**CS218 Design and analysis of algorithms** **Jan-Apr 2023**

# Exercises: Network Flow (No submission)

1. You have an alphabet of size $n$. You are given an encoding in $\{0, 1\}^*$ for each letter in your alphabet. How will you determine if the encoding is uniquely decodable? That is, whether there are two strings from you alphabet get mapped to the same 0-1 string?

2. For an $s$-$t$ path, the bottleneck is defined to be the least capacity of an edge on that path. Can you design an efficient algorithm to find the $s$-$t$ path with largest bottleneck? Or try this variant: given a threshold $\lambda$, is there an $s$-$t$ path where every edge has capacity at least $\lambda$?

3. Use max flow algorithm to solve the following problem. Given a graph with a source vertex $s$ and a destination vertex $t$, find the minimum number of vertices which can be removed to make $s$ and $t$ disconnected.

4. Recall the reduction from partitioning of a given partially ordered set into minimum number of chains to maximum flow. Prove that the algorithm indeed gives the minimum number of chains.

5. Use the max flow min cut theorem to prove that for any partially ordered set, the minimum number of chains in which we can partition the set is equal to the maximum number of mutually incomparable elements. You can use the same reduction to max flow. Assume max flow is equal to $s$-$t$-minimum-cut. Then given a $s$-$t$ cut with outgoing capacity of $k$, try to construct a set of $k$ mutually incomparable elements.

6. **Project selection** We are given a set of projects, where some projects are pre-requisites for others. This can be represented by a direct acylic graph on projects. An edge from $i$ to $j$ represents that $i$ is a pre-requisite for $j$. Clearly this is a transitive relation. A project can be done only if all its pre-requisite projects have been done.

   Some projects might have a positive value, i.e., you gain a net profit from them. While some other projects may have a negative value, i.e., you have to invest more into them than what you gain from them. The goal is to select a subset of projects which maximizes the total value, under the constraint that if a project is selected then all its pre-requisites must also be selected.

   Reduce this problem to $s$-$t$-minimum-cut problem. That is design an algorithm which can use $s$-$t$-minimum-cut subroutine. Note that minimum cut is also a subset (of vertices) selection problem.

7. Consider a two player game between Ankita and Puneet. Ankita starts the game by saying the name of an Indian film actor/actress, say X. Puneet has to respond with the name of any actress/actor, say Y, that has appeared in a film opposite X. Then Ankita has to respond with the name of any actor/actress, say Z, that has appeared in a film opposite Y. And they continue like this. At each player's turn, she/he has to respond with the name of an actor/actress that has appeared opposite the actress/actor whose name was last taken by the other player. Naturally, the names cannot be repeated. The player who cannot come up with a name loses.

   For simplicity, assume that they consider films only from last 10 years. Suppose both players have all the information about these films. One can represent this information simply by a directed graph – we put an edge from an actress to an actor if the two have a film together. Each player wants to use this information to come up with a winning strategy.

   Construct a flow network by adding a source, edges from source to each actress, a sink edges from each actor to the sink. All edges have capacity 1. Clearly, the maximum flow is upper bounded by the number of actors and also by number of actresses.

   - Prove that if the number of actors is equal to the number of actresses and the maximum flow is equal to that number, then the Puneet has a winning strategy, irrespective of how Ankita plays.

- Prove that if the condition above is not true then Ankita has a winning strategy, irrespective of how Puneet plays.

8. Recall the bipartite matching problem. Given a bipartite graph, find the largest set of edges such that no two of them have a common endpoint. We write the following linear program. Each edge $e$ has a variable $x_e$.

$$\max \sum_e x_e$$
$$\text{subject to}$$
$$\sum_{e \text{ incident on } v} x_e \leq 1$$
$$1 \geq x_e \geq 0$$

In general, it is not necessary that a linear program has a integer optimal solution. But, it is true for the above linear program. Use the connection of bipartite matching with maximum flow and prove that the above linear program always has an integer (0/1) optimal solution. Recall the flow algorithm, which always gives an integer flow, if the capacities are integer.