Name: Mauro Vargas Jr

ID: 107212593

CSCI 3104, Algorithms **Escobedo & Jahagirdar**
Homework 6 (100 points) **Summer 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: Verbal reasoning is typically insufficient for full credit. Instead, write a logical argument, in the style of a mathematical proof.

Instructions for submitting your solution:

- The solutions **should be typed**, we cannot accept hand-written solutions. Here's a short intro to **Latex**.
- In this homework we denote the asymptomatic *Big-O* notation by $\mathcal{O}$ and *Small-O* notation is represented as o.
- We recommend using online Latex editor **Overleaf**. Download the **.tex** file from Canvas and upload it on overleaf to edit.
- You should submit your work through **Gradescope** only.
- If you don't have an account on it, sign up for one using your CU email. You should have gotten an email to sign up. If your name based CU email doesn't work, try the identikey@colorado.edu version.
- Gradescope will only accept **.pdf** files (except for code files that should be submitted separately on Canvas if a problem set has them) and **try to fit your work in the box provided**.
- You cannot submit a pdf which has less pages than what we provided you as Gradescope won't allow it.

Name: Mauro Vargas Jr

ID: 107212593

CSCI 3104, Algorithms **Escobedo & Jahagirdar**
Homework 6 (100 points) **Summer 2020, CU-Boulder**

Piazza threads for hints and further discussion

Piazza Threads
Question 1
Question 2
Question 3
Question 4
Question 5
Question 6
Question 7
Question 8

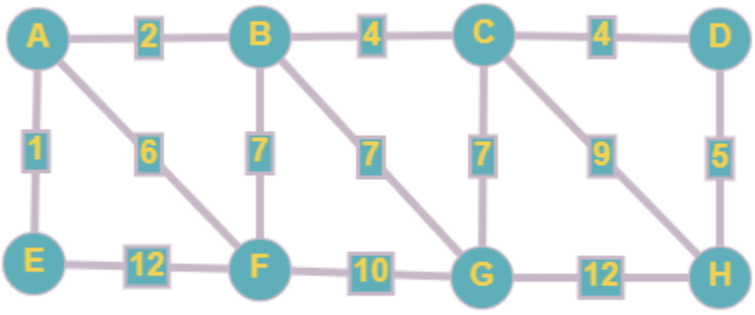
Recommended reading:
Graph Algorithms Intro: Ch. 22 → 22.1, 22.2, 22.3
Graph Algorithms SSSPs: Ch. 24 → 24.3

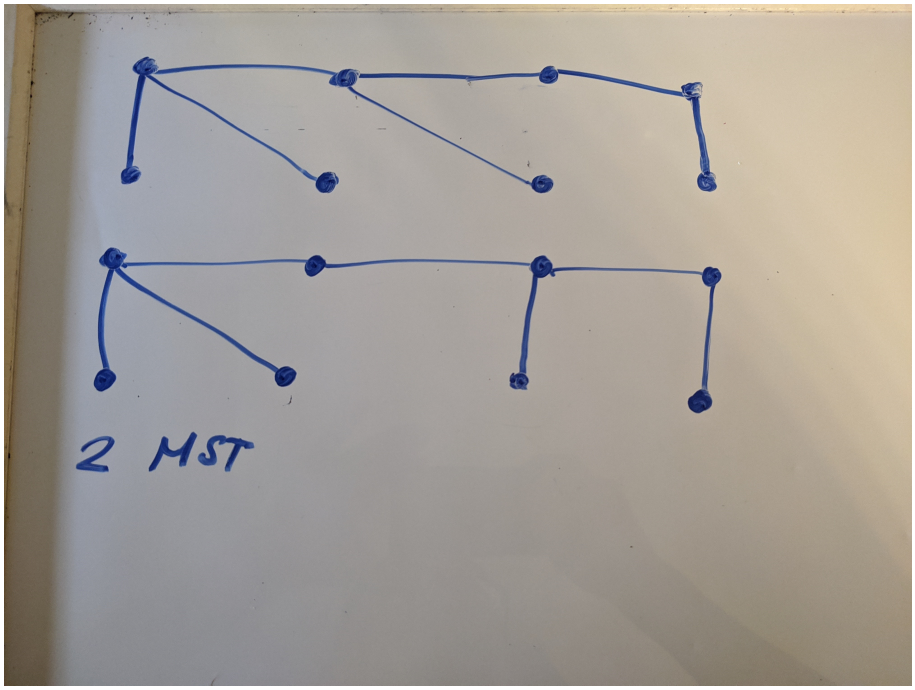
Name: Mauro Vargas Jr

ID: 107212593

CSCI 3104, Algorithms **Escobedo & Jahagirdar**
Homework 6 (100 points) **Summer 2020, CU-Boulder**

1. (5 pts) How many unique MSTs does the following graph have. Show the necessary work to justify your answer.



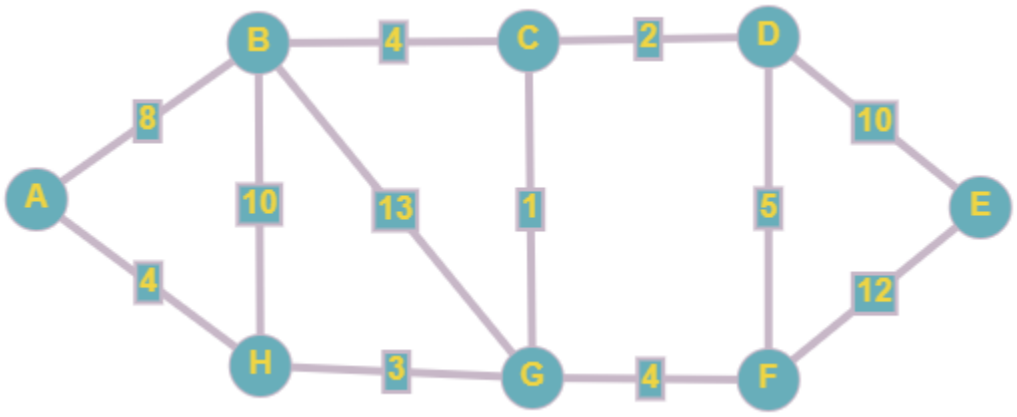


Name: Mauro Vargas Jr

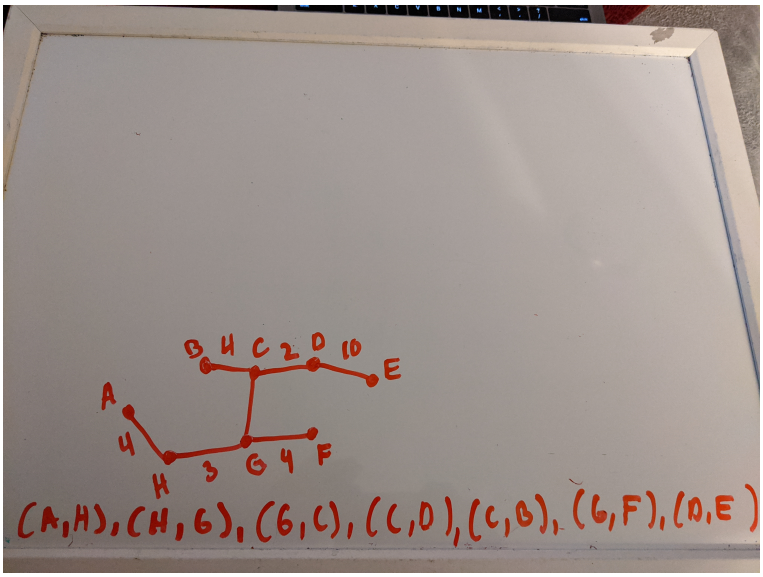
ID: 107212593

CSCI 3104, Algorithms **Escobedo & Jahagirdar**
Homework 6 (100 points) **Summer 2020, CU-Boulder**

2. (20 pts) Based on the following graph :



(a) (10 pts) In what order would Prim's algorithm add edges to the MST if we start at vertex A?

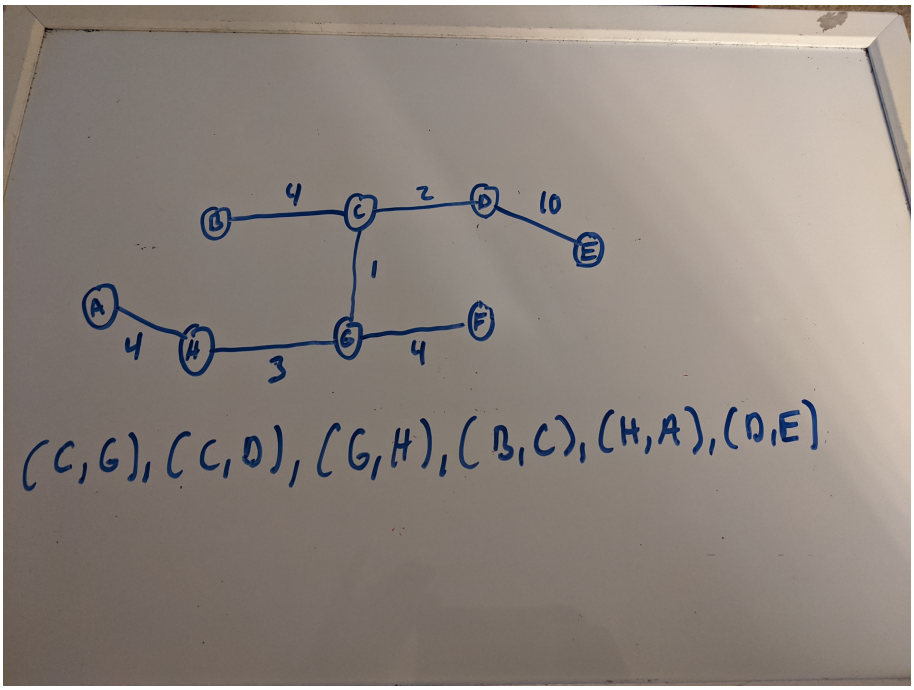


Name: Mauro Vargas Jr

ID: 107212593

CSCI 3104, Algorithms **Escobedo & Jahagirdar**
Homework 6 (100 points) **Summer 2020, CU-Boulder**

(b) (10 pts) In what order Kruskal's would add the edges to the MST?

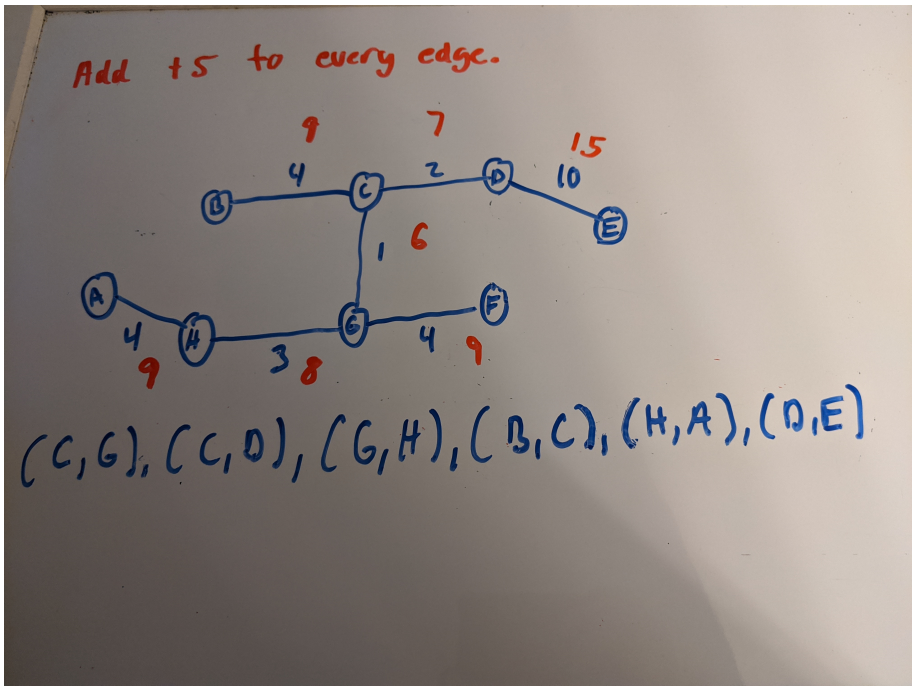


Name: Mauro Vargas Jr

ID: 107212593

CSCI 3104, Algorithms **Escobedo & Jahagirdar**
Homework 6 (100 points) **Summer 2020, CU-Boulder**

3. (10 pts) Suppose that you have calculated the MST of an undirected graph $G = (V, E)$ with positive edge weights. If you increase each edge weight by 5, will the MST change? Prove that it cannot change or give a counterexample if it changes. (Note: Your proof, if there is one, can be a simple logical argument.)



If we add +5 to every edge the order in which we add the edges will not change, since the weight is distributed evenly through out the MST.

Name: Mauro Vargas Jr

ID: 107212593

CSCI 3104, Algorithms **Escobedo & Jahagirdar**
Homework 6 (100 points) **Summer 2020, CU-Boulder**

4. (15 pts) Suppose you are given the minimum spanning tree T of a given graph G (with n vertices and m edges) and a new edge $e = (u, v)$ of weight w that will be added to G. Give an efficient algorithm to find the MST of the graph $G \cup e$. Your algorithm should run in $O(n)$ time.

We can use Boruvka's Algorithm to solve this problem by adding all current safe edges simultaneously and then recursing until there are no safe edges left.

```
Boruvka(G,w){
    F = {V,{}}
    While F has more than one component
        choose component leaders via DFS/BFS
        Find-Safe-Edges(G,w)
        for each leader v' {
            add safe(v')
        }
}
Find-Safe-Edges(G,w){
    for each leader v'{
         safe(v') = NULL
    }
    for each edge (u,v) in E{
        u' = leader(u) // look up component name of u
        v' = leader(v) // look up component name of v
        if u' not = v'{ // are u, v in same component
            if w(u,v) < w(safe(u')){ // update estimate safe egde
                safe(u') = (u,v)
            }
            if w(u,v) < w(safe(v')){
                safe(v') = (u,v)
            }
        }
    }
```

Name: Mauro Vargas Jr

ID: 107212593

CSCI 3104, Algorithms **Escobedo & Jahagirdar**
Homework 6 (100 points) **Summer 2020, CU-Boulder**

5. (20 pts) In a directed graph $G = (V, E)$ with positive edge weights, we define the maximum weight of a path from s to t to be the maximum of the edge weights along the path. For example, if the path from s to t has edges with weights $e_1 = 10, e_2 = 15, e_3 = 5$ then the maximum of the path is $max(e_1, e_2, e_3) = 15$. Give an algorithm to compute the smallest maximum weight paths from a source vertex s to all other vertices. (Hint: Your algorithm should be a modification of Dijkstra's algorithm)

We can use a modified version of Dijkstra's algorithm, Since it finds the MST of our graph. If we find a new path we need to relax all the edges and find the Maximum weight for that current path. Once we have gone through through the whole graph we can return the Maximum number of weight from path s to t.

```
DIJKSTRA_MOD(G,W,S)
    INITIALIZE-SINGLE-SOURCE(G,S)
    S = [] //visited nodes
    Q = G.V //add all nodes to queue
    MAX = 0 // keep track of max weight in MST.
    while Q is not empty
        u = EXTRACT-MIN
        S = S Union {u}
        for each vertex v is an element of G .Adj[u]
            Relax(u,v,w) // if we have a path with a lower weight, we can change
            //if we change paths we will relax all the edges and see if there is
            if(w > MAX) //if we find a bigger max weight update
                MAX == w
```

Name: Mauro Vargas Jr

ID: 107212593

CSCI 3104, Algorithms **Escobedo & Jahagirdar**
Homework 6 (100 points) **Summer 2020, CU-Boulder**

6. (2 pts) What are the two conditions that must be met for the flow on a graph to be valid?

A flow in **G** is a real - valued function f: V * V $\rightarrow$ that satisfies the following properties:
Capacity constraint: For all u, v, $\in V$,we require $0 \leq$ f(u,v) $\leq$ c(u,v).
Flow conservation: For all u $\in$ V - {s,t}, we require

$$\sum f(v,u) = \sum f(u,v)$$

.
When (u,v) $\notin E$, there can be no flow from u to v, and f(u,v) = 0.
The flow-conservation property says that the total flow into a vertex other than the source or sink must equal the total flow out of that vertex-informally, 'flow in equals flow out'

7. (3 pts) What do the edge weights in the residual graph G_f represent? Include the definition of both forward and backward edges.

Residual Graph "(G_f)" is a graph which indicates additional possible flow. If there is such path from source to sink then is a possibility to add flow. Residual Capacity is the original capacity of the edge minus flow. A **cut** is a partition of vertices (V_s, V_t) such that the $s \in V_s$ and $t \in V_t$. An edge that goes from u to v is a forward edge. If the opposite is true, then it is a backward edge from v to u. Note to let **F** equals the flow of a network.

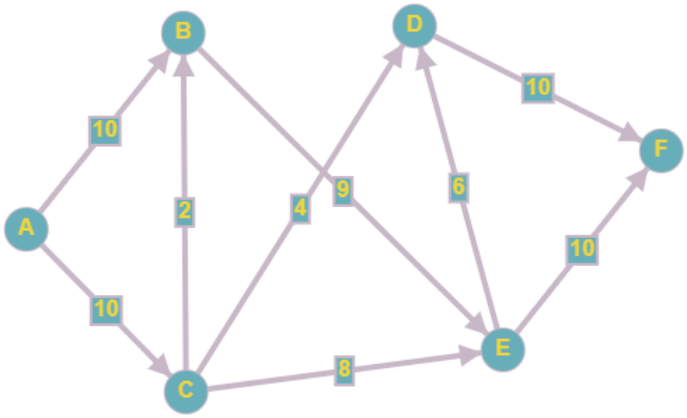
Name: Mauro Vargas Jr

ID: 107212593

CSCI 3104, Algorithms **Escobedo & Jahagirdar**

Homework 6 (100 points) **Summer 2020, CU-Boulder**

(25 pts) Based on the following network and the given edge capacities answer the following.



1. (15 pts) Suppose we start the Ford-Fulkerson algorithm with **A** being the source vertex and **F** being the sink and **select the path** $A->C->B->E->D->F$ **in the first iteration (Do not chose the first A-F path on your own).** Complete all the iterations of Ford-Fulkerson to find the Max-Flow (including the first round that is incomplete). Clearly show each round with

   (a) The path that you are selecting in that round.
   (b) The bottleneck edge on this path.
   (c) The additional flow that you push from the source by augmenting (pushing maximum allowed flow along) this selected augmenting path.
   (d) The residual graph with the residual capacities (on both the forward and backward) edges.

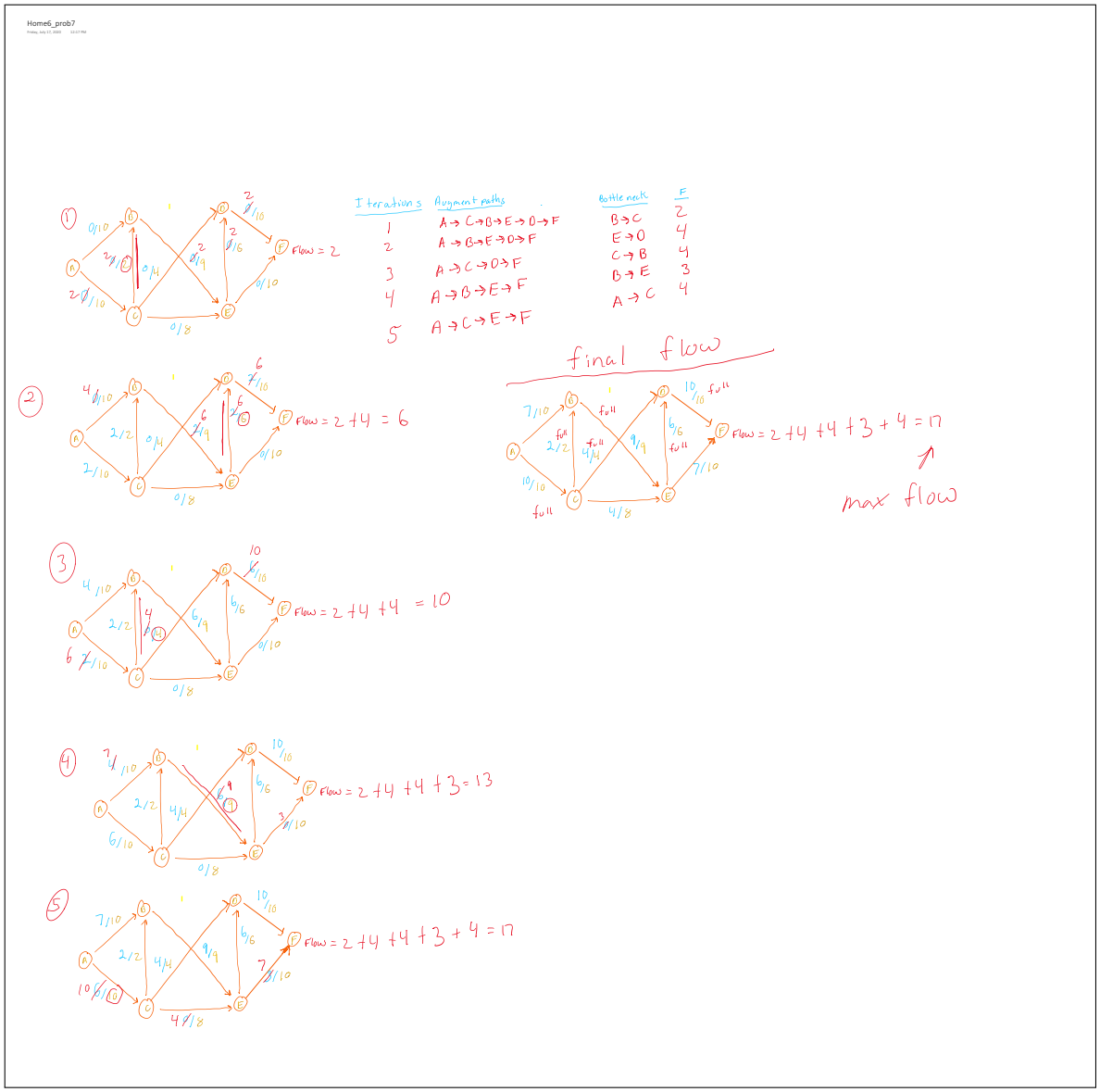
   Also, report the Max-Flow after the algorithm terminates.

Name: Mauro Vargas Jr

ID: 107212593

CSCI 3104, Algorithms **Escobedo & Jahagirdar**

Homework 6 (100 points) **Summer 2020, CU-Boulder**

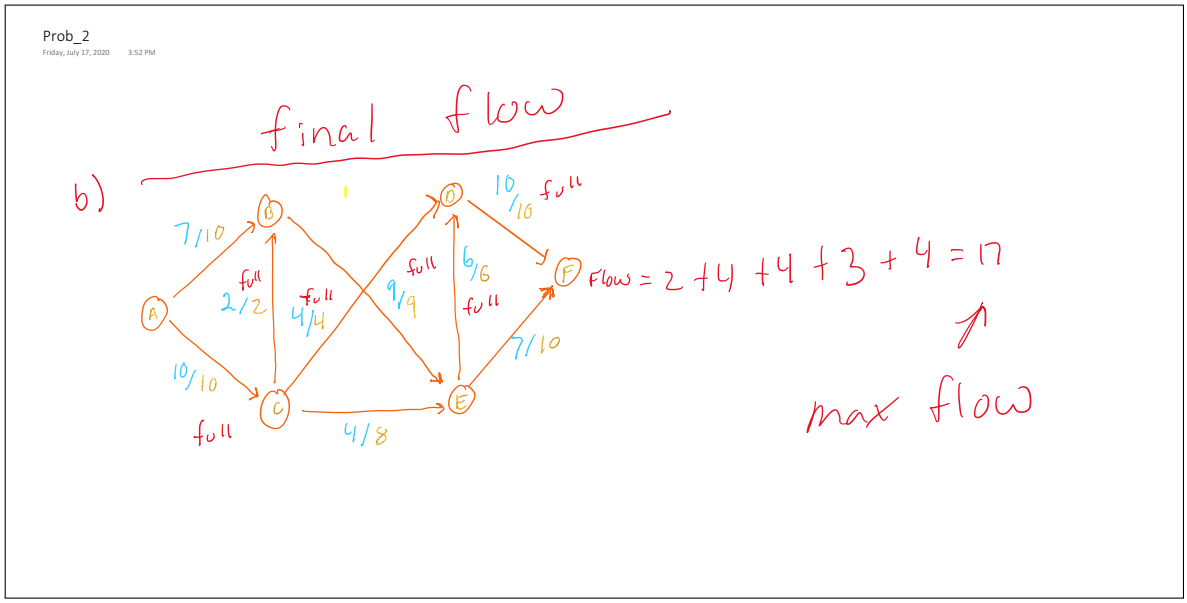


Name: Mauro Vargas Jr

ID: 107212593

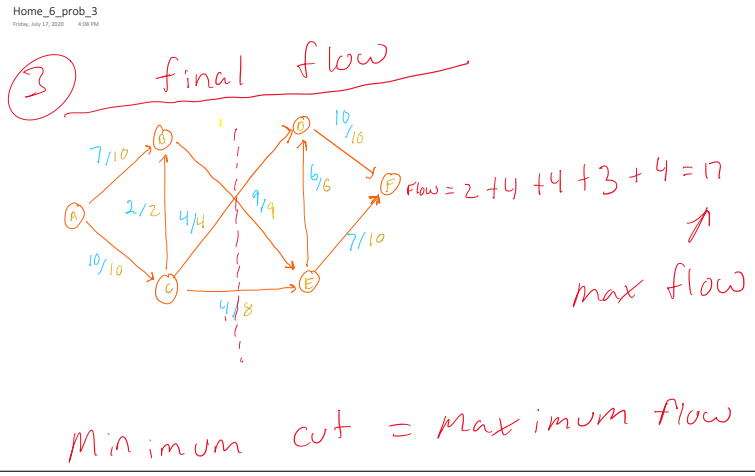
CSCI 3104, Algorithms **Escobedo & Jahagirdar**
Homework 6 (100 points) **Summer 2020, CU-Boulder**

2. (5 pts) Draw the graph and show the final flow f(e) for the edges of the original graph when the Ford-Fulkerson algorithm terminates.



3. (5 pts) Find the minimum capacity cut with respect to the capacities on the original graph. Is this minimum capacity equal to the Max-Flow that you earlier identified? Justify your answer in a sentence. Also, list the edges that are part of the min-cut and are saturated (can't carry any more flow).

You can see that the minimum cut is equal to max flow by taking all the vertices flow at the minimum cut and adding them all together $4 + 4 + 9 = 17$. Vertices at minimum cut: $B \to E$, $C \to D$, $C \to E$.



Name: Mauro Vargas Jr

ID: 107212593

CSCI 3104, Algorithms **Escobedo & Jahagirdar**
Homework 6 (100 points) **Summer 2020, CU-Boulder**

Extra Credit Question 1 (5 pts) *For this extra credit question, please refer the leetcode link provided below or click here. Multiple solutions exist to this question ranging from brute force to the most optimal one. Points will be provided based on Time and Space Complexities relative to that of the most optimal solution.*

Please provide your solution with proper comments which carries points as well.

https://leetcode.com/problems/cheapest-flights-within-k-stops/

```
Replace this text with your source code inside of the .tex document
```

Extra Credit Question 2 (5 pts) *For this extra credit question, please refer the leetcode link provided below or click here. Multiple solutions exist to this question ranging from brute force to the most optimal one. Points will be provided based on Time and Space Complexities relative to that of the most optimal solution.*

Please provide your solution with proper comments which carries points as well.

https://leetcode.com/problems/pacific-atlantic-water-flow/

```
Replace this text with your source code inside of the .tex document
```