Name: Luna McBride
ID: 107607144

CSCI 3104, Algorithms **Profs. Hoenigman & Agrawal**
Problem Set 6b (40 points) **Fall 2019, CU-Boulder**

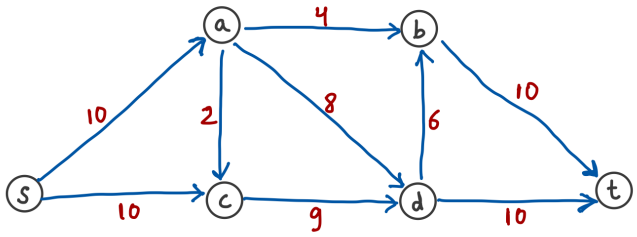
Instructions for submitting your solution:

- The solutions **should be typed** and we cannot accept hand-written solutions. Here's a short intro to Latex.
- 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 Gradescope 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.
- Verbal reasoning is typically insufficient for full credit. Instead, write a logical argument, in the style of a mathematical proof.
- 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.
- You may work with other students. However, **all solutions must be written independently and in your own words.** Referencing solutions of any sort is strictly prohibited. You must explicitly cite any sources, as well as any collaborators.

Name: Luna McBride
ID: 107607144

CSCI 3104, Algorithms **Profs. Hoenigman & Agrawal**
Problem Set 6b (40 points) **Fall 2019, CU-Boulder**

1. (19 pts) Based on the following network and the given edge capacities answer the following.



   (a) (12 pts) Suppose we start the Ford-Fulkerson algorithm and **select the path** $s- > a- > c- > d- > t$ **in the first iteration (Do not chose the first s-t 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

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

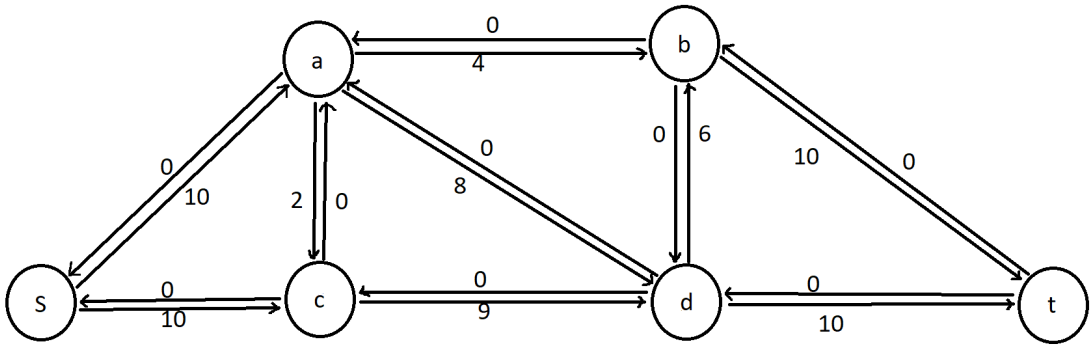
   Also, report the Max-Flow after the algorithm terminates.

   Solution.

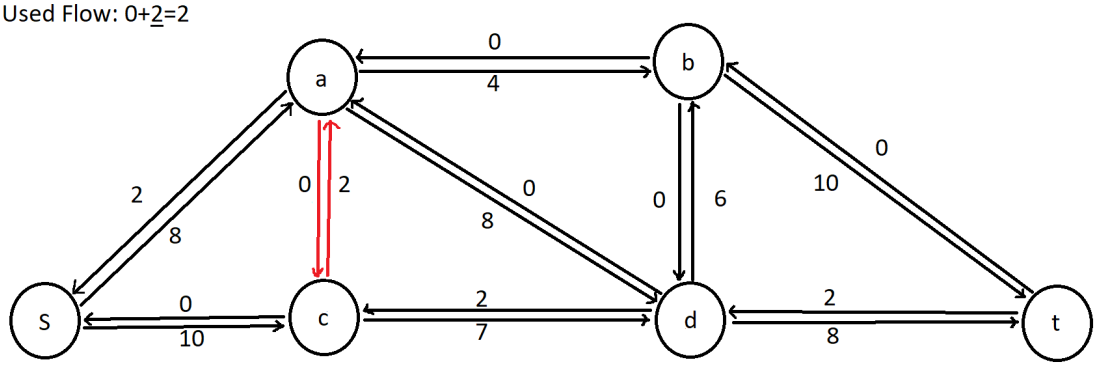
Name: Luna McBride
ID: 107607144

CSCI 3104, Algorithms **Profs. Hoenigman & Agrawal**
Problem Set 6b (40 points) **Fall 2019, CU-Boulder**

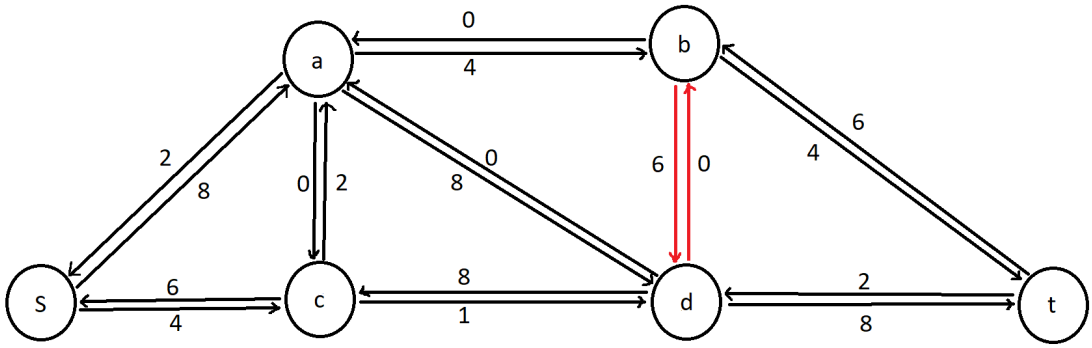
Initial (no Ford-Fulkenson)



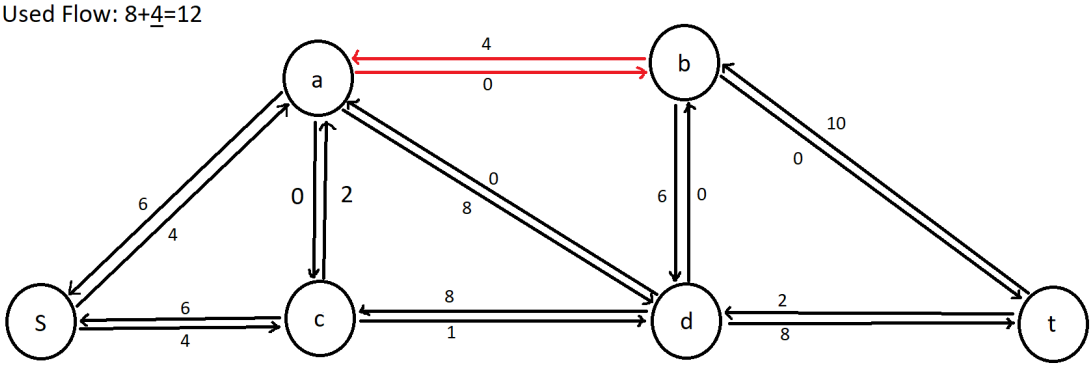
Used: SACDT
Bottleneck: AC (2)
Used Flow: 0+2=2



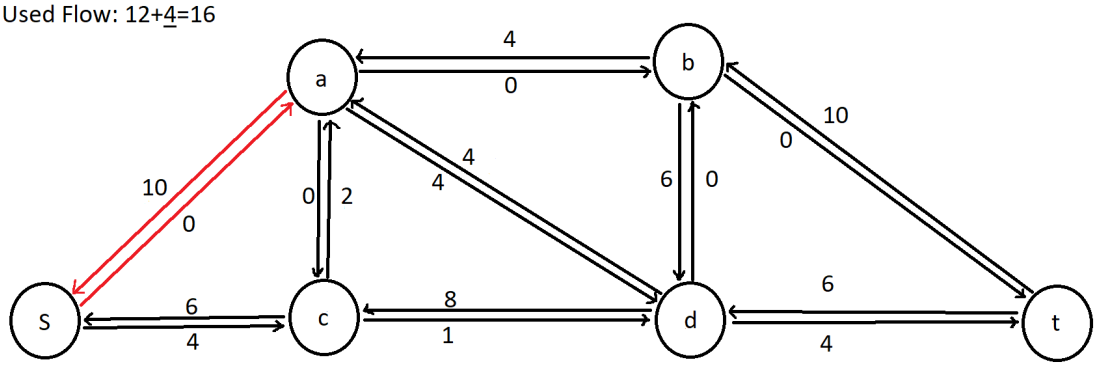
Used: SCDBT
Bottleneck: BD (6)
Used Flow: 2+6=8



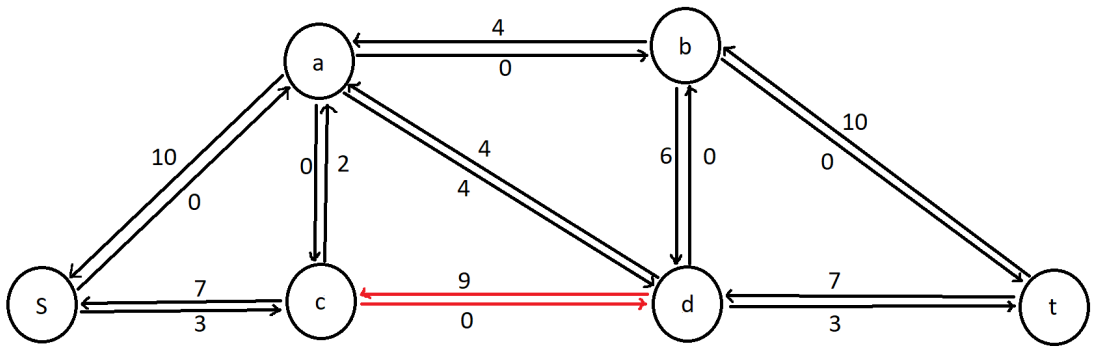
Used: SABT
Bottleneck: AB (4)
Used Flow: 8+4=12



Used: SADT
Bottleneck: SA (4)
Used Flow: 12+4=16



Used: SCDT
Bottleneck: CD (1)
Used Flow: 16+1=17

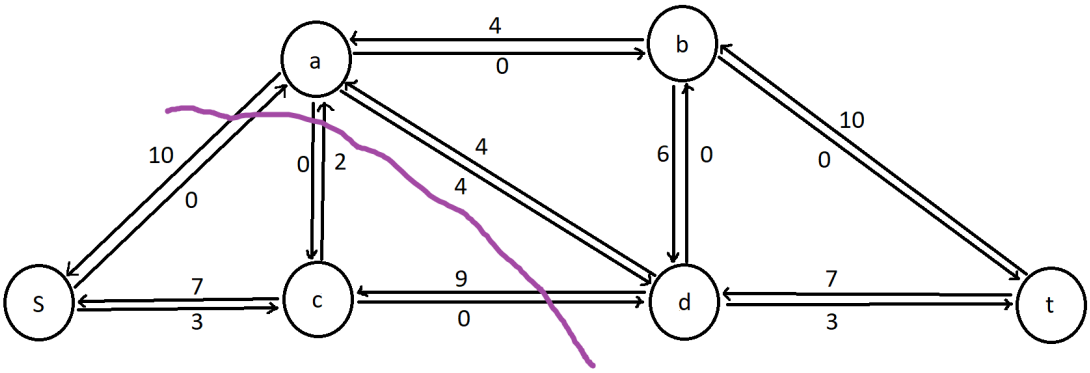


Name: Luna McBride
ID: 107607144

CSCI 3104, Algorithms **Profs. Hoenigman & Agrawal**
Problem Set 6b (40 points) **Fall 2019, CU-Boulder**

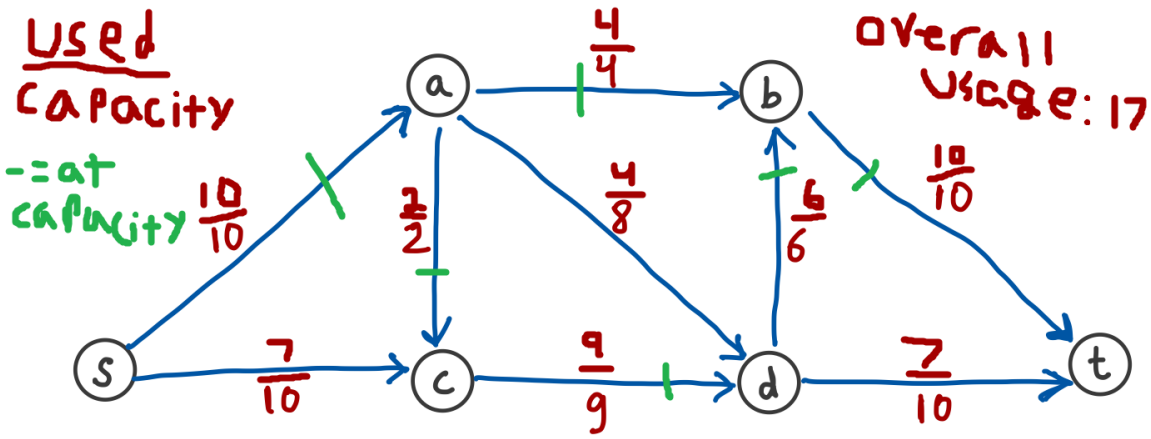
Final: 17

Cuts: {S,C}{A,B,D,T}



(b) (3 pts) Show the final flow f(e) for the edges of the original graph when the Ford-Fulkerson algorithm terminates.

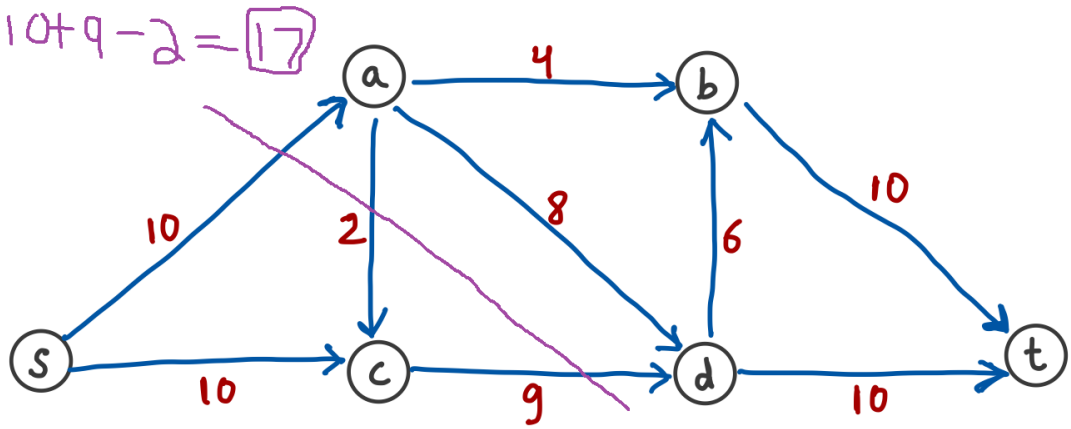
Solution.



Structure: The graph uses a used flow over capacity model to show used flow. There is a green mark to show the edges that got to capacity, showing overall usage in the top right corner.

(c) (4 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, report the crossing edges in this cut that are saturated (can't carry any more flow).

Solution.



Saturated Edges in the cut: SA, AC, CD

This minimum is equal to my max flow, creating the exact same cut. After filling the top source edge and the AC connection (occuring in the cut given at the very beginning), all you could do to leave was CD, which filled very quickly and left no way out after 17 flow, just as predicted by the minimum cut, following the Max Flow Min Cut Theorum.

Name: Luna McBride
ID: 107607144

2. (10 pts) Let (X, Y) be any s-t cut in the network G and a be any flow.

   (a) (5 pts) Prove that the value of the flow a equals the **net** flow that crosses the cut (X, Y).
   i.e. $value(a) = \sum_{\text{e out of X}} a(e) - \sum_{\text{e in to X}} a(e)$
   You should use the flow conservation property to complete the proof.
   Hint : Recollect the definition of a flow a.
   $value(a) = \sum_{\text{e out of s}} a(e) - \sum_{\text{e in to s}} a(e)$ where s is the source.
   Solution.
   Base Case: $value(a) = \sum_{\text{e out of s}} a(e) - \sum_{\text{e in to s}} a(e)$. This is the definition of flow and goes directly from the source.

   Inductive Hypothesis: $value(a) = \sum_{\text{e out of k}} a(e) - \sum_{\text{e in to k}} a(e)$ for some group of vertices within our flow graph

   Inductive step: Say we have a k+1 that adds a vertex to the previous k. The Flow Conservation Property states $\sum_{\text{e in to k+1}} a(e) = \sum_{\text{e out of k+1}} a(e)$, with k+1 being our vertex. Move one to the other side to get $0= \sum_{\text{e out of k+1}} a(e) - \sum_{\text{e in to k+1}} a(e)$, as what goes in must come out. However, as we are trying to count what is coming through, we will ignore the ins previously counted as out as the flow directly from this line in the source. Flow from a different line, however, needs to be accounted for, as that adds to what is coming out and we are looking at flow coming from this line of the source. Due to this and ignoring the main in, this shows the value of which we are flowing, and as such, $value(a) = \sum_{\text{e out of k+1}} a(e) - \sum_{\text{e in to k+1}} a(e)$

(b) (5 pts) Use the above proof (from part Q3a) to prove that the value of the flow a $\leq$ Capacity of the cut (X, Y).

Solution.

There are two cases that can happen between cuts:

Case 1: There is no inward flow. All flow is outward flow. This means the amount of flow is either the capacity or limited by some bottleneck edge further behind this point.

Case 2: There is at least one inward flow from Y. This is inserted as a negative value, lowering the flow and making it less than the capacity.

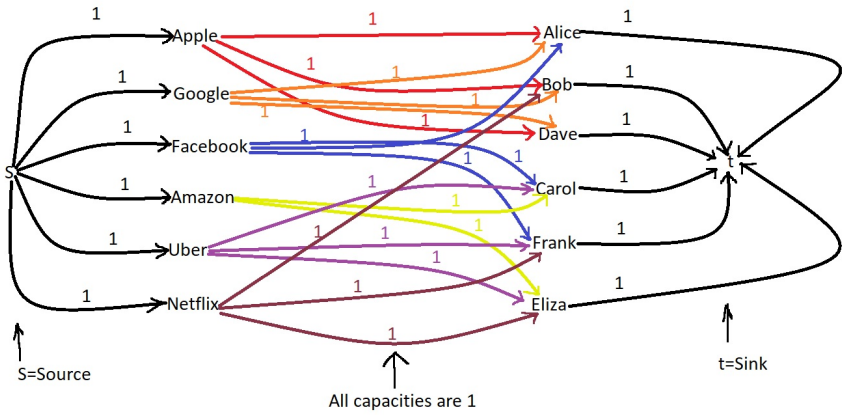
It is also fair to note that if any of these cases, we cannot reach above capacity in general following the capacity rule of flow graphs. This can always be further lowered by bottlenecks elsewhere at any point.

3. (11 pts) CU is organising a spot job fair were many companies participate and take tests to select students. After their final round of interviews, they all find a preference list of candidates that they would like to hire. All the companies just want to hire one student (because recession). All the companies sat together and they realised that if they extend offers to the same students, only one of them would get the student so they decide to run an algorithm to hire the maximum number of students they can together.
Example - Following is one such preference of each company after the final round of interviews. If they all give the offer to just their first preference only 3 students will get hired. But a better offer is Apple - Alice, Google - Dave, Facebook - Carol, Amazon - Eliza, Uber - Frank, Netflix - Bob and this gets 6 students hired.

   Help them come up with an algorithm to find an offer set that gets the maximum students the job using Ford-Fulkerson.

Preference	Apple	Google	Facebook	Amazon	Uber	Netflix
1	Alice	Alice	Alice	Carol	Carol	Bob
2	Bob	Bob	Carol	Eliza	Eliza	Eliza
3	Dave	Dave	Frank		Frank	Frank

   (a) (6 pts) Draw a network G to represent this problem as a flow maximisation problem for the example given above. Clearly indicate the source, the edge directions, the sink and the capacities and label the vertices.
   Solution.



(b) (5 pts) Assume that you have access to Fork-Fulkerson sub-routine called **Ford-Fulkerson(G)** that takes a network and gives out max-flow in terms of f(e) for all the edges. How will you use this sub-routine to find the offer set that employs the maximum number of students. Clearly explain your solution.

(Space to solve Q3)

Solution.

Ford-Fulkerson, in essence, finds the max flow within a graph. That is to say, in a typical graph, it finds the paths in a graph to maximize the flow moving through each vertex. The cool thing about this graph is the capacity of each edge is 1. When push comes to shove, the source gives out 6 and the sink brings in 6, which means all 6 companies will end up linked with one of the students once all is said and done. This can be said definitively because even if there is a solution with not every company getting a student, this will be negated by the algorithm, as Ford-Fulkerson looks for max flow, and not using all students is not max flow and a reverse flow for better is an option.

As for the details, the algorithm, for each employer, will find a path from source to sink, no matter the student it flows through. If later down the road all students are blocked for a specific employer, it can reverse the flow and replace it. So say if we already blocked Dave out of Google and Apple, since those are the only ones associated with Dave, the algorithm will reroute to get Dave a job at one of them and put the one at one of those companies before to another one of their options. This will continue until there are six content employers with six students excited to get into the big important companies.