Lab 0

Advanced Algorithms Spring 2021

Lilo Heinrich & Emma Pan

Cassandra, Shashank, Sherrie, and Manu were starting to get bored because they didn't have enough Advanced Algorithms work, so they decided they needed to find themselves a hobby. They found a local badminton league and each joined separate teams. Everyone on the team with the most wins at the end of the season will win a Dunkin Donuts gift card. Shashank has stated that once his team can no longer win the most games, he is going to quit. Therefore he wants to keep track of when his team has been mathematically eliminated from the competition.

Part 1 - Setting up the problem

The current standings are as follows:

				Against			
Team	**Wins**	**Losses**	**Left**	**Sherrie**	**Shashank**	**Manu**	**Cassandra**
Sherrie	83	71	8	-	1	6	1
Shashank	80	79	3	1	-	0	2
Manu	78	78	6	6	0	-	0
Cassandra	77	82	3	1	2	0	-

1) Which teams have been eliminated from getting the Dunkin Donuts prize? Which teams have not been eliminated? Why or why not?

Cassandra and Shashank are eliminated, Sherrie and Manu are still in the tournament. Even if Cassandra wins all of the 3 games she has left, her total wins would be 80 which is still less than Sherrie. Shashank's case is more complicated: if Shashank wins all of his remaining games, he would have a total of 83 wins. Since Sherrie and Manu still have 6 games left to play against each other, one of them will end the tournament with more wins than Shashank. If Sherrie wins at least 1 game, she will have more total wins than Shashank no matter what else happens. If Sherrie wins 0 games, then Manu by default wins all 6 of them, resulting in a total of 84 wins which is greater than Shashank's upper limit of 83. Thus, Shashank is eliminated, but Sherrie and Manu are not eliminated yet because they both have at least one path to victory.

2) Sherrie decides that she's going to be smarter than the rest of the Advanced Algorithms team and create an easy way to tell who's been eliminated. Due to bad record-keeping, Sherrie has access to none of the scores. However, she does have a 5 minute window to look at the standings to quickly determine what teams are eliminated. She decides she is going to set this up as a network flow problem.

The games won will be represented by w_{name} and the games remaining will be represented by r_{name}. For instance, Sherrie has won $w_{Sherrie}$ games and has $r_{Sherrie}$ games remaining. The teaching team trusts you can figure out the variable representation for the other players.

She sets it up as the following network flow:

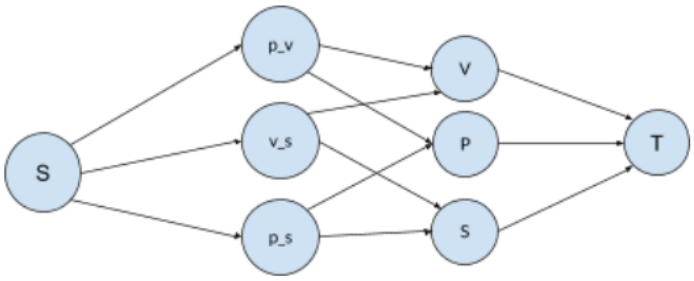

Figure 1: Sherrie's network flow

Figure 1 denotes the network flow diagram that Sherrie constructed to figure out if she was eliminated (note: the flow diagram is specific to her), without any of the capacities. The first node is a source node. The second series of nodes represent the matches between each of the other teams (i.e. Cassandra vs Shashank, Cassandra vs Manu). The third series of nodes represent the other teams (i.e. Cassandra, Shashank). The final node is a sink node.

Construct a procedure for determining if teams are eliminated. Note that to do this, you will have to determine what the capacities of the graph depicted in Figure 1 are. For this question, you must:

1. Draw out the graph with the capacities represented in variable form (explain what variables represent).

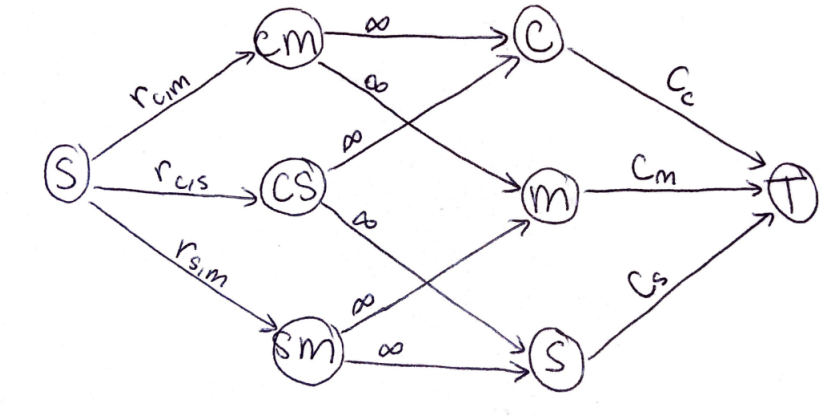

Figure 2: Sherrie's network with capacities

The capacities of the edges flowing from the sink into the team-pair nodes are the number of matches that haven't been played yet between each team combination not involving Sherrie. For example, $r_{c,m}$ is the number of matches left between Cassandra and Manu. These edges represent all of the remaining games that Sherrie does not have control over.

The flows from the team-pair nodes into the team nodes technically have infinite capacity, because we don't know how many games have been won between the each of the pairs. Each player can theoretically win any number of their matches against another player.

The capacity of the flow from the team nodes to the sink is calculated as the following:

$$C_{player} = w_{sherrie} + r_{sherrie} - w_{player}$$

This equation yields the number of games each team would have to win to tie with Sherrie, assuming Sherrie wins all of her matches.

2. Identify what the values of the variables would be for this specific problem

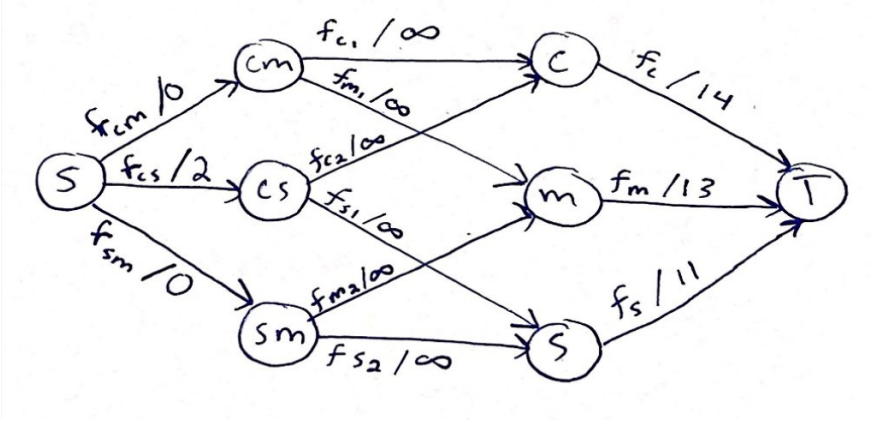

Figure 3: Sherrie's network with flows and capacity values

Here we have plugged in the correct values for each of the capacities and defined names for the variables that will hold the flow through each edge.

3. Write out the strategy for solving the problem.

To find out whether Sherrie is eliminated, first find a maximum flow through the network. Then iterate through the edges from the sink to each team pair. If any of these edges is saturated (the flow is less than the capacity), then Sherrie is eliminated. In this case, Sherrie is not eliminated because the maximum flow value is 2, requiring all of the remaining games to be played. This process can be repeated on a network for any team in any tournament, not just Sherrie in this badminton tournament.

4. Explain why this strategy works.

If the maximum flow through this network does not saturate all of the edges that represent the number of games left between each pair, it means that Sherrie (or whomever the network is for) will certainly lose the tournament. For example, say one of the edges from the sink to a team-pair is not saturated by the maximum flow, leaving at least one match unplayed. This can only be because the edges for the two players who were supposed to play in those leftover matches are saturated, meaning that they are both tied with Sherrie for wins at that point. Yet there is still at least one more game left to be played between them which will increase one of their wins to be greater than Sherrie's, eliminating her.

3) For the network flow diagram you finished above:

1. Convert it into a linear program (using variables, not the values). If you aren't sure how to do this, check out this link: http://www.mathcs.emory.edu/ cheung/Courses/253/Syllabus/NetFlow/max-flow-lp.html.

$$
\begin{aligned}
&\text{max. } && f \\
&\text{s.t. } && f_{c,m} + f_{c,s} + f_{s,m} - f = 0 \\
& && f - f_c - f_m - f_s = 0 \\
& && f_{c,m} - f_{c1} - f_{m1} = 0 \\
& && f_{c,s} - f_{c2} - f_{s1} = 0 \\
& && f_{s,m} - f_{s2} - f_{m2} = 0 \\
& && f_{c1} + f_{c2} - f_c = 0 \\
& && f_{s1} + f_{s2} - f_s = 0 \\
& && f_{m1} + f_{m2} - f_m = 0 \\
& && f_{c,m} \leq 0 \\
& && f_{s,m} \leq 0 \\
& && f_{c,s} \leq 2 \\
& && f_m \leq 13 \\
& && f_c \leq 14 \\
& && f_s \leq 11
\end{aligned}
$$

2. Provide an explanation of why this formulation makes sense, given the original context.

   In this formulation, we have flow capacity constraints (enforcing that the flow through each edge must be less than or equal to its' capacity) and flow conservation constraints (enforcing that the total flow going into each node is equal to the total flow going out of the node). The first constraint equates the flow out of the source to the variable we want to maximize, and the second constraint equates the flow into the sink with the variable we want to maximize as well, transitively equating the flow out of the source with the flow into the sink to enforce the conservation of flow. The next three flow capacity constraints are based on how many games each set of players excluding Sherrie had to play amongst themselves, and the last three are based on how many games each player needs to win in order to win the same number of games as Sherrie's best possible outcome. This makes sense because Sherrie's ability to win is limited by the number of games each player has left to play. The flow capacity constraints are important because each player has a limited number of games that are left to play, and that constraint is presumably defined by the structure of the tournament.

Part 2 - Implementation

Implement the network flows and the linear programming approach to the problem in Python (we are providing input files and starter code).

Make a fork of this github repo: https://github.com/AdvancedAlgorithms/Lab0.

Use "pip install -r requirements.txt" to install the requirements for the right libraries (you might want to use pip3 to use python3).

We have also provided a test file (test_badminton_elimination.py). At a minimum, your code should pass all of the tests in that file. Feel free to add your own additional test cases if you would like to more robustly test your code. If you think the test cases we have given you are sufficient, please explain how either in a comment or in your answer to this question. We aren't evaluating you on this test cases portion, but it's a good exercise to go through. To run your code on a specific input table (defined in a txt file, see teams2.txt and teams4.txt for examples), you can simply run "python badminton_elimination.py teams2.txt"

We recommend using the networkx function to solve the problem using network flows (documentation can be found here: https://networkx.github.io/documentation/networkx-1.10/reference/generated/networkx.algorithms.flow.maximum_flow.html) and using the picos solver to solve the problem using linear programming (documentation can be found here: https://picos-api.gitlab.io/picos/graphs.htmlmax-flow-min-cut-lp).

Your program should be able to answer the following question: Who is eliminated given a table of the current standings? You should be able to do this using a network flows approach and a linear programming approach.

Example input (the 4 at the top represents the # of teams in the division and the remainder of the rows and columns correspond to the same rows and columns as were specified in the table above):

4
Sherrie 83 71 8 0 1 6 1
Shashank 80 79 3 1 0 0 2
Manu 78 78 6 6 0 0 0
Cassandra 77 82 3 1 2 0 0

Corresponding output:
Sherrie: Eliminated? False
Shashank: Eliminated? True
Manu: Eliminated? False
Cassandra: Eliminated? True

To submit this lab - submit a link on Canvas to your Github repository with code and answers to questions 1-3.

Happy coding!