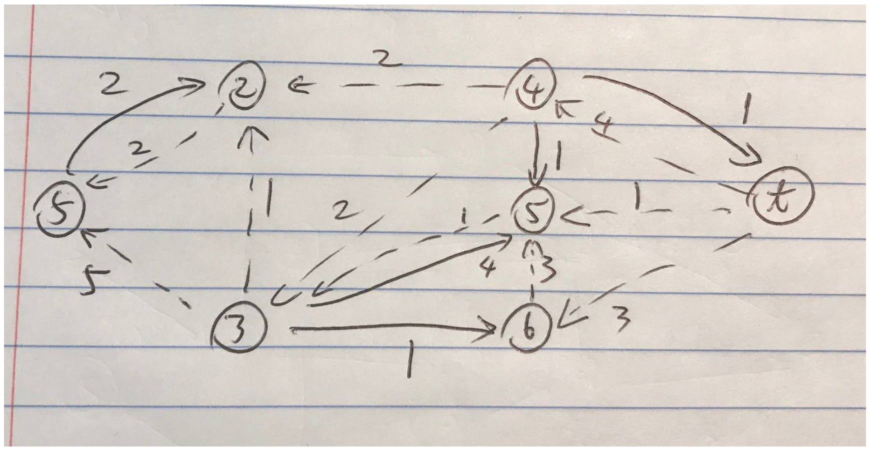
Sicheng Chu schu37@wisc.edu
Hiroshi Shu hshu5@wisc.edu
Lec 001
Hw 8

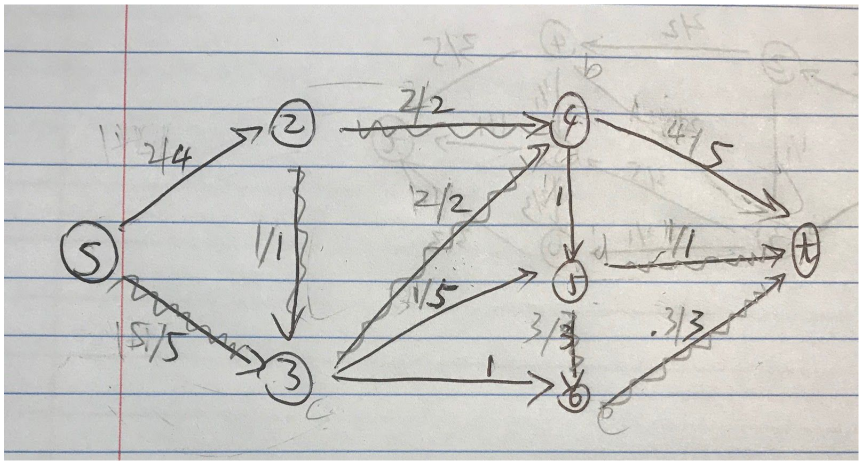
1. FIrstly. Find a minimum cut by finding a max flow using Ford Fulkerson Algorithm.
So we get the value of the min cut capacity. Suppose we have G with m vertices including s and t. Now we find all cuts that can't be more than m. Among all these cuts those cuts have the same capacity as the min cut are also min cuts. For all min cuts, return the one with the least edges in it.
Correctness: Ford Fulkerson correctly return us the max flow. Based on the min cut max flow theorem, min cut separates the graph in to two parts where one part contains all vertices reachable from s in residual graph and the rest are in the second part when the graph is at its max flow. The rest of the algorithm is just to compare every single cuts to the existing min cut and by common sense all cuts have the same capacity as the min cut’s capacity are also all min cuts. Among all min cuts, return the one with the least edge as the question desired hence solve the problem.
Runtime: Ford Fulkerson Algorithm has runtime O(mF). Iterating all vertices take O(m) and for each one we need to exam all edges in that cut which takes total of O(mn). So the total runtime is O(mn).

Sicheng Chu schu37@wisc.edu
Hiroshi Shu hshu5@wisc.edu
Lec 001
Hw 8

2.(a) Max flow of G is 8
$$G_{f*}$$ is shown below



The cut that cuts edges {(s,2).(2,3),(2,4)} has min capacity equals to 8
(b) After finding the max-flow, there are some edges are saturated and some are not saturated. The saturated edges places a upper restriction on the flow that could pass and unsaturated edges places a lower restriction on flow that could pass.



Therefore, to increase upper bounding by one unit by modify one edge, the edge needs to be the only saturated edge in a path from s to t. The edge set is $${(2,4)}$$. Similarly, for an edge to be lower-bounding, the edge would be the only edge need to be modified in a path from s to t. Therefore, the edge is the saturated. The edge set is $${(s,3),(2,3),(2,4),(3,4),(5,6),(6,t),(5,t)}$$. When an edge reduced to 0, this means the edge do not exist. In general, the bottleneck edge of a path determines the increasing or decreasing amount of final max flow.
(3) ***Algorithm:***
Take the original max flow $$f*$$ and compute $$Gf*$$, *//O(m+n)*
Convert $$Gf*$$ to $Gf$ *//O(m+n)*
A[m] *//Store possible upper bounding edges*
DFS search for paths contains only one saturated edge i and store in A[i]; *//O(m+n)*
Return A[m]

Sicheng Chu schu37@wisc.edu
Hiroshi Shu hshu5@wisc.edu
Lec 001
Hw 8

***Analysis:***
Since $$G_{f*}$$ is the residual plot, the constructing would take *O(m+n).* By summing edges, $$G_f$$ takes *O(m+n)*. DFS takes *O(m+n)* to traverse and find edge (store value takes *O(1)* in DFS)
**Correctness:** Since the algorithm would return array that contains saturated edges, which is the only saturated edges from a path s to t in the final residual network by DFS, the algorithm would return the expected edges in (2).