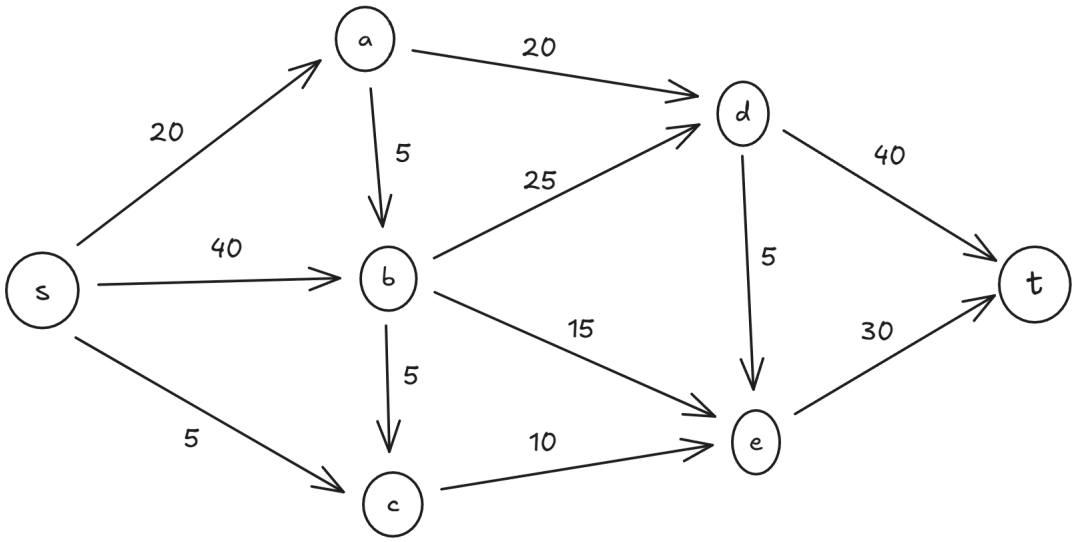
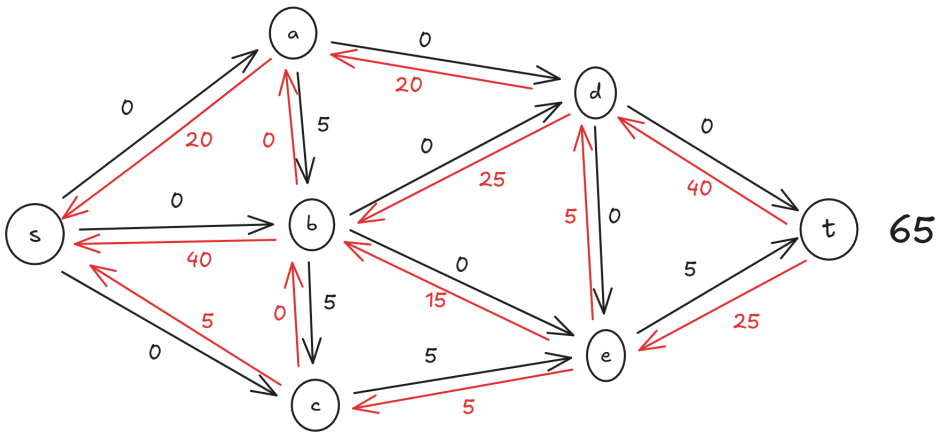
CS 630 – Fall 2024 – Lab 1
Sep 11, 2024

1. Running Ford-Fulkerson

Run the Ford-Fulkerson algorithm on the following graph. Find the max flow of this graph.



Solution: The max flow for this graph is 65. Here is one of the final residual graph.



2. Blood Donations

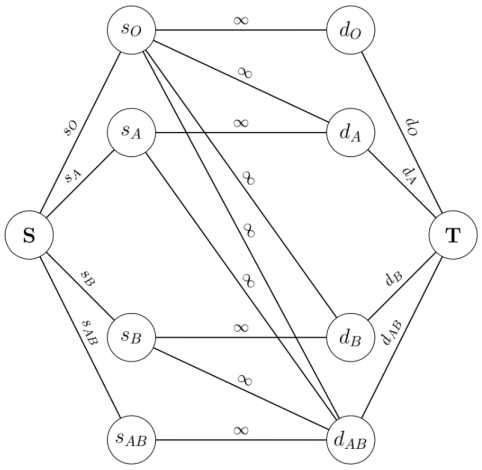
Human blood contains two antigens, A and B. Blood types O, A, B, and AB refer to the antigens that blood contains (none, A only, B only, both). An individual can receive donated blood of any type containing antigens that their own blood contains. Thus people with type O are universal donors (can donate to O, A, B, or AB), and people with type AB are universal recipients (but can only donate to others of type AB).

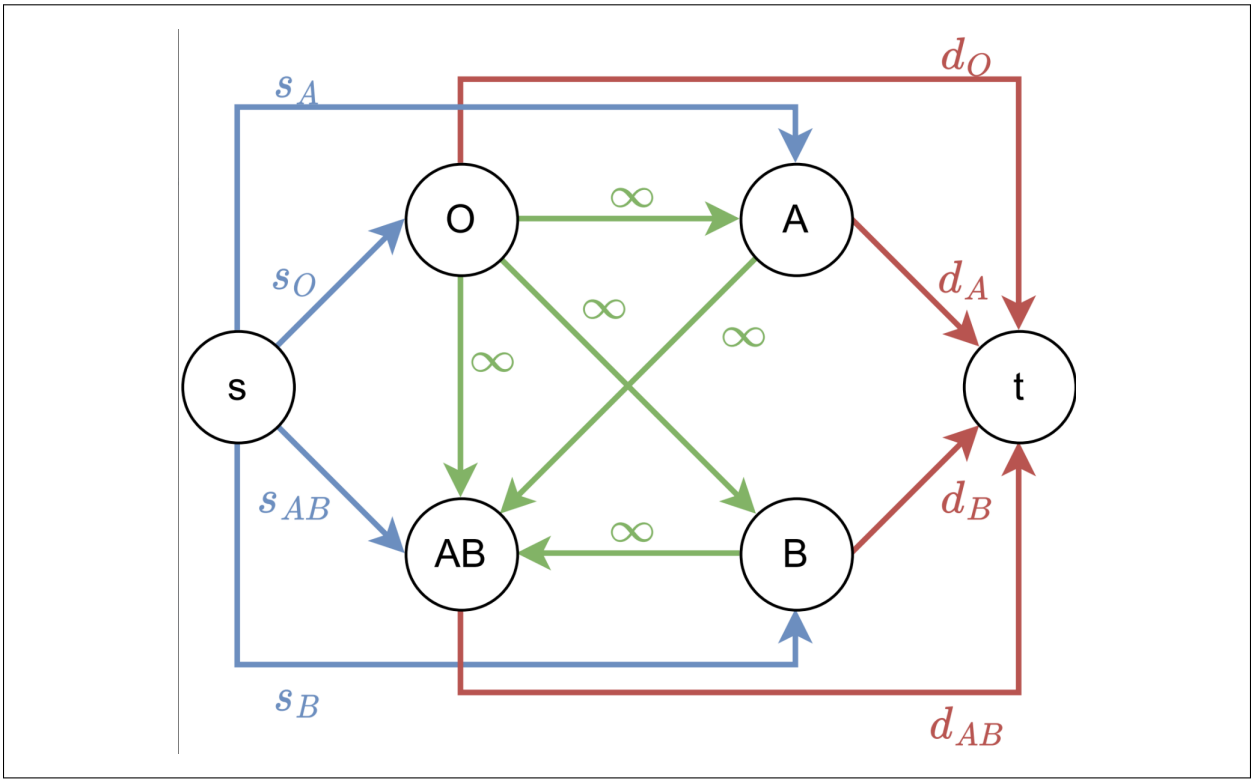
Recipient	Blood donor			
	O	A	B	AB
O	✓	x	x	x
A	✓	✓	x	x
B	✓	x	✓	x
AB	✓	✓	✓	✓

A hospital receives s_O, s_A, s_B, and s_{AB} blood donations of types O, A, B, and AB respectively and wants to determine whether it will have enough blood to meet the demand d_O, d_A, d_B, and d_{AB} of those blood types (one donation satisfies one unit of demand).

Can you create a graph where we can use Ford-Fulkerson algorithm to decide whether the blood supply can satisfy the demands?

Solution: There exists two solutions to this problem, both are valid to decide whether the supply can keep up with the demand.

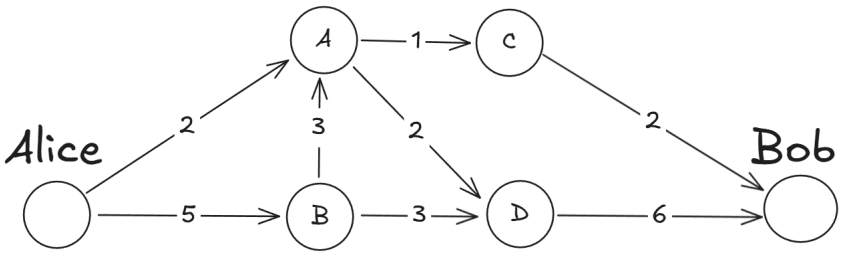




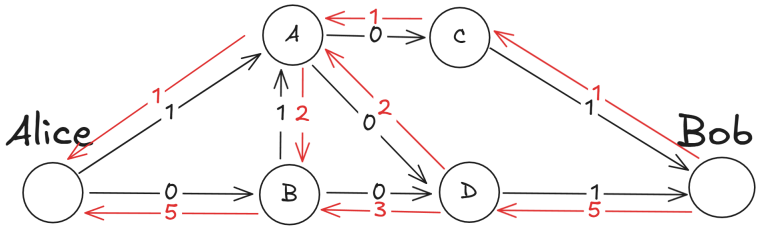
3. Sabotage Spy Network

Alice and Bob are two spies. However, because of their confidential identity, they cannot directly talk to each other. Instead, they talk to each other through their dedicated spy network (shown in the graph below). Their spy network is compose of intellectual station, which is marked as node A, B, C and D in the graph. Each directed edges in the graph shows how many line of communications exist from one node to another. (for example, there are 3 lines of communication from station B to station D)

You, as an agent from another country, wants to severed the communication between Alice and Bob. Whats the minimal number of lines of communication you need to sabotage?



Solution: We can solve this problem by running FF algorithm on the graph. We will get a max flow of 6, which means we need to sabotage 6 lines of communication to stop Alice from talking to Bob. Namely, we need to cut the edges AC, AD, BD based on our residual graph.



4. Dance Competition

'Jack and Jills' are a popular type of event at swing dance competitions. Here leaders and followers are paired up at random to dance in the competition as partners. Dancers are considered to have truly mastered the skills of leading or following if they can do it well with any partner, this competition is meant to demonstrate this skill.

This time however the assignment is not going to be entirely at random, you are in charge of assigning leaders to followers based on some rules:

- You are given the list of leaders and the list of followers. The number of leaders is denoted by L, followers by F. It's possible that some dancers end up without a partner and can't compete.
- Dancers belong to dance clubs, also given is the club affiliation of each dancer. To stay in the spirit of Jack and Jills only dancers from different clubs may dance together. You may assume that each competitor is in the same club with *at most* half of the leaders/followers.

- at most 10 couples can dance in the competition.

1. Design an instance of network flow that can model the above setup. Make sure to clearly describe every part of the graph. Briefly explain your design choices, e.g. why did you add a specific edge.

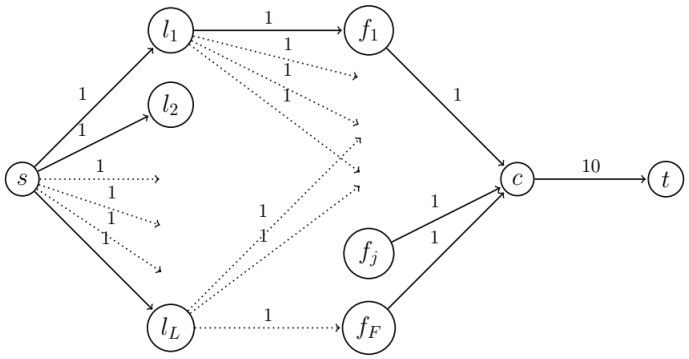
2. Compute the minimum number of edges in this graph as a function of L and F. Briefly explain how you came up with this number.

3. Suppose you found a maximum flow in the above graph. Explain how to find which leader is dancing with which follower given the flow. What is the maximum number of pairs that can compete given the value of the max flow?

Solution:

1. The vertices in G consist of a source s, sink t, one node l_i for each leader, one node f_j for each follower and a node c corresponding to the number of competitors.

   The edges are designed in such a way that each augmenting path carrying one unit of flow corresponds to a competing couple. There is a directed edge (s, l_i) from the source to each leader with capacity $c(s, l_i) = 1$. There is an edge between leader l_i and follower f_j if and only if they belong to different clubs, the capacity is also 1. Further, there is an edge (f_j, c) from every follower to c, with capacity 1. Finally, there is an edge (c, t) with capacity 10. This final edge makes sure that the max flow is at most 10, which corresponds to at most 10 competing couples.



2. There are L edges from s to the leaders, F edges from the followers to c. Each leader is connected to at least $\frac{F}{2}$ followers and there is one more edge (c, t). Thus, there are at least $\frac{FL}{2} + F + L + 1$ edges.

3. Leader l_i is dancing with follower f_j if there is an edge (l_i, f_j) with 1 unit of flow on it. We can find the dancing couples by checking the flow on each such edge. The maximum number of competitors is given by the value of the max flow. For this graph this is equal to the amount of flow on edge (c, t).

Some additional explanation: (The above is enough for full credit) The main idea is that each augmenting path of 1 unit of flow in this graph corresponds to a competing couple.

Since there is a single edge with capacity 1 entering each l_i , at most 1 unit of flow is passing through the node, which ensures that the leader is assigned at most one follower. The same argument can be made for followers; at most one unit of flow can leave a follower f_j. Hence, by flow conservation only one unit can enter. (This is a commonly used trick for network flow problems: setting the capacity to 1 on either the incoming our outgoing side of a vertex ensures that at most 1 unit of flow will go through.)

The edge (c, t) with capacity 10 ensures that the max flow is at most 10. (This is another trick. If you want to verify whether there is a flow of a certain value, then you can "collect" all the flow in a single node c and then add an edge (c, t) with the desired capacity.)