Exercise 5

Let's say we want to check if a team $t_k$ can win. We can bound the maximal number of additional to $\text{score}(j)$ points any other team $t_j, j \neq k$ can get (still having $t_k$ winning) as follows:

$$s_k = \sum_{j \neq k} \text{games}(k,j) \quad (1)$$

$$s_j = s_k + \text{score}(k) - \text{score}(j) - 1, \ j \neq k \quad (2)$$

Consider the following max flow network:

Connect $s$ with $\text{game}(i,j)$ by an edge with capacity $\text{games}(i,j)$, $\text{game}(i,j)$ with teams $i,j$ with capacity $\mathcal{G}$ each team $t_i, i \neq k$ with $t$ with capacity $s_t$.

We know that the team $k$ can attain $s_k$, so we omit all the games with team $k$ in the graph. By finding a maximal flow in the network, we can find out if the rest of the teams can play such that their respective scores do not exceed $s_j$ (which would make team $k$ win). If the maximal flow equals $\sum_{i,k \neq k,j \neq k} \text{games}(i,j)$, then its possible to to distribute scores such that no team's score exceeds $s_j$.

For the chess tournament, we have to make a small modification of the graph above. Namely, now each game gives 2 points to the playing teams. Hence just multiply all the capacities by 2, except for the capacities connecting teams with $t$. Now $s_k = 2 \sum_{j \neq k} \text{games}(i,j)$ and $s_j$ are defined as previously.

The running time of the algorithm is as follows: there are $2 + n - 1 + n(n - 1)/2$ vertices (for $s, t, t_i, t_j$, every team and every game) and $n(n - 1)/2 + 2n(n - 1)/2 + n - 1$ edges, as every game connects exactly 2 teams and there are $(n - 1)n/2$ possible games. Hence the running time is $O(n^6)$.

The main difficulty with the football tournament is that we cannot express games as a network flow in the same way, as we don’t have flow preservation anymore. In the paper “Football Elimination is Hard to Decide Under the 3-Point Rule” you can find a proof that the problem is actually NP-complete.

**Correction:** there are $2 + (n-1) + \frac{(n-1)(n-2)}{2}$ vertices

and $\frac{(n-1)(n-2)}{2} + 2 \frac{(n-1)(n-2)}{2} + (n-1)$ edges

there are $\frac{(n-1)(n-2)}{2}$ possible games.
2) We know that in the network $G$, $\text{max-flow} = \text{min-cut} = k$ and all edges of some min-cut were destroyed.

$s = v_1, v_2, \ldots, v_k = t$.

If edge $(v_i, v_{i+1})$ was destroyed, $\text{ping}(v_i) = \text{true}$ if $i < j$ and $\text{false}$ if $i > j$. We use binary search to determine which edge was destroyed.

```plaintext
e = 0; r = n;
while (r - e > 1)
{
    n = \lfloor (e+r)/2 \rfloor;
    if (\text{ping}(v_n) = \text{true}) e = n; else r = n;
}

return edge $(v_e, v_{e+1})$ was destroyed.
```

We have $k$ paths, each path has a length of at most $n = \sqrt{k}$, we use at most $O(k \log n)$ ping operations.
Exercises for Unit 27

3) One can come up with a counter-example as follows:

capacity of all edges is 1

Without reverse edges, one can only send a flow of 1 unit (if the augmenting path is as indicated) whereas the maximum flow can be arbitrarily large.