CSCI 570 HOMEWORK 8

Spring 2024

Q1. Determine if the following statements are true or false. If true, give a brief explanation. If false give a counterexample. (10 points)

(a) For a flow network, there always exists a maximum flow that doesn't include a cycle containing positive flow.

(b) If you have non-integer edge capacities, then you cannot have an integer max-flow value.

(c) Suppose the maximum s-t flow of a graph has value f. Now we increase the capacity of every edge by 1. Then the maximum s-t flow in this modified graph will have a value of at most f + 1.

(d) If all edge capacities are multiplied by a positive number k , then the min-cut remains unchanged.

Solution:

(a) True, we can always remove the flow from such a cycle and still get the same flow value.

(b) False, consider a graph with source s, sink t and two nodes a and b. Let's say there is an edge from s to both a and b with capacity 0.5 and from both a and b to t with capacity 0.5, then the max-flow value for this graph is 1, which is an integer.

(c) False. Counter-Example, consider a graph with source s, sink t and two nodes a and b. Let's say there is an edge from s to both a and b with capacity 3 and from both a and b to t with capacity 3 the maximum flow has value f= 3 + 3 = 6.Increasing the capacity of every edge by 1 causes the maximum flow in the modified graph to have value 4 + 4 = 8.

(d) True. The value of every cut gets multiplied by k, thus the relative-order of min-cut remains the same.

Rubric (10 pts)

For each question:

- 1 pt: Correct choice
- 1.5 pt: Valid reasoning/counter example for the answer

Q2. A tourist group needs to convert all of their USD into various international currencies.There are n tourists t1,t2, ...,tn and m currencies c1,c2, ...,cm. Each tourist tk has Fk Dollars to convert. For each currency cj , the bank can convert at most Bj Dollars to cj.Tourist tk is willing to trade at most Skj of their Dollars for currency cj. (For example, a tourist with 1000 dollars might be willing to convert up to 300 of their USD for Rupees, up to 500 of their USD for Japanese Yen, and up to 400 of their USD for Euros). Assume that all tourists give their requests to the bank at the same time. Design an algorithm that the bank can use to determine whether all requests can be satisfied. To do this, construct and draw a network flow graph, with appropriate source and sink nodes, and edge capacities.
Prove your algorithm is correct by making an if-and-only-if claim (10 points)

Solution: Network Flow Construction and Algorithm:

We will solve the problem by first constructing a network flow graph and then running Ford-Fulkerson or any other Network Flow Algorithm to get the max flow value.

Construction:

- Insert two new vertices, source node S and sink node T.
- Connect all the tourists tk with source node S assigning edge weight equal to Fk. Here, Fk stands for the maximum USD tourist tk can exchange.
- Then, connect all tourists tk to all the currencies available i.e., cj with each edge having weight Skj which is the tk tourist's limit to exchange their USD for a particular currency cj.
- Connect currencies cj with sink node T, with edge weight Bj, which is the maximum limit of that particular currency cj that the bank can convert from USD.
  In the graph constructed this way we can run Ford-Fulkerson or any Network Flow Algorithm from source S to sink T, and if the max-flow value is equal to Pk Fk then we know all requests can be satisfied. In this case the flow on each edge (tk,cj) represents the amount that tourist tk exchanges into currency cj.

The problem has a solution (i.e., all the tourists are able to exchange their specified USD while following all the constraints), if and only if the max-flow value on the constructed graph is Pk Fk.

Rubric

• 8 pts: Correct Construction of Flow Network
(-2 pts: For each incorrect edge weight/node.)

(source and sink can be reversed)

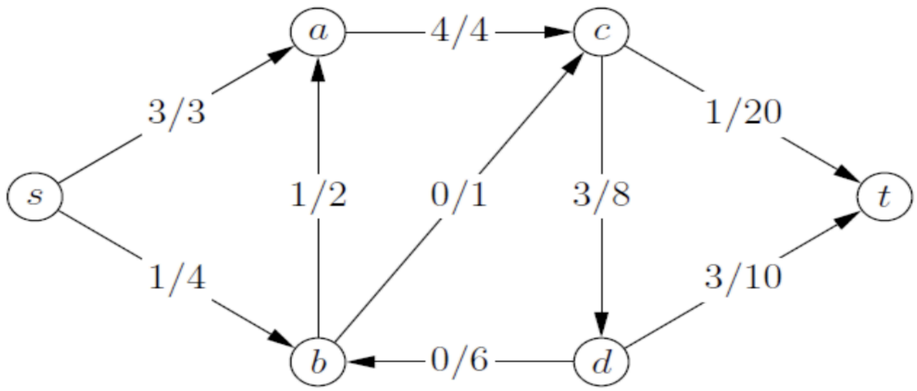
2 pts: Correct claim

Q3. You are given a collection of n points U={u1,...,un} in the plane, each of which is the location of a cell-phone user. You are also given the locations of m cell-phone towers,C={c1,...,cm}. A cell-phone user can connect to a tower if it is within distance ∆ of the tower. For the sake of fault-tolerance each cell-phone user must be connected to at least three different towers. For each tower ci you are given the maximum number of users, mi that can connect to this tower. Give a polynomial time algorithm, which determines whether it is possible to assign all the cell-phone users to towers, subject to these constraints. Prove your algorithm is correct by making an if-and-only-if claim. (You may assume you have a function that returns the distance between any two points in O(1) time.) (10 points)

Solution:
We will do this by reduction to network flow. (Create an s-t network G=(V,E) where .
Create a unit capacity edge (ui,cj) if cell-phone user is within distance of cell
phone tower . Create an edge from s to each user with capacity 3, and create an
edge from tower to t with capacity mj.
Compute the maximum (integer) flow in this network. We claim that a feasible schedule exists if and only if the max flow is 3n.

Rubric:
Design correct Network flow: 8 pts.
Correct claim: 2 pts.

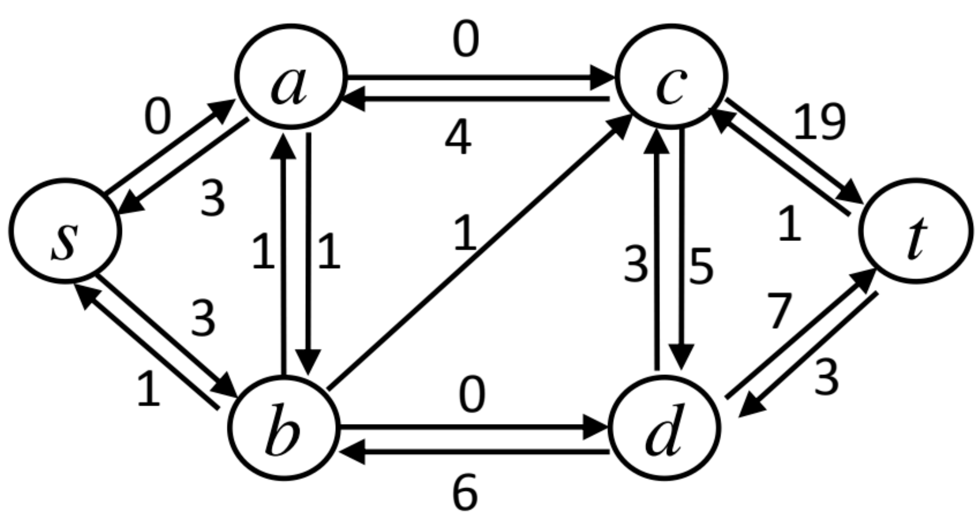
Q4. You are given a directed graph which after a few iterations of Ford-Fulkerson has the following flow. The labeling of edges indicate flow/capacity: (15 points)



a) Draw the corresponding residual graph.
b) Is this a max flow? If yes, indicate why. If no, find max flow.
c) What is the min-cut?

Solution:

a)



b) No, since there is still path like s-b-c-t that can be augmented with flow 1.The max flow is 4+1=5 that can be obtained from the residual graph shown below.
c) Min-cut: ({s,a,b},{c,d,t}) .The value of min-cut equals to 4+1=5.

Rubrics:

a) 6 pts - correct residual graph (-0.5 for each incorrect edge)
b) 3 pts - for stating that given graph is not max flow and there exist a s-t path

3 pts - for correctly finding the max flow

c) 3 pts - for correctly finding the min cut

Q5. USC has resumed in-person education after a one-year break, with k on-site courses available this term, labeled c_1 through c_k. Additionally, there are n students, labeled s_1 to s_n, attending these k courses. It's possible for a student to attend multiple on-site courses, and each course will have a variety of students enrolled. (20 points)

(a) Each student s_j wishes to enroll in a specific group p_j of the k available courses, with the condition that each must enroll in at least m courses to qualify as a full-time student (where pj is greater than or equal to m). Furthermore, every course c_i can only accommodate a maximum of q_i students. The task for the school's administration is to verify whether every student can register as a full-time student under these conditions. Propose an algorithm to assess this scenario. Prove your algorithm is correct by making an if-and-only-if claim.

(b) Assuming a viable solution is found for part (a) where each student is enrolled in exactly m courses, there arises a need for a student representative for each course from among the enrolled students. However, any single student can represent at most r (where r is less than m) courses in which they are enrolled. Develop an algorithm to check whether it is possible to appoint such representatives, building on the solution from part (a) as a foundation. Prove your algorithm is correct by making an if-and-only-if claim.

Solution: Part (a)
We can solve the problem by constructing a network flow graph and then running Ford–Fulkerson algorithm to get the max flow. If the max flow is equal to nm, then all students can be enrolled as full time students. The setup of the graph is described below.
1. Place the n students and k classes as vertices in a bipartite style graph. Connect each student sj with all the classes in pj using edges with capacity of 1.
2. Place a sink vertex which is connected to all the classes (or alternatively students). Also, place a source vertex which is connected to all the students (or alternatively classes).
3. Connect all sj student vertices to the source (or sink) vertex with capacity of m.

4. Connect all ci class vertices to the sink (or source) vertex with capacity of qi

The enrollment is feasible if and only if there exists a max flow of nm.

Solution: Part (b)
We can solve the problem by starting from the solution from part (a). We will modify the constructed network flow graph slightly and then will run Ford–Fulkerson algorithm to get the max flow. If the max flow is equal to the number of classes k, then a selection exists otherwise no such selection exists. The setup of the graph is described below.
1. Use the same nodes as constructed in Part (a) of the solution.
2. The course selections (edges between students and courses) for each student will be reduced to what is available in the solution from part
(a) (our feasible solution). i.e. we will remove some edges between students and courses that they did not get to sign up for.
3. The capacity of student to class edges will remain the same (i.e.1).
4. Assign 1 as the capacity from classes to sink (or alternatively source) vertex. (As there can only be 1 student representative from each class.)
5. Assign r capacity for edges between source (or alternatively sink) and student vertices.

The selection is feasible if and only if there exists a max flow of k.

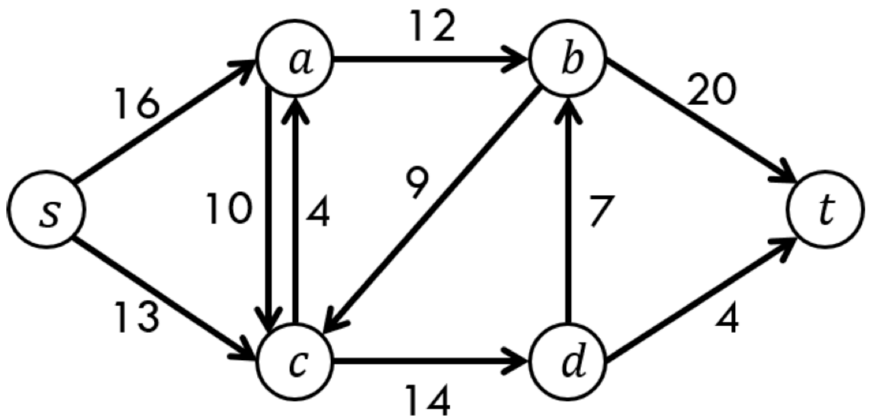
Rubric Part A (10 pts)

- 8 pts: Correct construction of the network flow graph.
- −2 pts: For each incorrect edge weight.
- 2 pts: Correct claim

Rubric Part B (10 pts)

- 8 pts: Correct construction of the network flow graph.
- −2 pts: For each incorrect edge weight.
- 2 pts: Correct claim

Q6. Given the below graph solve the below questions using scaled version of Ford-Fulkerson. (15 points)



a) Give the Δ and path selected at each iteration.
b) Draw the final network graph and the residual graph
c) Find the maxflow and mincut

Solution:

a) For Δ =16 no paths. afterwards

Δ =8

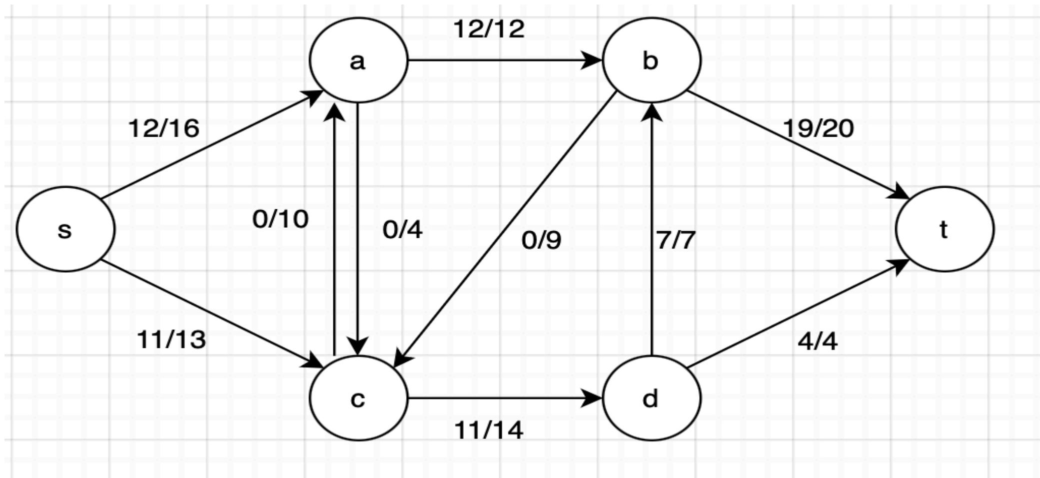
We will choose path s-a-b-t with flow 12 units

Δ =4

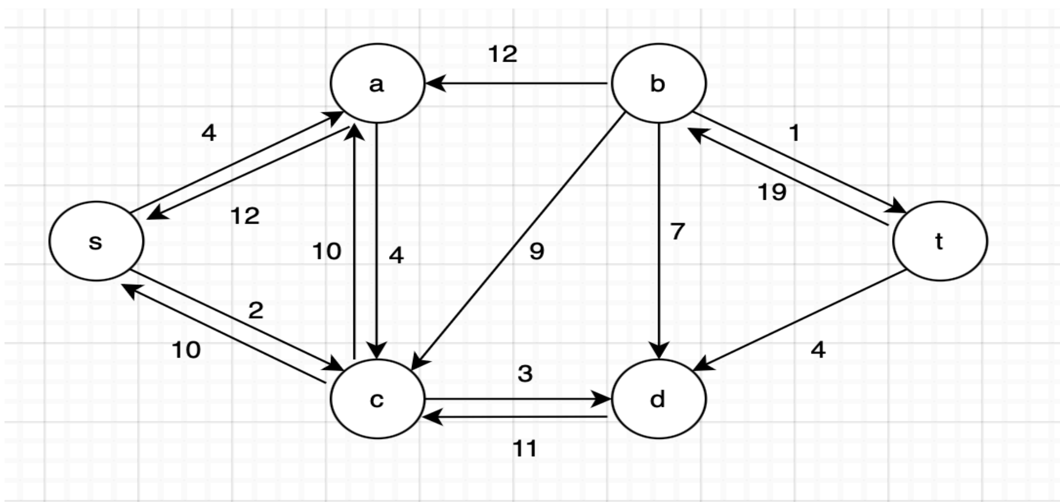
We will choose path s-c-d-b-t with flow 7 units

We will choose path s-c-d-t with flow 4 units

b) Final Network graph :



Final Residual Graph :

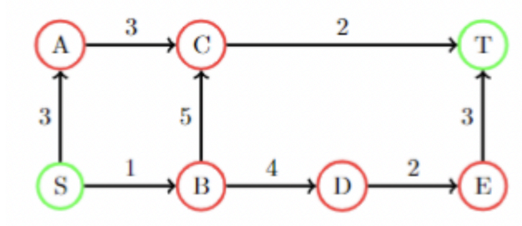


c) Max Flow will be 23 units and Min cut will be ({s,a,c,d},{b,t})

Rubrics:

- 6 pts - correct iterations of scaled ford fulkerson
- 3 pts - correct final network graph
- 3 pts -correct final residual graph
- 3 pts -correct maxflow and min cut

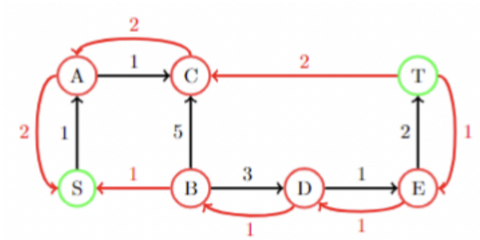
Q7:The following graph G has labeled nodes and edges between them. Each edge is labeled with its capacity. (10 points)



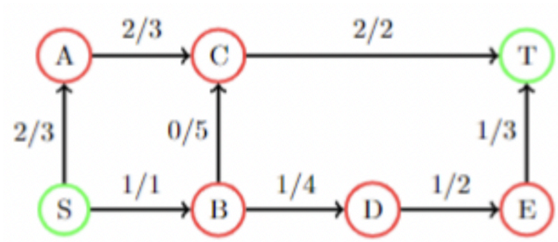
(a) Draw the final residual graph G_f using the Ford-Fulkerson algorithm corresponding to the max flow. Please do NOT show all intermediate steps.
(b) What is the max-flow value?
(c) What is the min-cut?

Solution:

(a) Final Residual Graph



Final Flow graph (not required to be drawn)



(b) The max flow is 2 + 1 = 3.

(c) The min cut is { S, A, C } and { B, D, E, T }. This is correct because all the edges to and from the min cut either are saturated CT, SB) or have zero flow (BC).

Rubric (10 pts)

•5 pts: Correct Final Residual graph. (−2 pts: Incorrect edge capacity.)
•2 pts: Correct max flow value.
•3 pts: Correct min cut sets.

Q8. You are provided with a flow network where each edge has a capacity of one. This network is represented by a directed graph G = (V, E), including a source node s and a target node t. Additionally, you are given a positive integer k. The objective is to remove k edges to achieve the greatest possible reduction in the maximum flow from s to t in G. Your task is to identify a subset of edges F within E, where the size of F equals k, and removing these edges from G results in a new graph G' = (V, E-F) where the maximum flow from s to t is minimized. Propose a polynomial-time strategy to address this issue.

Furthermore, consider if the capacities of the edges are greater than one, and discuss whether your strategy still ensures the lowest possible maximum flow. (20 points)

Solution:
Algorithm:
• Assume the value of max-flow of given flow network G (V, E) is g. By removing k edges, the resulting max-flow can never be less than g-k when |E| ≥ k, since each edge has a capacity 1.
• According to max-flow min-cut theorem, there is an s-t cut with g edges. If g ≤ k, then remove all edges in that s-t cut, decreasing max-flow to 0 and disconnecting 's' and 't'. Else if, g > k, then remove 'k' edges from g, and create a new cut with g − k edges.
• In both the cases, whether the max-flow is 0 or g − k, the max- flow cannot be decreased any further thus giving us the maximum reduction in flow.
• The algorithm has polynomial run-time. It takes polynomial time to compute the minimal-cut and linear time in 'k' to remove 'k' edges.
• If all the edges don't have unit capacity, removing 'k' edges from min-cut in the above mentioned way does not guarantee to have the minimum possible max-flow.

Rubric
• 15 pts: Correct Algorithm proposal with polynomial time.
• 5 pts Correct answer for non-unit edges.