Hyegi
Lab 0

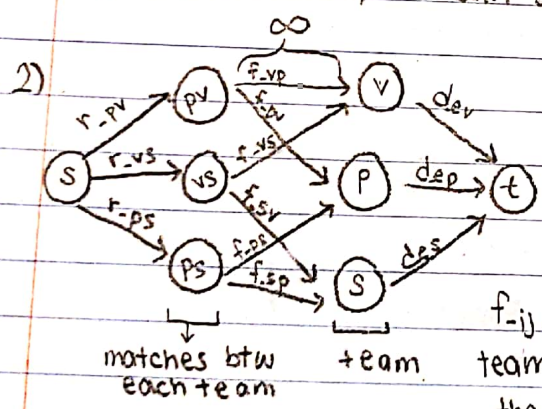
1) → Vicky have been eliminated as she currently has 77 wins with only 3 games left. Even though Vicky wins the remaining games, she only has 80 wins, which is less than Emily's current wins.

→ Shashank have not been eliminated as he currently has 78 wins, with 6 more games. If he wins all 6 games against Emily, he ends with 84 wins, which is greater than Emily's current win. In order for Shashank to win, Emily has to her all remaining games, resulting a total wins of 83.

→ Prava have been eliminated as she currently has 80 wins with only 3 games left. Even though Prava wins the remaining games, she has 83 wins, which ties with Emily's current score. If Shashank beats Emily all 6 times, he has a total score of 84. If Emily beats Shashank at least once, she has a total score of 84, which is greater than max score Prava can get. Therefore, Prava can't result with the highest score.

→ Emily is not eliminated as she currently has 83 wins and 8 games left. If she wins all 8 games, she will result in a total score of 91, which is the highest score anyone can achieve.

2)



· r_ij - remaining game between i and j

· d_ej - The difference between Emily's max total win and j team's win : $w_e + r_e - w_j$; this ensures the limitation for Emily → $91 - w_j$

f_{ij} - The edges connecting the nodes from matches btw each team, i j, and nodes from team column, ij has a capacity of the win team i had over team j. Due to conservation constraints, the diagram represents its capacity as ∞.

· By setting the inward edges of sink, d_{ej}, as the difference btw Emily's max win and team j's max win. This allows to test Emily's elimination as team j's max win has to be less than or equal to Emily's max win. [unless it will result in a negative flow]. From this, we know after flow has been maximized, if not all flow from the source has been maximized, there is no scenario for Emily to win.

· This network's maximal flow can be found through Ford-Fulkerson or Edmond's-Karp algorithm. Then, we'll look at the min-cut. If min-cut consists source node, it implies that all games have been played and there is a case where Emily wins. If it doesn't consist source node, it implies that there is no scenario for Emily to win

CamScanner로 스캔하기

x_{ij} = total numbers of games played between player i and j

3) maximize : $p = x_{pv} + x_{ps} + x_{vs}$. We are trying to optimize p to its max value where all games have been played, including the remaining games

Flow conservation constraints:

LHS node pv : $r\_pv = f\_vp + f\_pv$

node vs : $r\_vs = f\_vs + f\_sv$

node ps $r\_ps = f\_ps + f_{sp}$

RHS node v: $f\_vp + f\_pv = d\_ev$

node p: $f\_vs + f\_sv = d\_ep$

node s: $f\_sp + f_{ps} = d\_es$

LHS ↳ This set shows that number of remainder game between player i and j. equals to a sum of win player i had against player j and vice versa. This is true as there is always an winner for the game and no additional win can be made if game isn't played

RHS: This set shows the difference between Emily's maximum total win and person i's total win. This ensures Emily's final win as no flow can have a negative capacity.

This network allows Emily to either prove that individual's max wins are less than Emily's optimal total number of wins or the validation of a situation where Emily can be overthrown on her 1st place by others.

→ All variables needs to be an non-negative number

2 cont'd) $r_{pv} = 2$ # of matches btw Prava & Vick

$r_{vs} = 0$ " " Vicky & Shasank

$r_{sp} = 0$ " " Shashank & Prava

$l_p = 11$ # of wins Prava need to tie Emily

$l_v = 14$ " Vicky "

$l_s = 13$. " Shashank "

CamScanner로 스캔하기