COL351 Fall 2021 Assignment 4

Saptarshi Dasgupta, Tamajit Banerjee

TOTAL POINTS

60 / 60

QUESTION 1

1 Max-Flow Min-Cut 20 / 20

✓ **- 0 pts** **Correct**

- 20 pts No solution

- 5 pts Not correct proof

- 5 pts For not finding set of edges X / X is set of edges in H

- 7 pts No proof of correctness

QUESTION 2

2 Hitting Set 20 / 20

✓ **+ 5 pts** **HS is in NP**

✓ **+ 7 pts** **Poly Time Reduction (VC to HS)**

✓ **+ 4 pts** **"Yes" VC => "Yes" HS**

✓ **+ 4 pts** **"Yes" HS => "Yes" VC**

+ 0 pts Incorrect

QUESTION 3

3 Feedback Set 20 / 20

✓ **+ 5 pts** **Shown that it is in NP class**

✓ **+ 15 pts** **Correct reduction from vertex cover**

+ 7 pts Partially correct reduction

+ 13 pts Used directed feedback set

+ 3 pts Partially correct NP proof

COL351-A4

Saptarshi Dasgupta (2019CS50447)
Tamajit Bannerjee (2019CS10408)

November 2021

1 Flows and Min Cuts

Q. Let $G = (V, E)$ be a directed graph with source s and $T = \{t_1, \ldots t_k\} \subseteq V$ be a set of terminals. For any $X \subseteq E$, let $r(X)$ denote the number of vertices $v \in T$ that remains reachable from s in $G - X$. Give an $\mathcal{O}(|T|.|E|)$ time algorithm to find a set X of edges that minimizes the quantity $r(X) + |X|$. (Note that setting X equal to the empty-set is allowed).

1.1 Algorithm

1. Make new Graph H with edges and vertices of G
2. Add vertex t to $V(H)$.
3. For each terminal vertex t_i of $V(G)$, add edge $e = (t_i, t)$ to $E(H)$.
4. Compute $f = s$ - t max flow of H under with capacity of each edge being 1 using Ford-Fulkerson algorithm.
5. Compute A, set of vertices reachable from s using DSF in residual graph H_f and then compute A' the set of vertices not reachable from s.
6. Iterate over edges of H and return those edges that originate from some vertex in A to some vertex in $A' - \{t\}$

1.2 Complexity

Let V, E be vertex and edge sets respectively of graph G.

Step 1 and 2 takes $O(|E| + |V|)$ time together as we have to iterate over all vertex and edges. Step 3 takes $O(|T|)$ time as we need to iterate over vertices of T. Now, number of vertices of H is $|V|+1$ and its s-t max flow is atmost $|T|$ as exactly $|T|$ edges are incident on t. This gives complexity of step 4 to be $\mathcal{O}(|E(H)|.|T|) = \mathcal{O}((|E| + |T|).|T|) = \mathcal{O}(|E||T|)$.

Step 5 requires time $\mathcal{O}(|E| + |T| + |V|)$ as time complexity of DFS is $O(m + n)$

Step 6 requires us to iterate over edges of H and for each edge checking if vertices belong to set A takes $O(1)$ time, so total time in this step is $\mathcal{O}(m) = \mathcal{O}(|E| + |T|)$

Thus total complexity of algorithm is $\mathcal{O}(|E|.|T|)$

1.3 Correctness

Claim 1. *There exists a 0-1 max flow in H.*

Proof. We know that, at every intermediate stage of the Ford-Fulkerson Algorithm, the flow values f(e) and the residual capacities in the residual graph G_f are integers as all the capacities in the flow network are integers.

We know that, the Ford-Fulkerson algorithm terminates with the max-flow.

From the above 2 properties, we get if all capacities in the flow network are integers, then there is a maximum flow f for which every flow value f(e) is an integer.

So max flow f(e) found by the Ford-Fulkerson Algorithm is a 0/1 flow in H. □

Claim 2. *If there are k edge-disjoint paths in the directed graph H from s to t, then the value of the maximum s-t flow in H is at least k.*

Proof. We will proof by construction. Suppose there are k edge-disjoint s-t paths. We can set the flow to be $f(e) = 1$ for each of the edges e in any of the paths and $f(e) = 0$ otherwise. Hence now each of these paths carry one unit of flow. This defines a feasible flow of value k. Hence proved. □

Claim 3. *If there exists an 0/1 s-t flow of value f in graph H, then there are atleast f Edge Disjoint Paths from s to t in the graph.*

Proof. We prove this by induction on the number of edges carrying the flow.

Base Case. Let number of edges carrying flow be 0. Then trivially, number of edge disjoint paths is atleast 0.

Induction Hypothesis If in any graph with flow f number of edges carrying flow is less than k then that graph has atleast f s-t Edge Disjoint paths.

Induction Step. Assume graph has a flow of value f with k edges carrying flow. Since flow is nonzero, there is atleast one edge carrying nonzero flow. So, either that edge terminates at t or goes to another vertex which must have an edge that carries the flow out of it since total incoming flow must be equal to total outgoing flow for all vertices except the source and the sink.

Now, this chain of vertices will either end at the terminal node t or end up back at some vertex which we have a;ready traversed.

If it ends up at terminal node, then we obtain a s-t path carrying 1 unit of flow. We construct a new graph by removing these edges. Thus new graph will have a flow of $f-1$ and it will have number of edges carrying flow less than that of original graph (k). So, we apply Induction Hypothesis and get $f - 1$ Edge disjoint paths of new graph. Now combined with 1 path which we removed we get atleast f Edge Disjoint Paths in graph H.

If the chain of vertices end up back again at any vertex v, then we simply remove the cycle and obtain the new graph where we number of edges carrying flow is less than k. Again Induction Hypothesis applies and we obtain atleast f Edge Disjoint Paths. □

Now we claim that the maximum s-t flow in H is the minimum value of $r(X) + X$ in G.

We will prove that $r(X) + X \geq f$, where f is the max s-t flow in H and the find a set X_0 for which the equality holds.

Claim 4. *For any $X \subseteq V$ with $|X| = k$, max-flow$(H - X) \geq f - k$*

Proof. By **Claim 2** and **Claim 3** Max flow of H is equal to maximum number of EDP (edge disjoint paths) from s to t. Now, if we remove remove set X ($= k$ edges) from H, number of EDP can decrease by atmost k as removal of an edge can remove atmost one EDP. So, max flow of graph $H - X$ is atmost k less than that of graph G. So, max flow of $H - X$ is atleast $f - k$. □

Claim 5. *For any $X \subseteq V$ with $|X| = k$, $r(X) + |X| \geq f$*

Proof. The only edges of $H - X$ incident on t are from t_i.
From **Claim 4**, Max-flow($H - X$) $\geq f - k$ implies that there are $f - k$ vertices $t_i \in T$ from each of which a flow of 1 reaches t. But for a flow to reach from s to t_i, t_i must be reachable from s in $H - X$. Thus number of vertices t_i in $H - X$ reachable from s is atleast $f - k$.

Since the only edges in $H - X$ which are not in $G - X$ are between t_i and t, the number of t_i reachable from s in $G - X$ is also atleast $f - k$.

Thus, $r(X) = |\{t_i : t_i \in G - X \text{ and } t_i \text{ is reachable from } s\}| \geq f - k \implies r(X) + |X| \geq f$ □

Now, given the graph H, we find an X such that equality holds.

Claim 6. *If a vertex is reachable in the residual graph it was reachable in the original graph.*

Proof. We prove this by contradiction. Suppose there exists a set of vertices that is reachable from s in the residual graph H_f but was not reachable in the graph H. Let us consider the first vertex which became reachable. Since it was not reachable initially , it must have become reachable after adding some flow in the Ford-Fulkerson algorithm. But when we add a flow corresponding to an s-t path in an iteration of the algorithm, the forward edges become backward edges in the graph, if this makes a new vertex reachable then it must contain at least one of the vertices from the last path on which flow was added. But the edges after the last occurrence of one of these vertices in the path s-v are the same and also these vertices were initially reachable which contradicts the fact that the vertex v was not reachable earlier. Hence proved. □

Claim 7. *Consider the min cut (A, A') of H obtained from the Ford-Fulkerson Algorithm. Define X_0 to be set of edges in $E(A, A')$ that are also present in G. We claim that $r(X_0) + |X_0| = f$.*

Proof. We partition $E(A, A')$ into 2 sets of edges X_0 and Y_0 as follows. For any Edge $e \in E(A, A')$ if $e \in G$, then $e \in X_0$, otherwise $e \in Y_0$. Since $X_0 \cup Y_0 = E(A, A')$, we have $|X_0| + |Y_0| = |A| = f$, where f is the max-flow of H by the max-flow-min-cut theorem. Thus we have, $|Y_0| = f - |X_0|$

Now we prove that $r(X) = |Y_0|$.

Consider any edge in Y_0. It must be from some $t_i \in A$ to $t \in A'$. By **Claim 6**, t_i is reachable from s in $H - E(A, A')$. So, t_i is also reachable from s in $\{H - E(A, A')\} \cup Y_0 = H - X_0$ as adding edges cannot disconnect a path between vertices. Since there are not edges starting from t in H a path from s to t_i in H cannot pass through t. Thus t_i is reachable from s in $G - X_0$. Thus,

$$|Y_0| \leq r(X_0)$$

Conversely, assume $t_i \in A$ is reachable from s in $G - X_0$. So it is also reachable from s in $H - X_0$ as all edges of $G - X_0$ are also present in $H - X_0$. By construction the edge $e = (t_i, t) \in E(H)$ and $(t_i, t) \notin X$. So, this edge $e \in H - X_0$. Thus t is reachable from s through edge e in $H - X_0$. However $(H - X_0) - Y_0 = H - E(A, A')$ is a cut and the s cannot be reachable from t in it. Thus edge e must also be present in Y_0. Thus,

$$r(X_0) \leq |Y_0|$$

Thus we have $|Y_0| = r(X_0)$

□

Thus we get $r(X_0) = f - |X|$ or equivalently, $r(X_0) + |X_0| = f$. Hence the equality holds and hence the minimum value of $r(X) + X$ is f for $X = X_0$.

1 Max-Flow Min-Cut **20 / 20**

✓ **- 0 pts** **Correct**

- 20 pts No solution

- 5 pts Not correct proof

- 5 pts For not finding set of edges X / X is set of edges in H

- 7 pts No proof of correctness

2 Hitting Set

Q. Consider a set $U = u_1, \ldots, u_n$ of n elements and a collection $A_1, A_2, \ldots, A_m$ of subsets of U. That is, $A_i \subseteq U$, for $i \in [1, m]$. We say that a set $S \subseteq U$ is a hitting-set for the collection $A_1, A_2, \ldots, A_m$ if $S \cap A_i$. is non-empty for each i.

The Hitting-Set Problem (HS) for the input $(U, A_1, \ldots, A_m)$ is to decide if there exists a hitting set $S \subseteq U$ of size at most k.

2.1 Part 1.

Prove that Hitting-Set problem is in NP class.

Ans Given a candidate solution S, we have to check -

- $|S| \leq k \implies$ This can be verified in $\mathcal{O}(n)$ time as $k \leq n$
- $S \cap A_i \neq \phi$, for $i \in [1, m]$. $\implies$. Each A_i and S have atmost n elements. So, for each i, $A_i \cap S$ can be computed in $\mathcal{O}(n^2)$ time by iterating over all pairs $(x, y) x \in A_i and y \in S$. As $i \leq m$, computing all the intersections take $\mathcal{O}(mn^2)$. Thus we verify if each intersection is ϕ in $\mathcal{O}(mn^2)$ which is polynomial in input size n, m.

  Hence the Hitting Set problem is in NP class.

2.2 Part 2

Prove that Hitting Set is NP-complete by reducing Vertex-cover to Hitting Set.

Ans. For an instance $I(G, k)$ of the vertex cover problem which is finding a vertex cover of size at most k with $|V| = n$, $|E| = m$, we convert this to Hitting Set problem I' as follows. Define the set U of I' as the set $V(G)$ which consists of all the vertices (n elements) . For every edge $e_i \in E(G)$, we will construct A_i consisting of the 2 endpoints of the edge e_i. Here, we need to find a hitting Set of size atmost k.

Constructing the set U takes $O(n)$ time as $|V| = n$. Constructing each set A_i we take $O(1)$ time, so for all A_i we take $O(m)$ time. So total time taken to construct the instance I' is $O(m+n)$, polynomial in input size.

Now Let I be a yes-instance of Vertex cover problem. We need to prove that the instance of Hitting Set I' that is mapped from I is a yes-instance of Hitting Set.

Let a solution of $I(G, k)$ be S. Since S is a vertex cover of G, this implies that S contains atleast one end-point of each edge. Now, for each i, A_i contains end points of some edge e. This implies that $S \cap A_i \neq \phi \forall i$. Size of $S \leq$ k as it is a vertex cover of G of size atmost k. Therefore S is a solution of I'.

Now Let $I'(U, A, k)$ be a yes-instance of hitting Set problem. We need to prove that the instance of vertex cover $I(G, k)$ that is mapped to I' is also a yes instance of Vertex Cover problem.

Let a solution of $I'(A, U, k)$ be S. Since S is a hitting set, this implies that $S \cap A_i \neq \phi$. Now, for each edge $e \in E(G)$, $\exists i$ such that A_i contains end points of edge e. So we get that for each edge e, S contains atleast one its endpoints.Size of $S \leq$ k as it is a hiting set of size atmost k Therefore S is a solution of I.

So we have reduced vertex Cover problem to the problem of Hitting Set. Since Vertex Cover Problem NP-complete, Hitting Set is also NP-complete.

2 Hitting Set 20 / 20

- ✓ **+ 5 pts** **HS is in NP**
- ✓ **+ 7 pts** **Poly Time Reduction (VC to HS)**
- ✓ **+ 4 pts** **"Yes" VC => "Yes" HS**
- ✓ **+ 4 pts** **"Yes" HS => "Yes" VC**
- **+ 0 pts** Incorrect

3 Feedback Set

Q. Given an undirected graph $G = (V, E)$, a feedback-set is a set $X \subseteq V$ satisfying that $G - X$ has no cycle. The Undirected Feedback Set Problem (UFS) asks: Given G and k, does there exist a feedback set of size at most k.

3.1 Part 1

Prove that Undirected Feedback Set Problem is in NP class.

Ans. Consider a candidate solution S to an instance of UFS with $|V| = n$ and $|E| = m$.
We check the following

- $|S| \leq k \implies$ This can be computed in O(n) time as $k \leq n$
- $G - S$ has no cycle. We can compute this graph $G_1 = G \setminus S$. by iterating over the vertices of S and removing these vertices from G and those edges from G that have these vertices as an endpoint. There are atmost n vertices of S since $S \subseteq V$ and in total m edges can be removed. Thus it needs $\mathcal{O}(m + n)$ time to compute this graph. Now we check if G_1 has a cycle using DFS over G_1 which again takes $\mathcal{O}(m + n)$.

Thus total time complexity is $\mathcal{O}(m + n) = \mathcal{O}(n^2)$.

We have found a way to verify if any candidate solution to UFS is solves the problem in time polynomial in the input size n. Thus UFS is in NP class.

3.2 Part 2

Prove that Undirected Feedback Set Problem is NP-complete by reducing Vertex-cover to Undirected Feedback Set Problem.

Ans. For an instance $I(G, k)$ of the vertex cover problem with $|V| = n$, $|E| = m$, we convert this to an UFS problem with $I'(H, k)$, where we construct the graph H from G by adding m vertices, one for each edge of G. Let $v(x, y)$ refer to the the vertex corresponding to the edge $x - y$.

Then we add $2m$ edges, 2 from each newly added vertex $v(x, y)$ to x and y. More formally,

$$V(H) = V_1 \cup V_2 \text{ where } V_1 = V \text{ and } V_2 = \{v(x, y) \mid (x, y) \in E\}$$

$$E(H) = E_1 \cup E_2, \text{ where } E_1 = E \text{ and } E_2 = \{(x, v(x, y)) \mid (x, y) \in E\}$$

.

To compute H, we start with graph G and iterate over edges of G. There are m edges of G so need m iterations, and in each iteration, we add 1 vertices and 2 edges to H. Thus we can compute H in time $\mathcal{O}(3m)$. Thus we have mappped each instance $I(G, k)$ of Vertex cover to some instance $I'(H, k)$ of UFS in time $\mathcal{O}(m)$ which is polynomial in the input size.

Now, To prove Vertex-Cover is reducible to UFS we need go through 2 steps. First we prove that for any yes-instance $I(G, k)$ of vertex cover, $I'(H, k)$ is also a yes-instance. Let S be one solution of I.

Claim 1. *Consider S' to be the set of vertices in H corresponding to the vertices of G. For all vertices v of $T = H - S'$ such that $v \in V_1$, all neighbours of v in T are in V_2.*

Proof. If vertex v of $T - S' \in V_1$ then $v \notin S'$ of H and $v \notin S$ of G. As S was a vertex cover, and $v \notin S$, we have $N(v) \subseteq S$. Thus no vertex from vertex set V_1 neighbourhood of v is in $H - S'$, which means that all neighbours of v in $H - S'$ are in V_2 □

Claim 2. *All vertices v of $H - S'$ such that $v \in V_2$ $deg(v) \leq 1$*

Proof. If $v \in V_2$, then $v = v(x, y)$ for some $(x, y) \in E$, and atleast one of x, y must be in S, since it is the vertex cover of graph G. Thus atmost one of x, y can be in $H - S$'. Now the only neighbours of $v(x, y)$ in H are x, y. So, $deg(v) \leq 1$. □

Claim 3. *S' is a UFS of H of size atmost k.*

Proof. Assume that there was a cycle in $T = H - S'$. A cycle must consist of at least 2 vertices. All vertices of the cycle cannot be from V_1 as there should be at least 2 vertices and from Claim 1 we get , the neighbour of any vertex in V_1 belongs to V_2. But a vertex from V_2 cannot belong to a cycle as from **Claim 2**, it has degree atmost 1. This leads to a contradiction. Hence by contradiction, T has no cycle.

Now, we have proved that $T = H - S'$ has no cycle, and $|S'| = |S| \leq k$. So S' is a UFS of size atmost k. Thus $I'(H, k)$ is a yes-instance of UFS. □

Next we prove the other direction. Let $I'(H, k)$ be a yes-instance of UFS, we prove that $I(G, k)$, that was mapped to $I'(H, k)$ is also a yes-instance of vertex cover. Let S be a solution to $I'(H, k)$.

Claim 4. *For each triangle $v_1 = x, v_2 = y, v_3 = v(x, y), (x, y) \in E_1$ in S, alteast one vertex from this triangle is in S.*

Proof. As S is a UFS of H, $H \setminus S$ has no cycles. So, for each such triangle atleast one vertex is not present in $H \setminus S$. Thus atleast one vertex from every such triangle is present in S. □

Claim 5. *There is a vertex cover of G of size atmost k.*

Proof. We construct S' from S.

For each vertex v in S, if $v \in V_2$, say $v(x, y)$ where, replace $v(x, y)$ with x in S. If x is in S, simply remove vertex $v(x, y)$ from S. Let the vertex set thus obtained be S'.

After the exchange of vertices of S to arrive to S' only vertices of V_1 remain in S'. From previous Claim, or each triangle $v_1 = x, v_2 = y, v_3 = v(x, y)$, atleast one vertex is present. But we replace vertices of the form (x,y) with a vertex of that triangle in S'. So for every edge $(x, y) \in E_1$, either x or y is in S'. But $E_1 = E(G)$. So, for every edge of G one of its endpoint vertices must be in S'. Thus S' is a vertex cover of G.

Now, we observe that we replace or remove vertices of S to get S'. So, $|S'| \leq |S| \leq k$. Thus we found S', a vertex cover of G, which implies that $I(G, k)$ is a yes-instance. □

So we have reduced vertex Cover problem to the problem of UFS. Since Vertex Cover Problem NP-complete, UFS is also NP-complete.

3 Feedback Set 20 / 20

- ✓ + **5 pts** **Shown that it is in NP class**
- ✓ + **15 pts** **Correct reduction from vertex cover**
- + **7 pts** Partially correct reduction
- + **13 pts** Used directed feedback set
- + **3 pts** Partially correct NP proof