COL351 Assignment - 4

Aditya (2019CS50471)
Ajay (2019CS10323)

13^{th} November, 2021

Contents

1 Flows and Min-Cuts

We are given a directed graph G = (V, E) with source s and set of terminals $T = \{t_1, t_2, ..., t_k\}$ $\subseteq$ V. For any X $\subseteq$ E, r(X) denote the number of vertices v $\in$ T that remains reachable from s in G - X. Let's create an auxiliary graph H, which is initialized to original graph G. In H, we create a new node t. Then, we add a directed edge (t_i, t) in H $\forall$ $t_i \in T$. Now, we assign capacity of 1 to all edges in H, i.e., c(e) = 1 $\forall$ e $\in$ E(H).

Let f be the max-flow of H (computed using Ford-Fulkerson Algorithm as discussed in the class).

Claim - 1: r(X) + |X| $\geq f$
Proof: We know that if there is a flow of f in a graph with all edge capacities 1, then there exists f edge-disjoint paths from s to t (Claim proved in tutorial). Using this claim, there exists f edge-disjoint paths from s to t in H. From the graph construction, the only incoming edges to t are from the set of terminals T. This implies that, there exists f vertices $T' = \{t'_1, ... ,t'_f\} \subseteq T$ such that the flow value of directed edge between the vertex and t is 1, and for all other vertices in T, the flow value of edge between the vertex and t is 0. As there exists f paths from s to t in H which are edge-disjoint, there exists paths from s to each vertex in T' in H which are edge-disjoint. Let P_1, P_2,..., P_f be the paths, where P_i is path from s to t'_i. All these paths P_1, P_2,..., P_f are edge-disjoint. All the edges in P_i's will be in E(G) (because the only edges that are not in E(G) are those connecting terminals and t). So, these disjoint paths exist in G.

To minimize r(X) + |X|, we need to disconnect more vertices of T, using less number of edges. Let's say we disconnect k' vertices of T' (k' can be 0). So, number of vertices in T connected to s are atleast f - k' (because remaining f - k' vertices of T' will be connected to s, as the paths are disjoint). And, there should be atleast k' edges in X (i.e., there should be atleast one edge in each path from s to k' vertices because all paths are disjoint). Formally, r(x) $\geq f$ - k' and |X| $\geq k'$. So, r(X) + |X| $\geq f$.

Algorithm:
1) Create graph H from G as described above.
2) Find the max-flow f of the graph H, using Ford-Fulkerson Algorithm.
3) Let A be the set of vertices reachable from s in residual graph of H.
4) Let E' be the set of edges in H connecting vertices in A and $\bar{A}$.
5) From E', remove edges that are connected from t_i (any terminal node $\in T$) to t.
6) X is E' i.e., edge set E' minimizes r(X) + $|X|$.
7) return E'.

Proof of Correctness:
Claim - 2: $E' \subseteq E(G)$
Proof: From the algorithm, initially E' is a set of edges in H, that connects A and $\bar{A}$. Now, we are removing edges that are connected from t_i (any terminal node $\in T$) to t. Now, all the edges in E' belongs to $E(G)$, because the only edges in $E(H)$ that are not present in $E(G)$ are those connecting terminals and t. As we are removing those edges from E', all the remaining edges are present in $E(G)$.

Claim - 3: E' minimizes r(X) + $|X|$
Proof: From the algorithm, we first compute max-flow of the graph H, using Ford-Fulkerson Algorithm. Let f be the max-flow. Now, we compute A containing vertices reachable from s in residual graph of H. Number of edges in H connecting vertices in A to A' will be f, because A is the min-cut and all the edges have flow 1 (all the edges in min-cut are saturated). So, initially $|E'|$ = f. Let p vertices of A belong to T. Now, there will be p edges in E' that connect from terminals to t. Now, we remove these edges. So, $|E'|$ = f - p. And, number of terminals reachable from s in G will be p (because A and $\bar{A}$ will be disconnected in G, if we remove E' from G, so the terminals reachable from s should be in A). Formally, r(E') = p. So, r(E') + $|E'| = p + f - p = f$. In claim 1, we proved that minimum value of r(X) + $|X|$ = f. Clearly, E' is giving the minimum value of r(X) + $|X|$. So, the required set of edges is E' and can be obtained from the above algorithm.

Time Complexity:
1) Step 1, takes O(m + n + k) time. (m is number of edges in G, n is number of vertices in G)
2) Step 2, takes O((m + n + k)*f), where f is the max flow of H.
3) Step 3 takes O(m + n + k) time. We can use BFS on residual graph of H.
4) Step 4, 5 takes O(m + k) time.
So, overall time complexity is O((m + n + k)*f). Max flow of H is less than $|T|$, because the incoming flows to t are through vertices of T and the max inflow to t is $|T|$, which is the maximum bound on max flow of H.
So, overall time complexity is O((m + n + k)*$|T|$). Also, n = O(m) and k = O(m). So, overall time complexity is O(m*$|T|$) = O($|E|$*$|T|$).

2 Hitting Set

2.a NP Class

Proof that Hitting Set belongs to NP Class:
Problems which have polynomial time verifier belongs to NP-Class.
We need to prove that Hitting set problem(HSP) has a polynomial time verifier . We are given a set S, $|S| \leq$ k. We can find a intersection of S, A_i in O(nk) time = O(n^2) time. So, For A_1,A_2...A_m we find intersection in O(mn^2) time. If all intersections are non empty then S is a Hitting Set.If not, S is not a hitting set.
So HSP has a verifier of O(mn^2) which is polynomial time of input size. So HSP belongs to NP class

2.b NP Complete

Vertex Cover to Hitting Set

Step 1: Mapping instance of vertex cover to hitting set
Let G = (V,E) be a graph with n vertices, m edges. Let (G, k) be an instance of vertex cover. Consider U be the set of vertices of G i.e., U = V. For each edge e = (x,y), construct a set A_e = {x,y}. So a total of m sets A_1,A_2,......,A_m will be formed and A_1,A_2,......,A_m are subsets of U. So, we constructed HSP = (U,A_1,A_2,......,A_m). And (HSP, k) is an instance of HSP. This mapping takes O(n+m) time, which is polynomial time.

Step 2: G = (V,E) has a vertex cover of size atmost k iff HSP has a hitting set of size atmost k

Part 1:
If HSP has a hitting set of size atmost k then G = (V,E) has a vertex cover of size atmost k
Proof:
let S be a hitting set of HSP, $|S| \leq$ k. So, S $\cap$ A_1 $\neq \phi$ for all A_i. Consider S to be the vertex cover of G. From the above mapping, A_i corresponds to edges. So for each edge e = (x,y), S $\cap$ e $\neq \phi$. So, S is a vertex cover of G. Size of S is atmost k. We found a vertex cover of G = (V,E) of size atmost K. Solution of HSP can be converted to solution of vertex cover in O(1) time.

Part 2:
If G = (V,E) has a vertex cover of size atmost k then HSP has a hitting set of size atmost k
Proof:
let W be vertex cover of G, $|W| \leq$ k. So, for all edges e = (x,y), e $\cap$ W $\neq \phi$ From the mapping edges correspond to A_i'. So, for all A_i, A_i $\cap$ W $\neq \phi$. And size of W is atmost k. So, W is hitting set of HSP. We found a solution of HSP of size atmost k. Solution of Vertex cover can be converted to solution of HSP in O(1) time.

So, from claims in step 1 and 2 we can say that Hitting set is NP-complete.

3 Feedback Set

G = (V,E) has n vertices , m edges

3.a NP Class

Proof that Undirected Feedback set problem belongs to NP Class:
We need to prove that undirected Feedback set problem (UFS) has a polynomial time verifier. We are given a set X of size k. We can construct G - X in O(m + n) time (we need to remove vertices of X in G). We can check for cycle in G - X using DFS algorithm. This takes O(m + n) time. If G - X has no cycles, then X is a feedback set. If not X is not a feedback set. So, UFS has a verifier of O(m + n) time which is polynomial time of input size. So UFS belong to NP class.

3.b NP Complete

Vertex cover to UFS

Step 1: Mapping instance of vertex cover to UFS
let G = (V, E) be a graph with n vertices and m edges and (G, k) is an instance of vertex cover. We construct a graph H = (V',E') from G. Initially define H to be G i.e., H = G. For each edge e = (x,y) in G, we add a vertex v_e to H and connect it to endpoints x and y. $|V'|$ = n + m and $|E'|$ = m + 2m = 3m. This (H, k) is an instance of UFS. This mapping takes O(n + m) time which is polynomial time of input size.

Step 2: G = (V,E) has a vertex cover of size K iff UFS has a feedback set of size k

Part 1:
If UFS has a feedback set of size k, then G = (V,E) has a vertexcover of size K.
Proof:
Let S be a feedback set of UFS and $|S|$ = k. From above mapping x, y, v_e form a cycle in H. So, S must contain a vertex from x, y, v_e
let S contain v_e, e = (x, y) ∈ G. So, any cycle in H containing v_e also contains (x, y). Because v_e is only connected to (x, y). So,we can replace v_e by x or y in S
From above claims, for all edges e = (x, y) ∈ G, S must contain x or y (or both).
So, S is a vertex cover of G of size k. Solution of UFS can be converted to solution of vertex cover in O(1) time.

Part 2:
If G = (V, E) has a vertex cover of size k, then UFS has a feedback set of size k.
Proof:
let W be a vertex cover of G of size k. Now, we remove vertices of W from H (i.e., we remove the vertices in W and edges connected to them). From the mapping, for any edge (x, y) ∈ H , if (x, y) ∈ G then (x, y) ∉ H - W (because either of x, y must be in vertex cover W).
So the only possible edges in H - W are (x, y) such that only one of x, y ∈ G and other is newly added vertex. So any cycle in H - W must contain edges of this type only. Without loss of generality, let x be newly added vertex. So if x is in a cycle then there should be atleast 2 edges incident to x in H - W. From construction, x has only 2 incident edges from u, v and edge(u, v) ∈ G. But atleast one of u, v belong to W. So, atleast one edge incident of x is removed.So, x has atmost 1 edge, incident to it. So, x can't be a part of any cycle. Since every edge in H - W has one newly added vertex no edge can't form a cycle. So, H - W has no cycle. So, W is a feedback set of size k. Solution of vertex cover can be converted to solution of UFS in O(1) time.

Hence, from claims in steps 1 and 2, we can say that UFS is NP-Complete