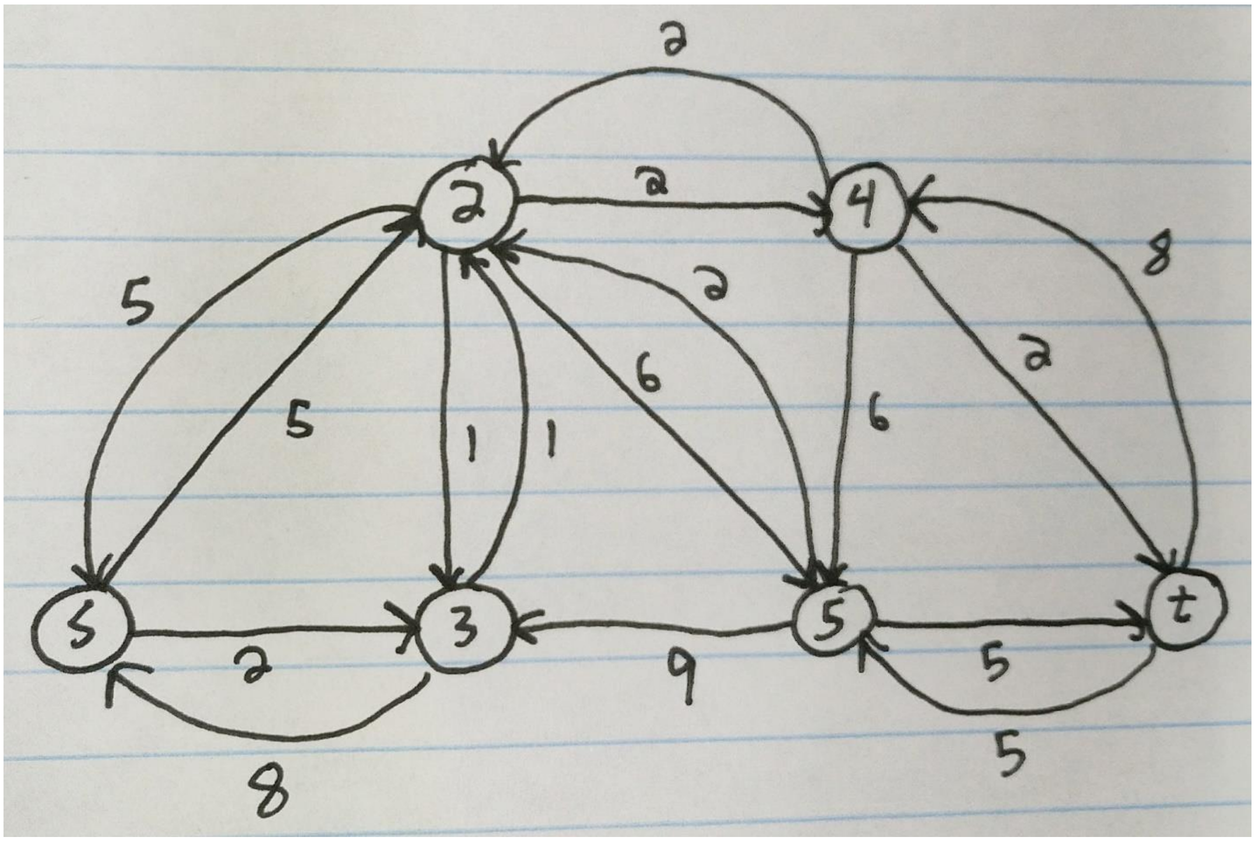
Jeff Hildebrandt

Homework 4

COT 5405

1.

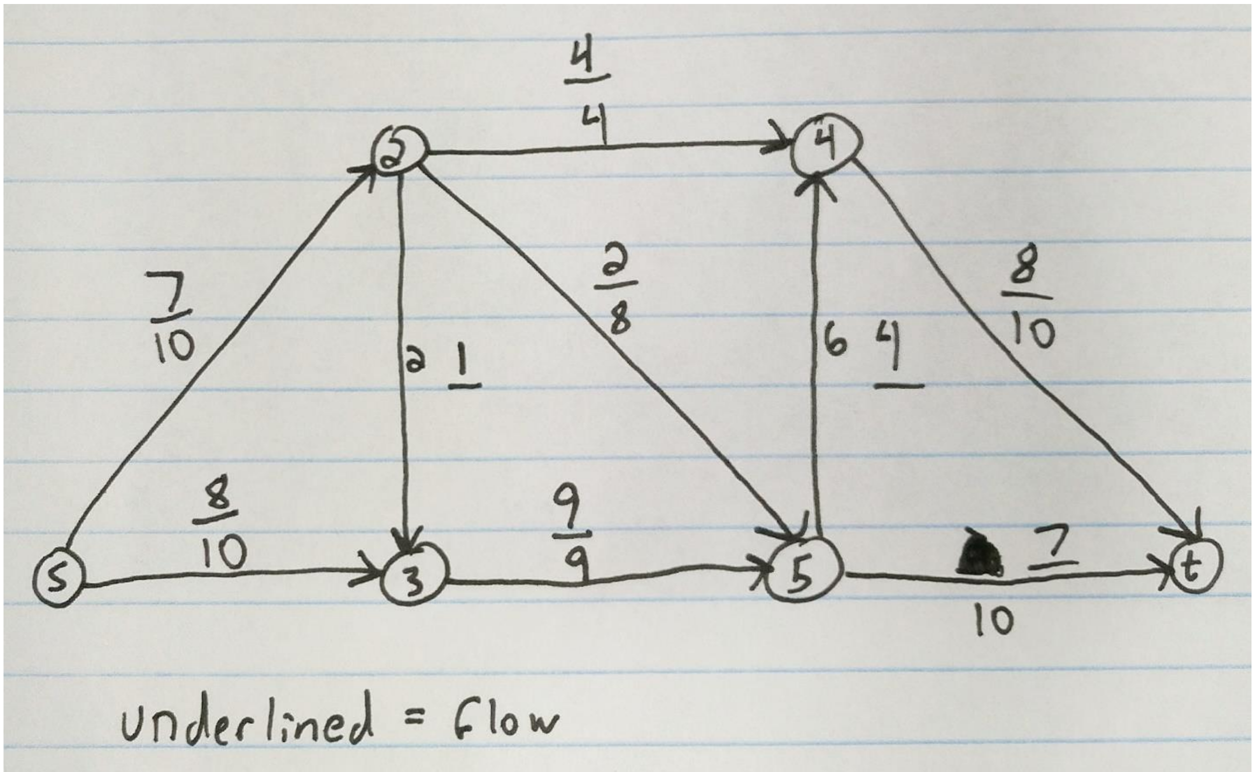
a.



Flow = 13

b.

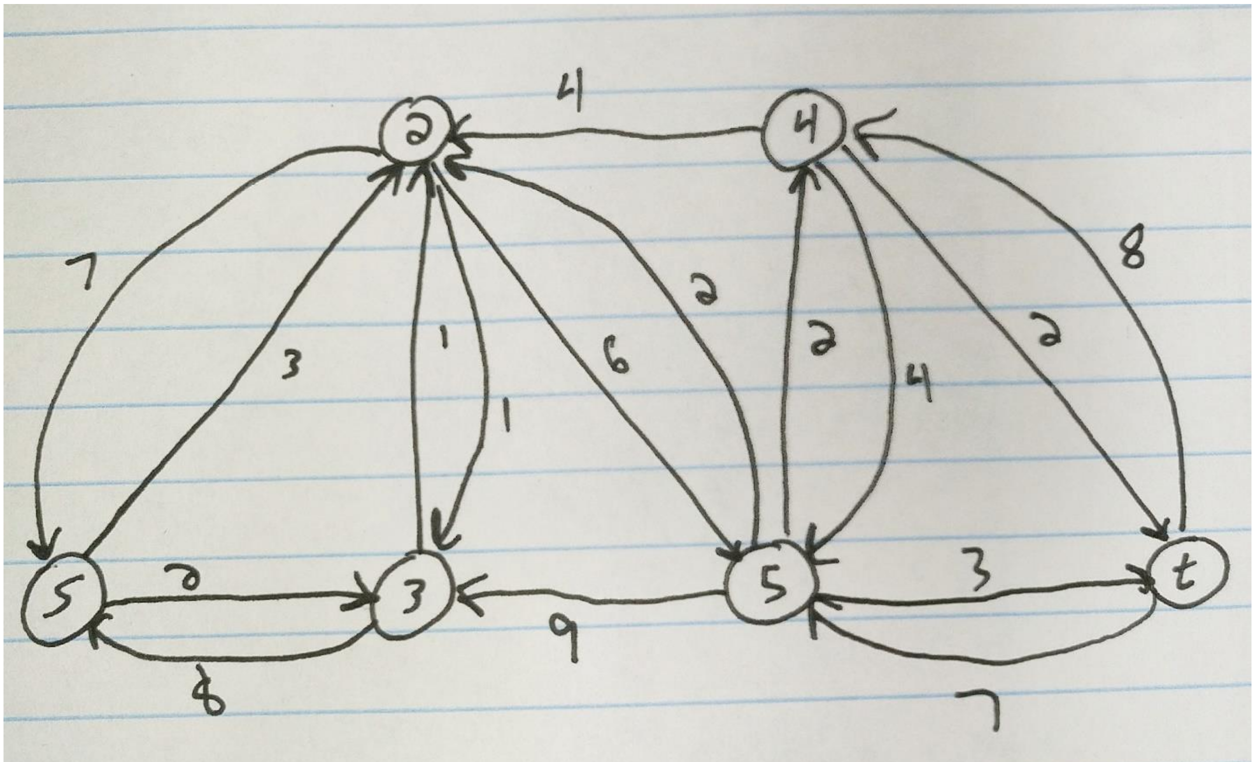
Updated G:



Flow = 15

c.

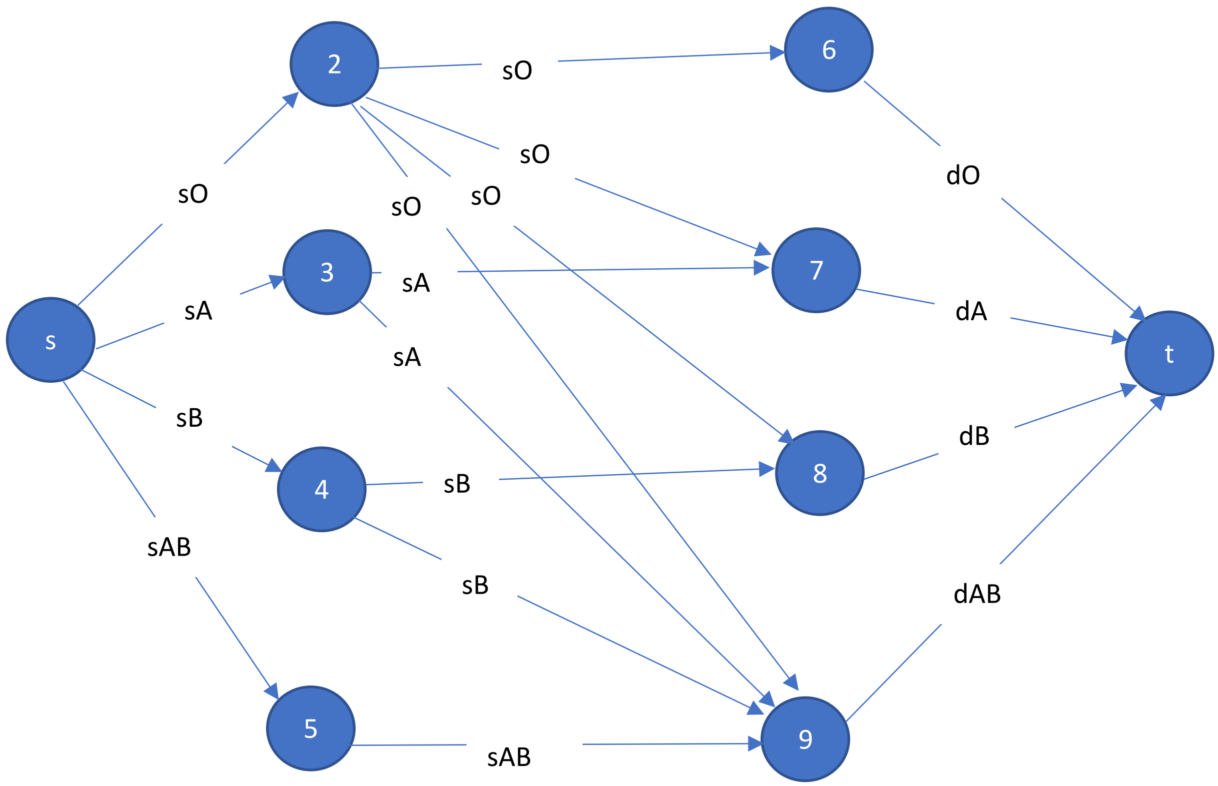
Corresponding residual graph:



2.

a.

This problem can be solved using the Ford-Fulkerson max-flow algorithm, which runs in polynomial time O(E max| f |). There is a total of 10 vertices, 4 for the supply of bags of types of blood, 4 for the demand from students by blood type, then s and t. If the max-flow is equal or higher than the number of people, then there is enough supply. The max that can be on any one line is the amount of supply while leading up to the demand vertices. The set-up of the graph can be seen below:



b.

For this problem, the numbers given in the table did not add up to 100 for the demand, so I used the numbers given earlier in the problem and displayed in the following table:

Blood Type	**Supply**	**Demand**	**Number of bags left over**
O	50	45	0
A	36	42	0
B	11	10	1
AB	8	3	5

So, the number of bags left over chart which is the result after the algorithm is run shows that there are not enough bags for everyone. One person of blood type A will not get the blood transfusion, and there are 6 bags leftover, one of type B and one of type AB.

Explanation:

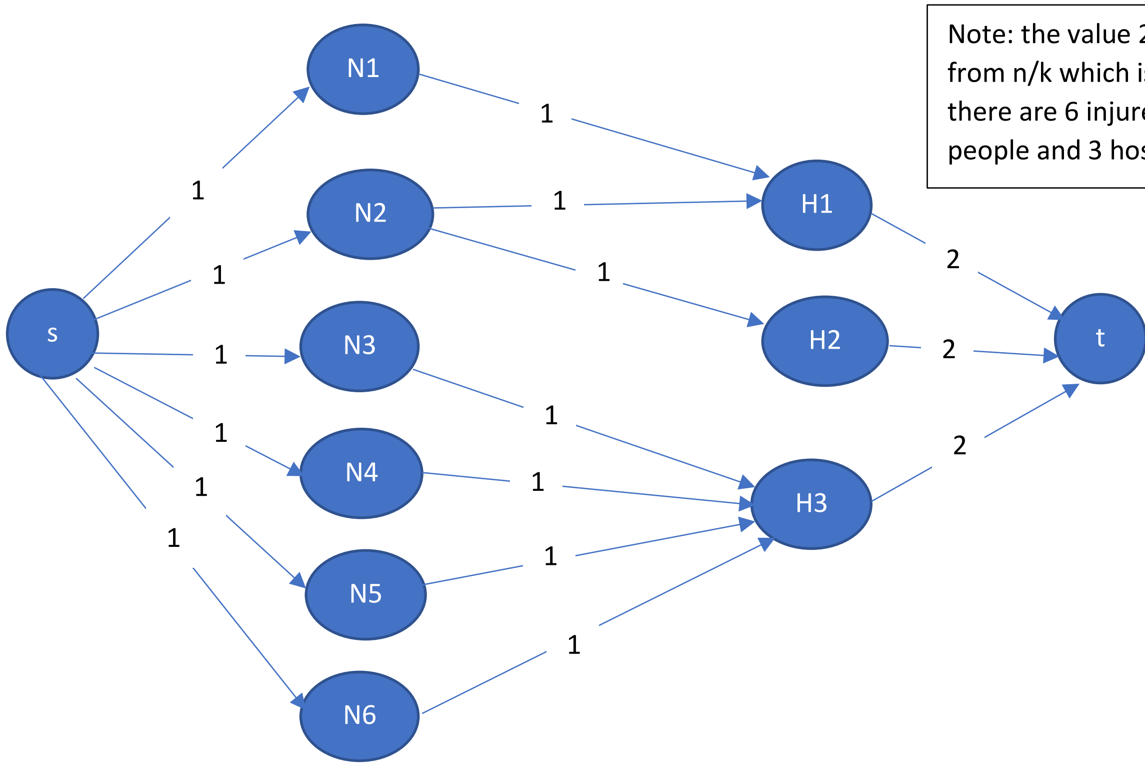
There would not be enough bags for everybody with 1 bag of B and 5 bags of AB left at the end. Unfortunately, one person with blood type A would not get a transfusion. Since AB blood can only be used by AB people, even though there are extra they can't be used. The 5 extras from type O were used for type A, but even though there was an extra bag for type B, it can't be used for people with blood type A. Therefore, there will be one person of blood type A that cannot get a transfusion.

3.

This problem can be solved using the Ford-Fulkerson max-flow algorithm, which runs in polynomial time O(E max| f |). The graph for it can be set up by giving each injured person their own vertex, each hospital their own vertex as well, a source, s, and a sync, t. All the injured people would be connected to s with capacity 1, then all injured people are connected to all the hospitals within a 30-minute drive also with capacity 1. The capacity is 1 since the amount the hospitals can receive is calculated assuming each injured person has a capacity of 1. Every hospital will be connected to t with a capacity of n/k since that was given in the instructions. After the graph is set up as per the previous instructions, the Ford-Fulkerson max-flow algorithm is run. If the max-flow of the graph is equal to the number of total injured people, then it can be determined that every injured person can be placed at a hospital. Here is an example graph based off the following table:

Injured Person	**Hospitals within a 30 minute drive**
N1	H1
N2	H1, H2
N3	H3
N4	H3
N5	H3
N6	H3

In this example two people will not get to the hospital due to four people trying to get to H3 which has a max capacity of two.



Note: the value 2 comes from n/k which is 6/3 as there are 6 injured people and 3 hospitals

4.

a.

Yes, because there is a poly-time certifier for Interval Scheduling, which follows that anything in NP can be reduced to Vertex Cover.

b.

Unknown, because it would resolve the question of whether P = NP. This is the case since both independent set and interval scheduling can be solved in polynomial time. Also, Independent Set is NP complete and solving it in polynomial time lets us assume that P = NP.

5.

Those three clauses if all of them return true, is proof that x0 is true if and only if both x1 and x2 are true. This can be proven by making x1 = 0 and x2 = 1, then x1 = 0 and x2 = 0, then both x1 = 0 and x2 = 0. If all of these return false when plugged into the three clauses then it proves that x0 is true if and only if both x1 and x2 are true.

1. First setting x1 = 0 one of the clauses will return false
    a. We can see $\neg x_0 \; or \; x_1$ which is equivalent to false or false, which returns false
2. Next setting x2 = 0
    a. We can see $\neg x_0 \; or \; x_2$ which is equivalent to false or false, which returns false
3. It follows that setting both x1 = 0 and x2 = 0 would return false due to number 1 and 2