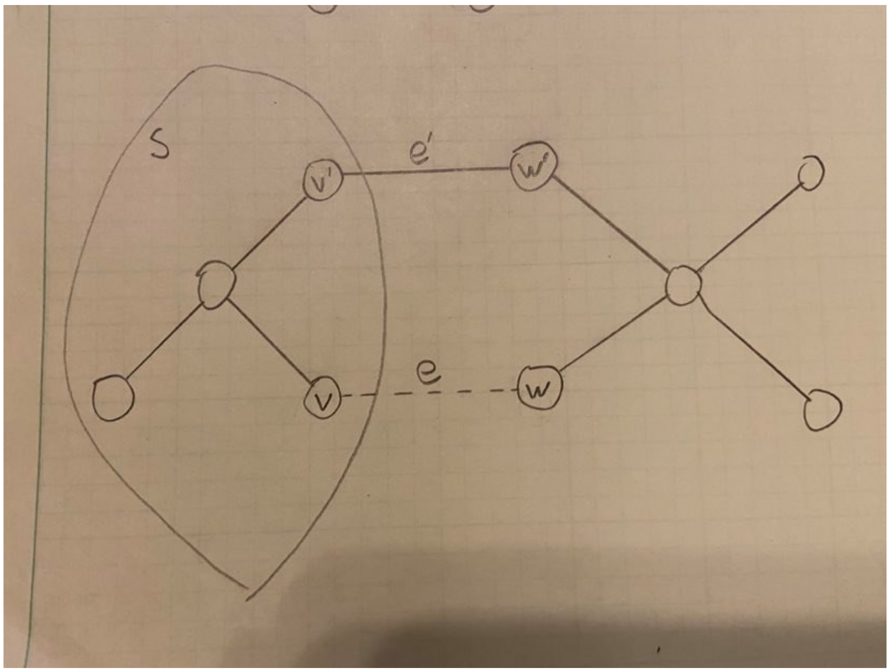


2. *Let $G(V, E, w)$ be a connected, weighted graph. A cut of G is a set $C \subset E$ of edges such that removing the edges of C from G disconnects the graph. The* **Cut Property** *provides conditions for when an edge* ***must*** *be included in a minimum spanning tree (MST) of G. The* ***Cut Property*** *is stated as follows. Let $C \subset E$ be a cut. Suppose $e = (u, v)$ is the minimum-weight edge in C. Then every MST contains the edge e.*

   *Use an exchange argument to show why this property holds. You may assume that all edge costs are distinct. [**Note:** You may freely use any of the tree properties covered in Michael's Tree Notes on Canvas.]*

   Let's say that edge e is the minimum cost edge and T be a spanning tree that does not contain e. In addition, there is edge e' which is part of T and more expensive than e. At each ends of e, there are nodes a and b, so there must be a path P in T from a to b. Starting at a, suppose we follow the nodes of P in sequence. There is a first node b' on P that is in $V - S$. Let $a' \in S$ be the node just before b' on P, and let $e' = (a', b')$ be the edge joining them. e' is an edge of T with one end in S and the other in $V - S$. (See example below).
   If we exchange e for e', we get a set of edges $T' = T - e' \cup e$. We believe that T' is a spanning tree and therefore (V, T') is connected. Because (V, T) is connected, any path in (V, T) that used the edge e' =(a', b') can now be "rerouted" in (V, T') to follow the portion of P from a' to a, and the edge e and then the portion of P from b to b'. Due to the deletion of e', we also need to see if (V, T') is also acyclic looking at the only cycle in $(V, T' \cup e')$ is the one composed of e and the path P, and this cycle is not present in (V, T')

   We have proved above that the edge e' has one end in S and other in $V - S$. However, e is the cheapest edge with this property, and so $C_e < C_{e'}$. The total cost of T' is less than that of T.


S
v'
e'
w'
v
e
w

Name: Daniel Kim

ID: 102353420

CSCI 3104, Algorithms **Profs. Chen & Grochow**

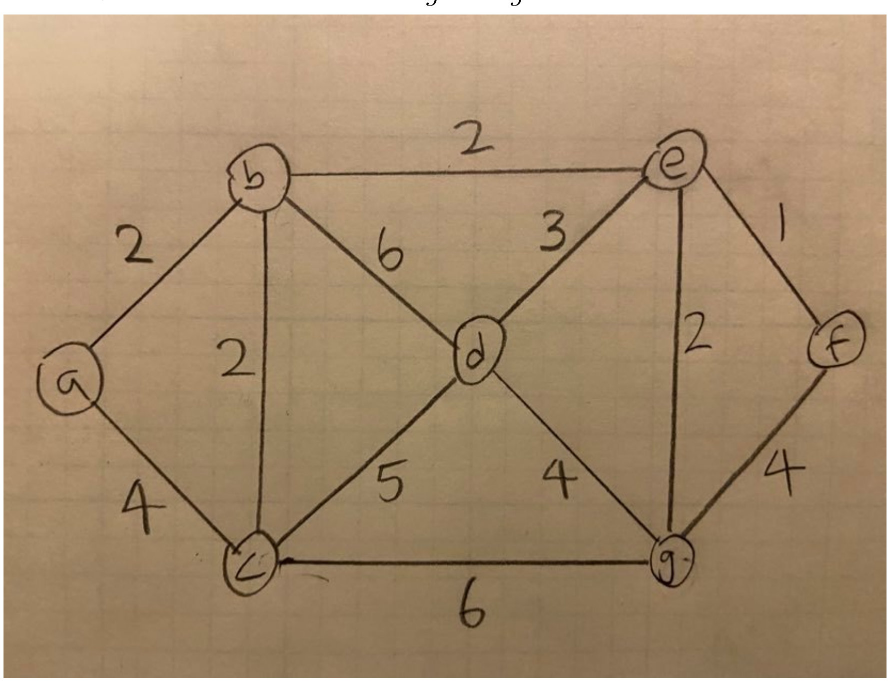
Problem Set 7 – Due Thur Mar 12 11:55pm **Spring 2020, CU-Boulder**

3. *Given the following unweighted graph, you're asked to assign a positive integer weight to each edge, such that the following properties are true regarding minimum spanning trees (MST) and single-source shortest path (SSSP) trees:*

   - *All edge weights are distinct*
   - *The MST is distinct from any of the seven SSSP trees. Note that there is a SSSP tree for each node as the source.*
   - *The order in which Prim's algorithm adds the safe edges is different from the order in which Kruskal's algorithm adds them.*

graph_mst.pdf

   (a) *Put your picture of the graph (the same one as above)* ***with your edge weights*** *here. Remember that all the edge weights should be distinct:*

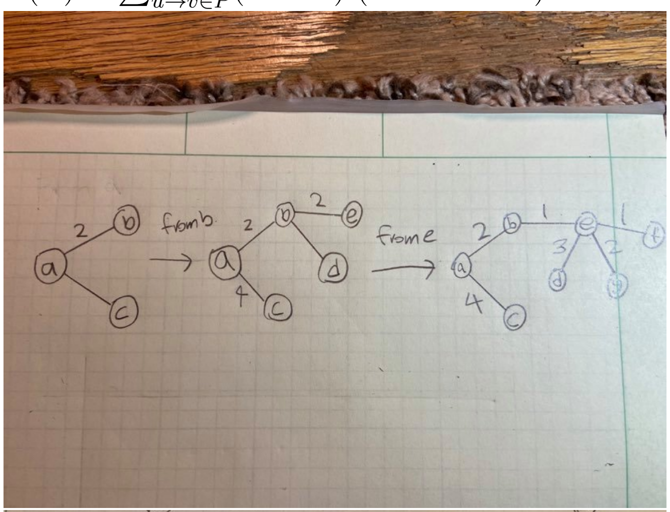


Name: Daniel Kim
ID: 102353420

CSCI 3104, Algorithms **Profs. Chen & Grochow**
Problem Set 7 – Due Thur Mar 12 11:55pm **Spring 2020, CU-Boulder**

(b) *Draw the seven SSSP trees (one for each starting vertex). In each tree, indicate which vertex you used as the starting vertex. For one such tree, clearly articulate the steps taken to select the first three edges using **Dijkstra's algorithm**.*

We need to take the source node a
$w(P) = \sum_{u \to v \in P}(w \to v)$ (P is vertices)

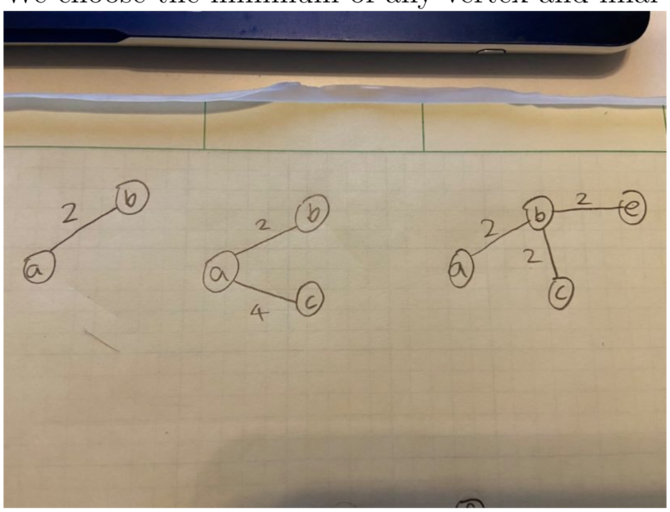


	a	b	c	d	e	f	g
	0	∞	∞	∞	∞	∞	∞
a		2	4	∞	∞	∞	∞
b			4	8	4	∞	∞
c				8	4	∞	0
e				7		5	6
f				7			6
g				7			
d							

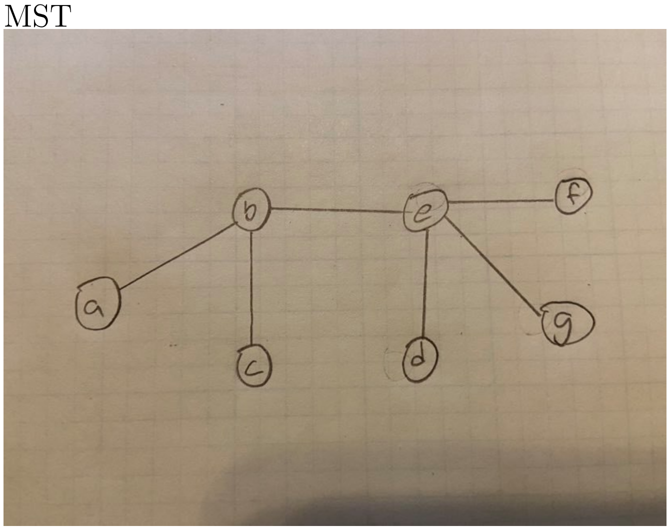
The order is $a \to b \to c \to e \to f \to g \to d$

(c) *Use Prim's algorithm to compute the MST. List the order in which Prim's algorithm adds the edges, and draw the MST. Furthermore, clearly articulate the steps the algorithm takes as it selects the first three edges.*

We choose the minimum of any vertex and final address as that vertex



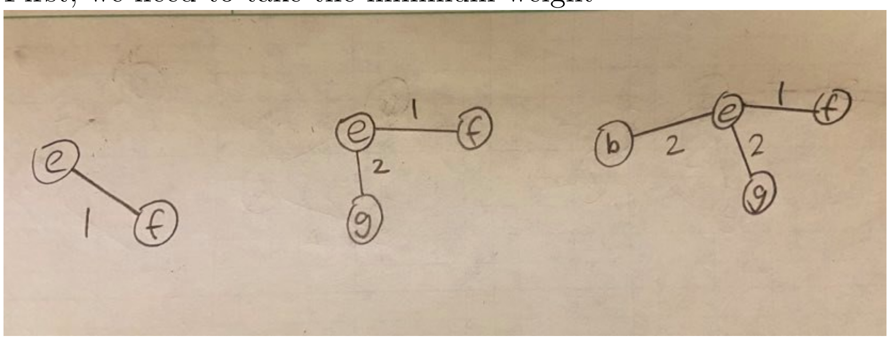
Find adi as a new vertex and take min
MST



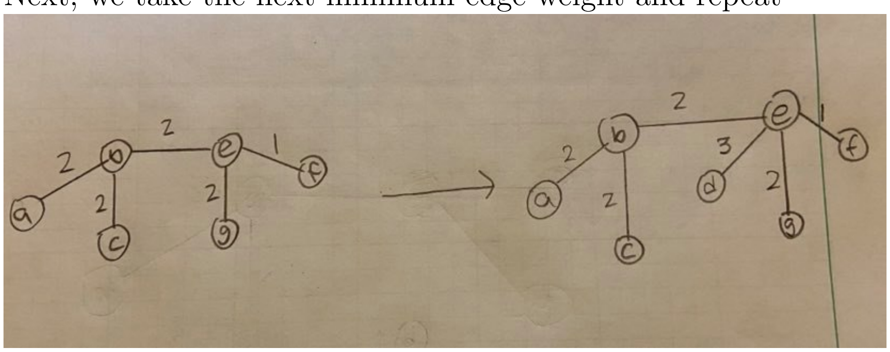
The order of adding these edges is $a \to b \to c \to e \to f \to g \to d$

(d) *List the order in which Kruskal's algorithm adds the edges. Furthermore, clearly articulate the steps the algorithm takes as it selects the first three edges.*

First, we need to take the minimum weight



Next, we take the next minimum edge weight and repeat



The order of adding edges is $e \to f \to g \to b \to a \to c \to d$

Name: Daniel Kim
ID: 102353420

COLLABORATED WITH:
RUIJIANG MA
JINGQI LIU