Homework 8

● Graded

Student

Abhishek Soundalgekar

Total Points

89 / 89 pts

Question 1

Q1

21 / 21 pts

✔ **– 0 pts** Correct

– 5 pts First residual graph

– 5 pts Second residual graph

– 5 pts Final resudual graph

– 2 pts max flow

– 2 pts first min cut

– 2 pts second mincut

– 2 pts more min cuts!

+ 1 pt extra point!

Question 2

Q2

20 / 20 pts

✔ **– 0 pts** Correct

– 4.5 pts Incorrect first graph

– 4.5 pts Incorrect second graph

– 4.5 pts Incorrect final graph

– 2 pts Incorrect max flow

– 4.5 pts Incorrect min cut

– 20 pts Incorrect

Question 3

Q3 **34** / 34 pts

3.1 **Part a** **18** / 18 pts

✔ **+ 18 pts** Full Score

+ 2 pts Construct a bipartite graph with (n) entity vertices (e_1, e_2, \ldots, e_n) and (k) group vertices (g_1, g_2, \ldots, g_k).

+ 2 pts Add an edge from (e_j) to (g_i) for each (g_i \in p_j) with capacity 1.

+ 2 pts Add a source vertex (s) and a sink vertex (t).

+ 2 pts Connect (s) to each (e_j) with capacity (m).

+ 2 pts Connect each (g_i) to (t) with capacity (q_i).

+ 2 pts Run Ford-Fulkerson from (s) to (t). If the max flow is (n \cdot m), the assignment is feasible; otherwise, it is not.

+ 3 pts Proof: Forward Direction: If feasible, then max flow = (n \cdot m)

+ 3 pts Proof: Backward Direction: If max flow = (n \cdot m), then feasible

+ 0 pts Incorrect / Not Attempted

3.2 **part b** **Resolved** **16** / 16 pts

+ 2 pts Use the same nodes as in part (a): **s, t, e_j, g_i**

+ 2 pts **s** to each **g_i**: capacity 1 (one representative per group).

+ 2 pts **g_i** to **e_j** if **e_j** is assigned to **g_i** : capacity 1.

+ 2 pts **e_j** to **t** : capacity **r** .

+ 2 pts Run Ford-Fulkerson from **s** to **t** . If the max flow is **k** , a selection exists; otherwise, it does not.

+ 3 pts Proof: Forward Direction: If feasible, then max flow = **k**

+ 3 pts Proof: Backward Direction: If max flow = **k** , then feasible

✔ **+ 10 pts** Correct construction of the network flow graph.

✔ **+ 6 pts** Correct proof in Forward and Backward Direction.

+ 2 pts Weak proof in Forward and Backward Direction.

+ 0 pts Incorrect / Miss to answer.

Regrade Request **Submitted on: Apr 03**

For Part b, I provided proofs in both the forward and backward directions. However, I did not receive any points for the backward direction proof. Could you please recheck it.

Rectified.

Reviewed on: Apr 05

Question 4

Q4

6 / 6 pts

✔ **+ 4 pts** Algorithm

✔ **+ 2 pts** Proof

+ 0 pts Incorrect

Click here to replace this description.

+ 0 pts Click here to replace this description.

Question 5

Q5

8 / 8 pts

✔ **+ 8 pts** Full score

– 2 pts Did not explicitly describing the use of the Ford-Fulkerson algorithm or any other maximum flow method

+ 2 pts Create a bipartite graph with p nodes representing men and q nodes representing women.

+ 2 pts Draw an edge with capacity one from a man to a woman if they are mutually compatible.

+ 1 pt Add a source node s and connect it to each man with an edge of capacity one.

+ 1 pt Add a sink node t and connect each woman to t with an edge of capacity one.

+ 2 pts determine the maximum flow using the Ford-Fulkerson algorithm, which is equal to the maximum number of mutually compatible dates possible.

+ 0 pts Wrong/ Missing Q5

Question assigned to the following page: 1

CSCI 570 Spring 2025

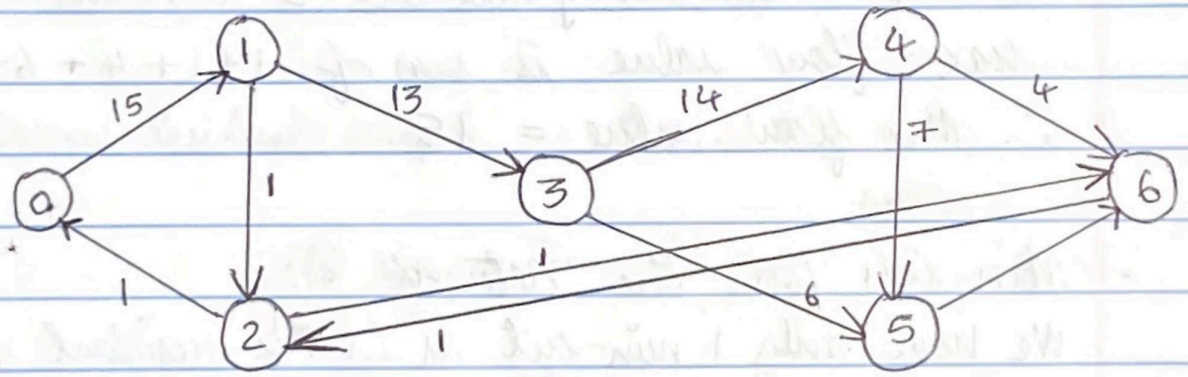
Homework 8

Name : Abhishek Soundalgekar

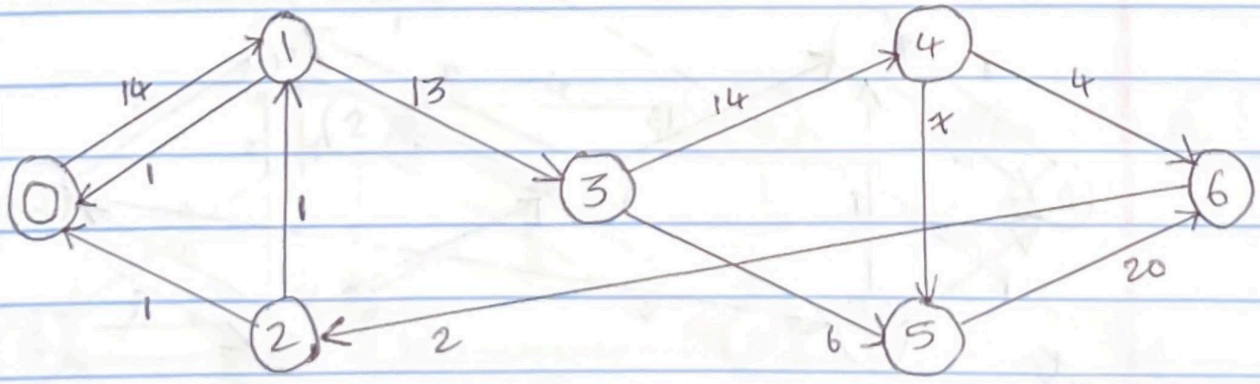
USC ID : 2089011000

Question 1
Answer 1

- First residual graph using Edmonds - Karp Algorithm
  0-2-6 is the shortest path with bottleneck = 1

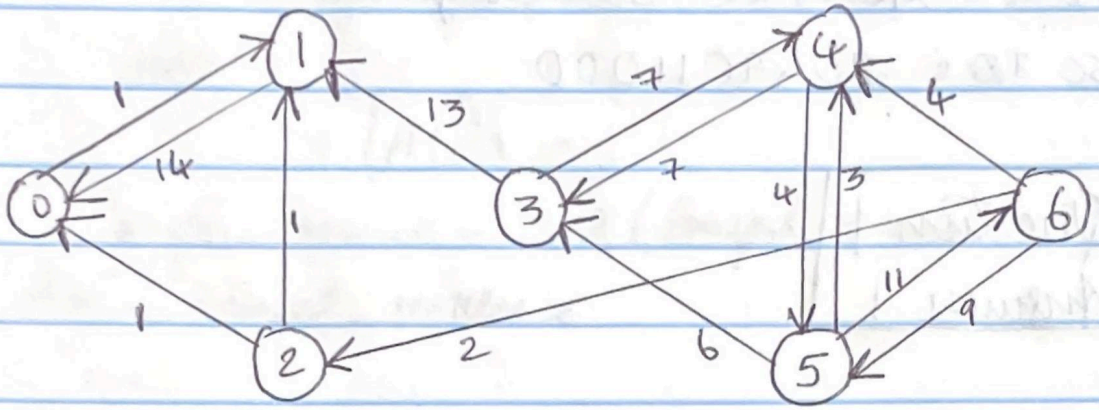


Second Residual graph :
0-1-2-6 is the shortest path with bottleneck 1



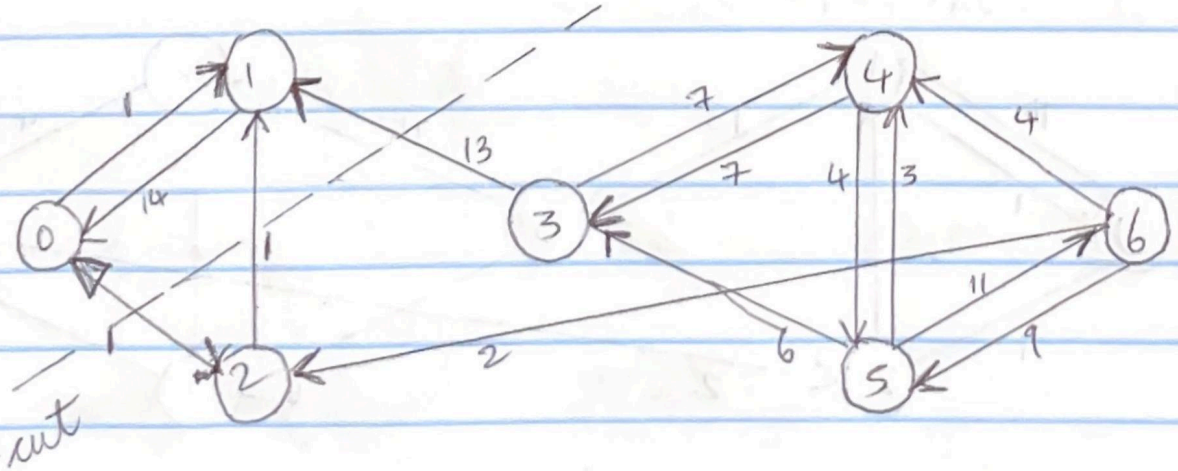
Question assigned to the following page: 1

Final residual graph (after 5 iterations):
0-1-3-4-5-6 is the path with bottleneck = 3



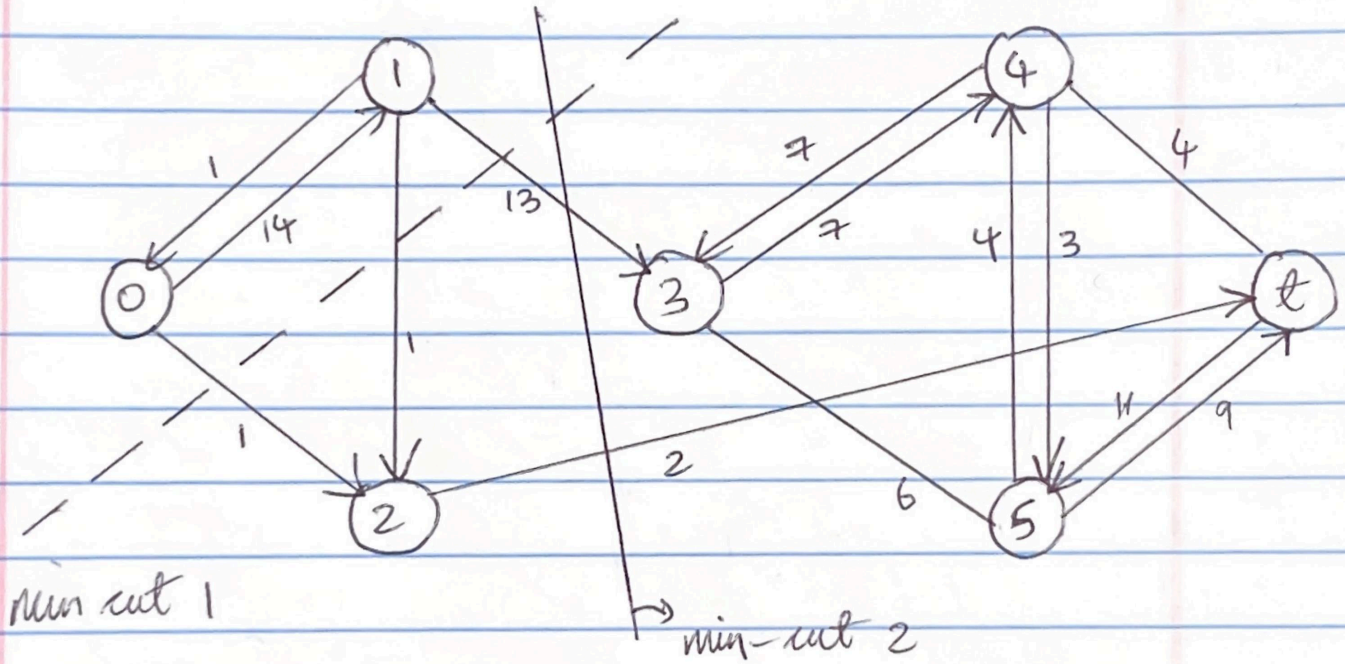
No more s-t paths exist after the final iteration. Combining all the s-t paths, the max-flow value is sum of 1+1+4+6+3
∴ Max flow value = 15

- Min cuts for the network:
We have only 1 min-cut as in the residual graph only 1 is reachable from the source node 0.
[0, 1] and [2, 3, 4, 5, 6] are the two parts.
Edges belonging to min cut: (0,2), (1,2), and (1,3)



Question assigned to the following page: 1

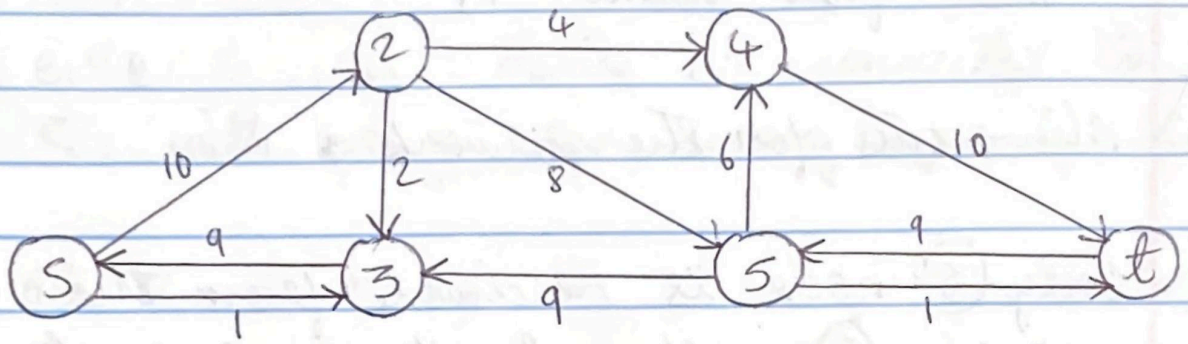
Another min cut can be observed when we reverse the edges in residual graph and check from the sink (t), till the (3) node we are reachable so we get one cut on the edges (1,3) and (2,t) separating groups [0,1,2] and [3,4,5,t]



Question assigned to the following page: 2

Question 2
Answer 2

First Residual graph $s \to 3 \to 5 \to t$ path

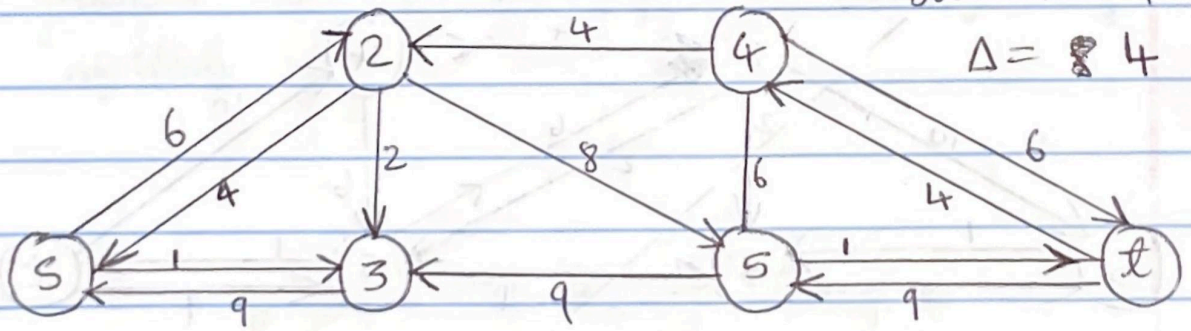


Bottleneck = 9 , $\Delta = 8$

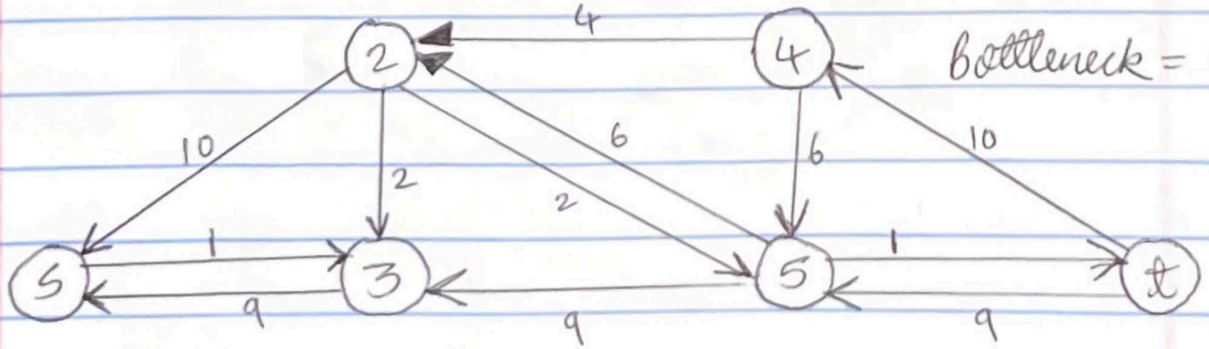
Second Residual Graph $s \to 2 \to 4 \to t$

Bottleneck = 4

$\Delta = 4$



Final Residual Graph $s \to 2 \to 5 \to 4 \to t$



Question assigned to the following page: 2

- Max flow value

The max-flow combining all the s-t paths is given as $9+4+6$

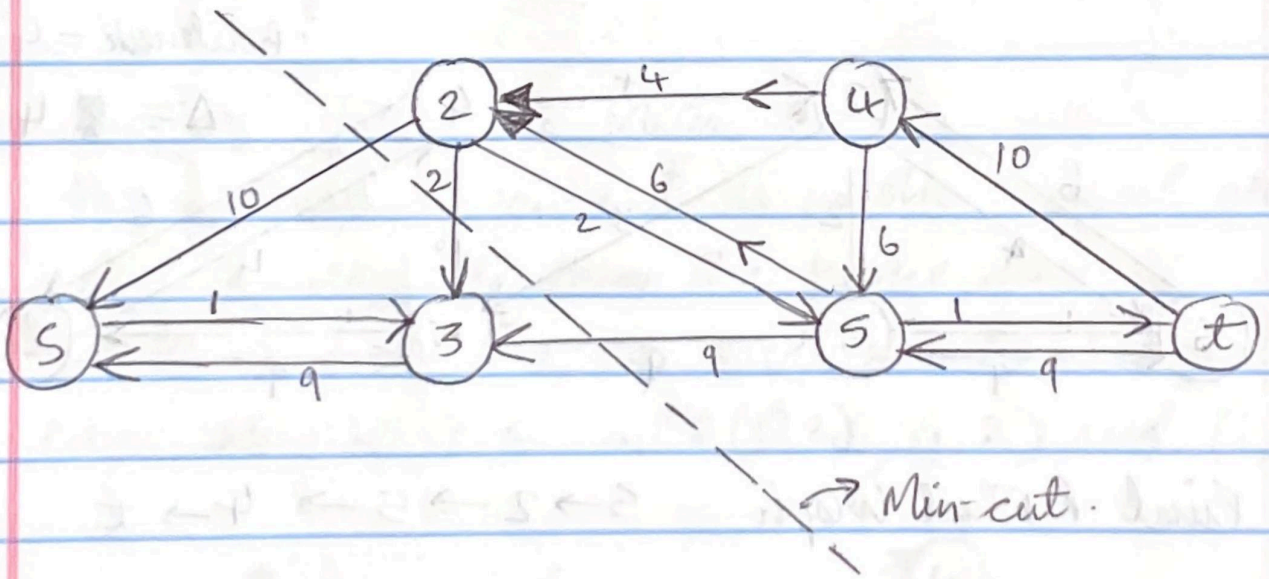
Max-flow value = 19

- Min-cuts for the network :

Only (3) node is reachable from the source (S) with all other nodes not reachable.

[S, 3] and [2, 4, 5, t] are the two parts.

(S, 2), (2, 3) and (3, 5) are the min-cut edges.



Question assigned to the following page: 3.1

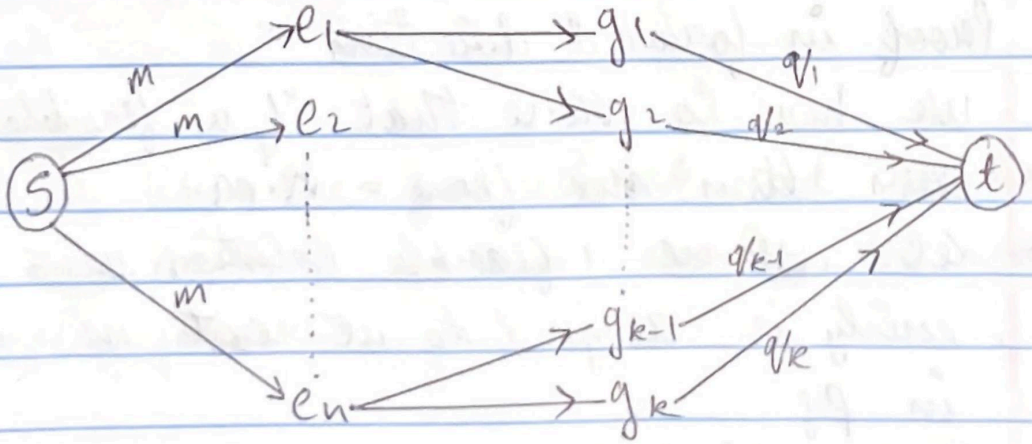
Question 3
Answer 3

Part a] Assignment Feasibility

$e_1, e_2, \ldots e_n$ entities are connected to source 's' with every edge having capacity of 'm'

All the 'k' groups $g_1, g_2 \ldots g_k$ are connected to sink 't' with every edge having capacity of q_i for group g_i where $1 \leq i \leq k$

Connecting edges from entities to their assigned groups in the set P_j for entity e_j with capacity of 1.



Question assigned to the following page: 3.1

- Algorithm

Step 1 — Constructing the flow network

Step 2 — Execute the max-flow algorithm (with the help of algorithms like Edmonds-Karp) to evaluate the max-flow value of the above network from s to t.

Step 3 — Check if total flow is $n \cdot m$, if it is equal, feasible assignment exists as each entity is assigned to atleast m groups.
Else a feasible assignment does not exist.

- Proof of correctness:

Proof in forward direction
We have to prove that if a feasible assignment exists then max flow = $n \cdot m$
→ Let's assume a feasible solution exists then each entity is assigned to at least m groups in P_j.
→ Each entity is connected to its m groups with edges of capacity 1 sending total flow of m.

Questions assigned to the following page: 3.1 and 3.2

→ Using conservation of flow, each of the k groups can receive up to q_i capacity for group g_i meaning total max flow = $q_1 + q_2 + \dots q_k$.

→ This is equal to total flow from entities who are sending m units each. ∴ for n entities max flow = $n \cdot m$, this implies that feasible assignment gives max flow of $n \cdot m$.

Proof in Backward direction:
We have to prove if max flow is $n \times m$, then feasible assignment exists.

→ Each entity sends m units of flow to the assigned groups if max flow is equal to $n \times m$.

→ Since we are not exceeding capacity of each of the k groups if max flow is calculated as $n \times m$ we get a feasible assignment
∴ The solution is correct.

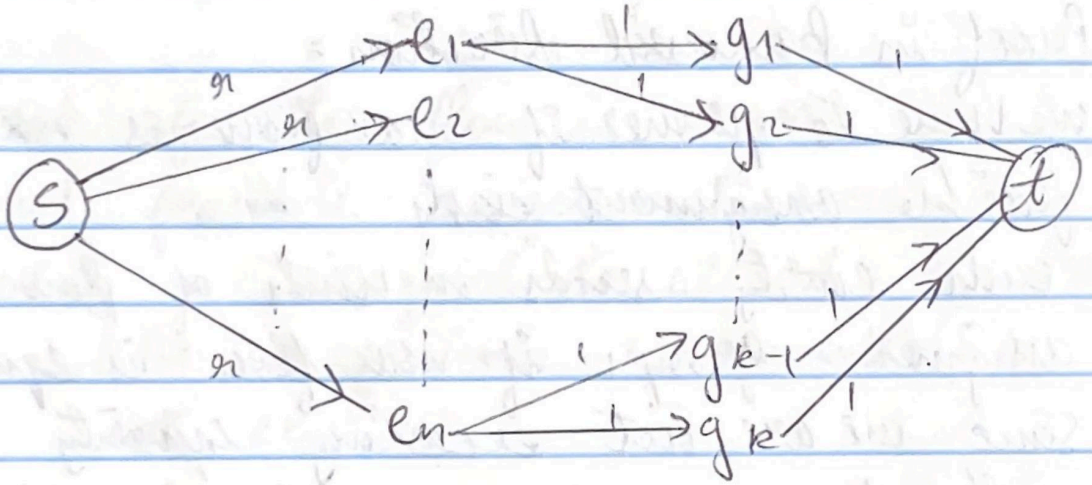
Part b] Using the flow network and solution steps in part(a), we can modify the network as follows:

Each edge from source s to entities $e_1 \dots e_n$ has max capacity 'x' meaning that they can be assigned to a maximum of x groups.

Question assigned to the following page: 3.2

Each edge from entities to groups has capacity 1 if entity is a representative entity.

Every edge from groups $g_1 - g_k$ has capacity of 1 meaning that there can only be one representative entity for a group.



Algorithm:

1. Construction of flow network as above.
2. Run max-flow algorithm to find max-flow possible on the given network (using efficient version of Ford-Fulkerson algorithm like Edmonds-Karp algo.)
3. If max-flow is k then feasible selection exists as each group has exactly one representative entity.

   Else feasible selection does not exist.

Question assigned to the following page: 3.2

- Proof of Correctness :

Proof in Forward Direction :

If feasible solution exists, max flow is k has to be proved.

→ Each entity can be selected representative for r groups meaning entities can send total flow of r to the groups.

→ Each group can receive only a max-flow of 1 as only one representative is selected meaning that total max flow is n x r from every entity.

→ Conservation of flow gives that the total group flow i.e. k meaning that feasible solution gives max flow value = k which means each of the k groups has exactly one representative entity.

Question assigned to the following page: 3.2

Proof in Backward Direction:

If max flow is k then a feasible solution exists has to be proved.

→ Every groups recieves exactly one unit of flow which means each group has exactly one representative if the max-flow is k since capacity of edges from groups to t is 1

→ Each entity sends only 0 units of flow assuming it does not exceed the value.

As we are finding max flow using algorithm then we are finding feasible solution with k representative group entities.

Therefore we prove the correctness of the algorithm.

Question assigned to the following page: 4

Question 4
Answer 4

We consider the min-cut edges to minimize the max-flow as they contribute to the value directly as bottle neck. The max s-t flow equals the min s-t flow cut. Since all edges have unit capacity we need to target those edges which can disconnect s from te leading to minimized max flow.

Algorithm:

1) Calculating the max-flow using effient algorithm (like Edmonds Karp algorithm)

2) Find min-cut with nodes having least neighbours which disconnects s from t using graph algorithms like BFS or DFS

3) Iteratively remove k edges from min-cut edges preferring those that contribute to max-flow and find max flow after each edge is removed using Edmond's-Karp- (contributes most directly).

Question assigned to the following page: 4

4.) Set $F \subseteq E$ consisting of the removed edges is returned that minimizes max flow in the network and the final max flow value.

☆ Proof of Correctness:

- Proof in Forward direction

We get minimized max-flow if valid k edges removal exists. The min-cut edges are targeted which is, we remove the bottleneck edges first ensuring we are strategically minimizing max-flow by removing k-edges.

- Proof in Backward direction

The valid k-edge is removed when max-flow is minimized. When the max-flow is minimized then using given integer k, we only remove edges directly contributing to max flow.

We remove min-cut edges as the edges that are part of the bottleneck must be in the min-cut as min cut = max-flow proving that the k-edge removal is valid in min-cuts.

Question assigned to the following page: 5

Question 5
Answer 5

→ Source S connecting to all p men with capacity 1 and sink t connected to by all q women with same capacity 1.

→ based on mutual compatibility add edges between each man in p and each woman in q; given by data D meaning edge exists between man m and women w if (m, w) is compatible in D. These edges will have capacity 1. If they are incompatible, no edge exists.

→ Each man and each women can be assigned only one date so only one edge exists for any man or women from or towards each other and no more edges can be added.

Algorithm:

1) Flow network constructed as described above.

2) Max-flow calculated using an efficient algorithm

Question assigned to the following page: 5

(like Edmonds Karp algo.) · The max flow is given by total number of pairs possible which is the number of edges between p men and q women.

3) Return the maximum flow ie. max number of compatible pairs and return the pairs of men and women between whom an edge exists making them a compatible pair.