Lab 0

Alex Wenstrup and Hwei-Shin Harriman

Due February 22, 2021 before class

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's team can no longer win the Dunkin Donuts gift card. With 77 wins and 3 games to go, even if they win all of their games and the current best team loses all their games, they will still be 3 games behind.

Manu could theoretically still win the gift card. If he wins all 6 games against Sherrie, he will have 84 wins, and if Sherrie loses to both Cassandra and Shashank as well, she will be stuck at 83 wins, and Manu will secure first place because nobody else can reach 84 wins.

Shashank cannot win the championship. Even if he wins all of his games and EVERYBODY else loses all off of theirs (which is impossible), he will still only be tied for first with Sherrie, and will not receive the Holy Box of Munchkins which he so desires.

Sherrie is currently favored to win, because she only needs to win two out of her eight remaining games to crush Manu's chances of winning a Dunkin' Donuts gift card.

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:

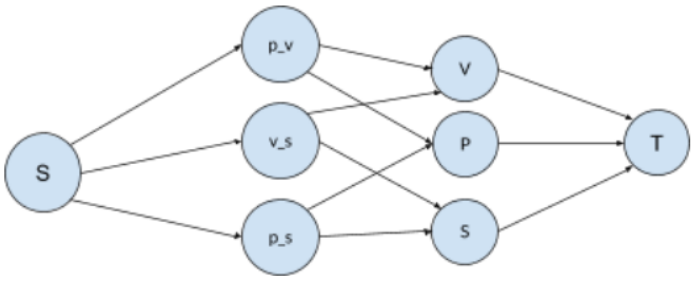

Figure 1: Sherrie's network flow

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 the variables represent).

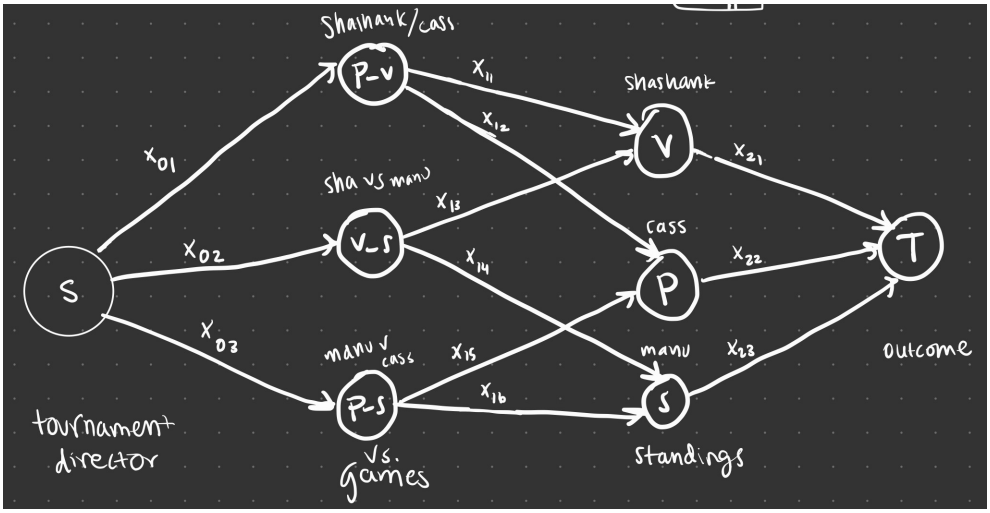

Figure 2: Sherrie's network flow with variables

Beginning with the edges exiting the sink, each edge represents the number of games that are left to play. The edges between the second and third sets of nodes represent the possible outcomes of each game. Because they are limited by the total number of games played, they have no capacities. Finally, the edges between the third set of nodes and the sink represent the number of games that Shashank, Cassandra, and Manu could win before they ruin Sherrie's chances of winning the tournament.

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

   - $x_{01} = 2$, representing the remaining number of games that Shashank and Cassandra will play against each other.

- $x_{02} = 0$, representing the remaining number of games that Shashank and Manu will play against each other.
- $x_{03} = 0$, representing the remaining number of games that Manu and Cassandra will play against each other.
- x_{11} through x_{16} are equal to ∞ since they have no capacities.
- $x_{21} = 10$, the number of games Shashank can win before Sherrie is eliminated.
- $x_{22} = 13$, the number of games Cassandra can win before Sherrie is eliminated.
- $x_{23} = 12$, the number of games Manu can win before Sherrie is eliminated.

3. Write out the strategy for solving the problem.

   To figure out if Sherrie still has a chance to win, we can first assume that she will win all of her remaining games. In this case, we assume she wins all 8 remaining games, for a total of 91 wins.

   Next, want to check if there is ANY way for Sherrie to stay on top of the leader board. To do this, we will find the maximum number of games everyone else can win WITHOUT eliminating Sherrie. In the given example, Shashank can win up to 10 more games because $80 + 10 = 90$, and Sherrie can win up to 91 games. We can calculate the number of games Manu and Cassandra can win with the following formula: $w_{sherrie} + r_{sherrie} - w_{other} - 1$. In Manu's case, he can win $83 + 8 - 78 - 1$ games, or 12 games. Cassandra can win up to 13 games without eliminating Sherrie from the tournament.

   Note: if any of these capacities turn out to be negative, that means that player has already won enough games that it is impossible for Sherrie to catch up. If this happens, we don't need to do the network flow problem; Sherrie is eliminated from winning the Dunkin Donuts prize.

   Also note: The -1 term at the end of the formula above represents that nobody wins the prize in the case of a tie.

   We use these capacities to construct the graph above. Once this graph is constructed, checking if Sherrie is eliminated is as simple as finding the maximum flow through the graph. If the maximum flow is equal to the number of games that still have to be played between the other three players (not including Sherrie), then we know that there is a possibility that all remaining games can be played without knocking Sherrie off the top of the leader board.

   If the maximum flow is NOT equal to the number of games remaining, then Sherrie cannot possibly win the tournament. It is not possible to play all of the remaining games without somebody bumping Sherrie off the top of the leader board. Even after winning all of her games, it is guaranteed that someone else will catch up to her, and she will not win the prize.

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/323/Syllabus/NetFlow/max-flow-lp.html.

   Before we begin, we check if any of the capacities in the third layer of our graph are negative. If this is the case, we know that another team already has so many wins, the team in consideration has no way of catching up. In this case, we don't need to solve the problem at all; we know the team is already eliminated.

The network flow problem can be converted into the following (large) system of inequalities.

First, we enforce that all flows are non-negative. This is because each edge represents a number of games played or won, and we cannot "take back" games after they have happened.

$$f_i \geq 0, \forall i \in \{01, 01, 03, 11, 12, 13, 14, 15, 16, 21, 22, 23\}$$

Next, we enforce that all flows are less than or equal to their capacities. The capacities in the first layer indicate games remaining, and we cannot play more games than those which remain. In the third layer, capacities represent how many games each team is allowed to win. If we let them win any more games, the team we are looking at would be eliminated, even if still have a chance in the actual tournament.

$$f_i \leq x_i, \forall i \in \{01, 01, 03, 21, 22, 23\}$$

Next, we enforce that the net flow through each node is 0 (except for the source and drain nodes). This just means that we want to ensure that the sum of each team's wins and losses equal the number of games that they play.

$$f_{01} - f_{11} - f_{12} = 0$$
$$f_{02} - f_{13} - f_{14} = 0$$
$$f_{03} - f_{15} - f_{16} = 0$$
$$f_{11} + f_{13} - f_{21} = 0$$
$$f_{12} + f_{15} - f_{22} = 0$$
$$f_{14} + f_{16} - f_{22} = 0$$

Lastly, we define the max flow to be equal to the sum of all flows leaving the source, where F is the maximum flow of the network. Remember, each of these flows represents a game that still has yet to be played. To maximize F is to maximize the number of games that we play. Ideally, we can play all of the remaining games.

$$F = f_{01} + f_{02} + f_{03}$$

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

   Explanation provided in-line.

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!