# CSE Template

Rohan Mukherjee

May 22, 2024

## Problem 1.

We use a truly mindblowing trick to solve this problem. First replace every edge of $G$ with two directed edges, one going forward and the other going backwards. Then for each $v \in G$, replace $v$ with two vertices $v_{in}$ and $v_{out}$, with one edge $v_{in} \to v_{out}$, and for each neighbor edge $w \to v$ add $w \to v_in$ and for each $v \to u$, add $v_{out} \to u$. After running this on the entire graph, let $\hat{S}$ be the collection of $s_{in}$'s attained in this way from the original vertices in $S$, and similarly $\hat{T}$ to be the collection of $t_{in}$'s attained in this way from the original vertices in $T$. Remove all the $t_{out}$s and $s_{in}$s, and add the vertices $\hat{S}$ and $\hat{T}$ with edge $\hat{S} \to s_{out}$ for each $s_{out}$ in the graph, and $t_{in} \to \hat{T}$. Return the number of edge-disjoint paths from $\hat{S}$ to $\hat{T}$ in the new graph.

We now prove the correctness of the above algorithm. Suppose we start with $\mathcal{P}$ being some set of edge disjoint paths in the original graph. Let $P = s = v^{(1)}, \dots, v^{(n)} = t$ be one of those paths. We then can put $P$ on the transformed graph $G'$ by $v_{in}^{(1)} \to v_{out}^{(1)}, \dots, v_{in}^{(n)} \to v_{out}^{(n)}$. This is a path in the new graph by construction, and the set of such paths are edge-disjoint because the original ones were vertex disjoint by what we did in class.

Similarly, given a set of edge-disjoint paths in the new graph, we can put them backe on the original graph by noticing the following: if $v_{in}$ is in the path $P$ for some $v \in G$, then we must have $v_{out}$ be the next vertex in the path because that is the only place $v_{in}$ goes to. Recalling then that we start at some $s_{in}$ for some $s \in S$ and end at some $t_{out}$ for some $t \in T$, we then notice that every path is of the following type:

$$s_{in} \to s_{out} \to v_{in} \to v_{out} \to \dots \to t_{in} \to t_{out}$$

From here we can put this path on the original graph by decreeing that $s \to v \to \dots \to t$ in the original graph. The set of such paths in the original graph is vertex disjoint because

otherwise we would be using the edge $v_{in} \to v_{out}$ multiple times, which completes the proof.

For the runtime, we added one vertex and one middle edge for each vertex not in $S$ or $T$. Thus our new graph has potentially less vertices than the original and $O(n)$ more edges, which shows that after constructing the new graph in linear time, Ford Furkerson runs in polynomial time on the new graph and hence the full algorithm is polynomial time.

## Problem 2.

Suppose instead that the max number of edge-disjoint paths from $s$ to $u$ is $\ell < k$. By Mengers theorem, this says that the min cut between $s$ and $u$ is just $\ell$. Thus we can remove $\ell$ edges from the digraph $G$ to separate $s$ and $u$, i.e. put them into two different connected components. Since there are at least $k$ edge-disjoint paths from $s$ to $t$ and $t$ to $u$, we need to remove at least $k$ edges to separate $s$ from $t$ or $t$ from $u$. In particular, since we removed $< k$ edges from the graph, there is still at least one path from $s \to t$ and $t \to u$. Walking along this first path to $t$ and then walking along the second path to $u$ shows that $s$ and $u$ cannot possibly be separated, a contradiction. Thus the max number of edge-disjoint paths from $s$ to $u$ is $\geq k$.

## Problem 3.

Given a min $s - t$ cut $(A, B)$ of $H$, we can construct a vertex cover $S \subset X \cup Y$ by doing the following. First initialize $S \leftarrow \emptyset$. Then for each edge $e$ crossing the cut, we have 3 cases:

1. $e = (s, x)$ for some $x \in X$. Then $S \leftarrow S \cup \{x\}$.
2. $e = (y, t)$ for some $y \in Y$ Then $S \leftarrow S \cup \{y\}$.
3. $e = (x, y)$ for some $x \in X, y \in Y$ Then $S \leftarrow S \cup \{x\}$.

We shall show that this is indeed a vertex cover. Let $e = (x, y)$ be an edge. If $e$ crosses the cut, then we have added $x$ to $S$ by the third step. Otherwise, $e$ does not cross the cut, which means that either $x, y \in A$ or $x, y \in B$. In the first case, this means that the edge $(y, t)$ crosses the cut, which shows that $y \in S$, and the second case is similar. Since for each edge we added $\leq 1$ vertex to the set $S$, we have that $|S| \leq \text{cap}(A, B)$. We show this is in fact an equality. Suppose that some $x \in X$ was added twice to $S$ via the above algorithm. This would mean there is some $y$ so that $x \to y$ crosses the cut while also $s \to x$ crosses the cut. But this is impossible–for the first says that $x \in A$ while the second says that $x \in B$. Thus $S$

is a vertex cover of size cap$(A, B)$.

$$s \longrightarrow x \longrightarrow y \longrightarrow t$$

Finally we see that each $y \in S \cap Y$ can be added at most once by the second step because there is at most one edge of the form $y \to t$ for each $y$.

Now let $S$ be a minimum vertex cover for $G$. For each $x \in S \cap X$, remove the edge $s \to x$ from $H$, and for each $y \in S \cap Y$, remove the edge $y \to t$. Could it be that we can still reach $t$ from $s$? By construction, we would need to walk along the path $s \to x$, $x \to y$, and $y \to t$. But this cannot possibly happen as either $x$ or $y$ is in the vertex set, whence the edge $s \to x$ or $y \to t$ has been removed. Recall in class that the minimum number of edges to remove to separate $s$ and $t$ is the same as the min cut, from class. This shows that the min cut has size $\leq |S|$. From the last part we know that the min cut has size $\geq |S|$, which shows there exists a min cut with size $|S|$, completing the proof.

Now our algorithm is as follows:

Given a bipartite graph $G = (X \cup Y, E)$, we want to find a minimum vertex cover.

**Input:** Bipartite graph $G = (X \cup Y, E)$

**Output:** Minimum vertex cover $S$

Create the graph $H$ by adding two vertices $s$ and $t$;

Connect $s$ to each vertex in $X$ and connect each vertex in $Y$ to $t$;

Find a minimum $s$-$t$ cut $(A, B)$ in $H$;

Construct the vertex cover $S$ from $(A, B)$ as follows:;

  Initialize $S \leftarrow \emptyset$;

  For each edge $e$ crossing the cut, do the following:;

    If $e = (s, x)$ for some $x \in X$, then $S \leftarrow S \cup \{x\}$;

    If $e = (y, t)$ for some $y \in Y$, then $S \leftarrow S \cup \{y\}$;

    If $e = (x, y)$ for some $x \in X, y \in Y$, then $S \leftarrow S \cup \{x\}$;

Return $S$;

The above steps show the correctness of this algorithm. This algorithm runs in polynomial time because we have added 2 vertices and $O(n)$ edges, and then we run Ford Furkerson on this graph, which is polynomial time.

We claim there is an inclusion-reversing bijection between vertex covers and independent sets by $S \mapsto V \setminus S$. Indeed, if $S$ is a vertex cover, then for each $x, y \in V \setminus S$, we cannot possibly have $x \to y$ because otherwise the edge $x \to y$ would have neither of its endpoints in the vertex cover $S$. Similarly, if $I$ is an independent set, then let $e = (x, y)$ be any edge. We must have either $x \in I^c$ or $y \in I^c$, otherwise both $x, y \in I$ which shows that $I$ is not

independent. As $U \subset V$ is equivalent to $U^c \supset V^c$, we have shown the claim. In particular, finding the maximum independent set is thus easily seen to be equivalent to finding the minimum vertex cover, by this argument, which completes the proof.

## Problem 4.

Make a graph from the $n \times n$ chess board by making each allowed square a vertex, and connecting two vertices with an edge if the corresponding squares can be attacked by a knight. Make $G$ bipartite by coloring the board with adjacent squares having different colors. Then return the maximum independent set on this graph.

First, since every knight can attack at most 8 squares, every vertex in this graph has degree $\leq 8$, thus Ford-Ferkerson runs in polynomial time.

We now prove the correctness of this algorithm. Given a maximum independent set $I$ in the graph, we can construct a set of knights by placing a knight on each square in $I$. This shows that the maximum number of knights on the board that are not attacking each other is bounded below by the maximum independent set. Similarly, given a set of knights that cannot attack each other, we can add the corresponding vertices to our set and get an independent set, which shows that the maximum independent set is bounded below by the maximum number of passive knights. This completes the proof.