Name: ANKIT SHUBHAM

Entry number: 2014CS10158

- You may use any of the following known NP-complete problems to show that a given problem is NP-complete: 3-SAT, INDEPENDENT-SET, VERTEX-COVER, SET-COVER, HAMILTONIAN-CYCLE, HAMILTONIAN-PATH, SUBSET-SUM, 3-COLORING, CLIQUE.

There are 5 questions for a total of 100 points.

1. (20 points) Consider the following problem:

   F-SAT: Given a boolean formula in CNF form such that (i) each clause has exactly 3 terms and (ii) each variable appears in at most 3 clauses (including in negated form), determine if the formula is satisfiable.

   Answer the following questions with respect to the above problem under the assumptions (i) $\mathsf{P} = \mathsf{NP}$, and (ii) $\mathsf{P} \neq \mathsf{NP}$. Give reasons.

   (a) Is F-SAT $\in \mathsf{NP}$?

   (b) Is F-SAT NP-complete?

   (c) Is F-SAT NP-hard?

   (d) Is F-SAT $\in \mathsf{P}$?

(ii) $P \neq NP$

(a) F-SAT $\in$ NP as given an assignment of variables, we can check the satisfiability in polynomial time.

(b) Yes. We know 3SAT is NP complete. We try to show 3-SAT $\leq_p$ F-SAT.
3-SAT instance may have single literal clauses and double literal clauses.

(x) {in 3SAT instance} is replaced by $(x, y, z), (x, \bar{y}, z), (x, y, \bar{z}), (x, \bar{y}, \bar{z})$ in F-SAT instance.

x \ yz	00	01	11	10
0	$\boxed{0}$	$\boxed{0}$	$\boxed{0}$	$\boxed{0}$
1	1	1	1	1

Here y and z are new variables.

(x, y) {in 3SAT instance} is replaced by $(x, y, z), (x, y, \bar{z})$ in F-SAT instance.

Here z is new variable.

z \ xy	00	01	11	10
0	$\boxed{0}$	1	1	1
1	$\boxed{0}$	1	1	1

(x,y,z) type of clauses are left as they are, in F-SAT clause.

Now, in the F-SAT instance, we have clauses with exactly 3 literals. We now ~~fix the~~ make sure that each variable occurs in at most 3 clauses.

For every variable x in the 3-SAT instance, let $n(x)$ be the number of occurences of x. If $n(x) \le 3$, then its alright. Else we create $\left(\frac{n(x)}{3}\right)$ new variables, say $x_1, x_2, \dots x_k$ and replace x in the F-SAT instance with these in the boolean expression of F-SAT such that no x_i occur more than 3 times in the F-SAT instance. we also add clauses of type $(x \Leftrightarrow x_i)$ in proper CNF form in F-SAT instance. Now, the Boolean expression satisfies all conditions to be an F-SAT. Let's call the F-SAT instance as E_F and the original 3-SAT instance as E_T.

Claim: E_T is satisfiable iff E_F is satisfiable.

Proof: Suppose E_T is satisfiable. Then each clause has a variable which evaluates to true.

Case 1: If there is a singleton clause of the form (x) in E_T

Here we put $(x,y,z), (x,\bar{y},z), (x,y,\bar{z}), (x,\bar{y},\bar{z})$ instead of (x) in E_F. If E_T is satisfiable then x must be true and so is each of the four clauses in E_F. (Notice that here we have 4 occurences of x in E_F. So this is not the final instance. Refer to case 4 now to see that if (x) is satisfiable then the final replacement of (x) in E_F is satisfiable too.)

Case 2: If there is a clause of form (x,y) in E_T.

Here we put $(x,y,z), (x,y,\bar{z})$ instead of (x,y) in E_F. If E_T is satisfiable, then either of x or y must be true, which makes the two clauses true too.

Case 3: If there is a clause of form (x,y,z) in E_T then if E_T is satisfiable, then (x,y,z) in E_F is true too.

Case 4: If x occurs more than 3 times. Then whatever be the assignment of x makes 3-SAT satisfiable, if we assign same to each of x_i's, then all the clauses with x_i's will be true too.

So, whenever ~~a~~ we pick a clause in E_T, even if we have made more than 1 clauses with corresponding to that in E_F, they all evaluate to true if that picked clause in E_T was ~~sate~~ true. So, E_F is satisfiable if E_T is.

Suppose E_T is not satisfiable. Then there is at least one clause that evaluates to false, whatever be the assignment of variables be.

Case 1: if the clause is (x), then x = false makes the clause unsatisfiable. In the E_F, we have $(x,y,z), (x,\bar{y},\bar{z}), (x,y,\bar{z}), (x,\bar{y},z)$. Whatever be the assignments of y and z (which are new variables), one of the 4 clauses has to be false making E_F unsatisfiable.

Case 2: if the clause is (x,y), then $x=y$ = false makes the clause false. In the E_F, we have $(x,y,z),(x,y,\bar{z})$ and whatever be the value of z be, one of the 2 clauses will be false making E_F unsatisfiable.

Case 3: trivial for clause of the form (x,y,z).

Case 4: If x occurs more than 3 times and suppose one of the clauses in which x occurs, is made unsatisfiable due to $x = b$ (b = true or false). In E_F, we have some x_i instead of x and x_i has to be same as x (due to the clause $(x \Leftrightarrow x_i)$) which again makes that clause in E_F false making E_F unsatisfiable.

So, E_F gets unsatisfiable too.

So, F-SAT is NP-complete.

(iii) F-SAT is NP-hard as every NP-complete problem is NP-hard too

(iv) F-SAT $\in$ P because F-SAT $\in$ NP and P $\subset$ NP.

2\. (15 points) Consider the following problem:

> LONG-PATH: Given a weighted, directed graph $G = (V, E)$, two vertices $s, t \in V$ and a number W, determine if there is a simple path between s and t such that the sum of weights of edges in this path is $\geq W$.

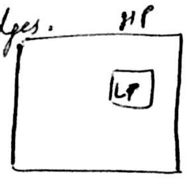
Recall that a simple path is a path that does not have any vertices repeated. Show that LONG-PATH is NP-Complete.

(i) Given a simple path between s and t, sum of weights of edges in this path can be calculated in polynomial time and determine whether it is greater than or equal to W.

So LONG-PATH $\in$ NP

HAMILTONIAN PATH $\leq_p$ LONG-PATH

(ii) I try to prove that HAMILTONIAN PATH $\leq_p$ LONG-PATH

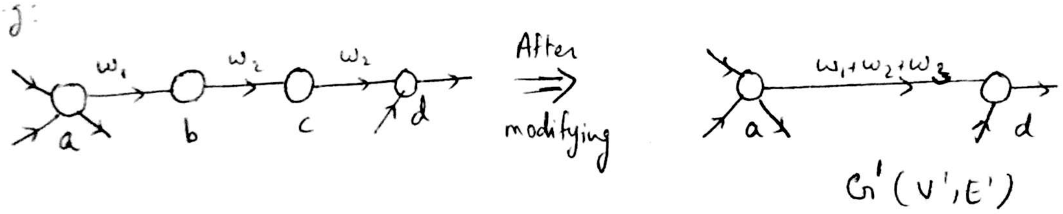


G^o is the original graph where s and t may have incoming and outgoing edges.
Let $G = (V, E)$ be a non weighted, digraph s.t $|V| = W+1$, $s \in V$ has no incoming edge, $t \in V$ has no outgoing edge.
Construct G' from G in the following way:

$G' = (V', E')$ and G' is weighted digraph.

Let $G' = G$ initially and set the weight of each edge to 1.
Now $\forall v \in V'$ such that v has exactly 1 incoming edge and 1 outgoing edge; we do the following: we annihilate v and join the incoming and the outgoing edge and sum up their weights and assign this new weight to this new joined edge.

e.g:



Claim: G has a hamiltonian path iff G' has a simple path between s & t such that sum of weights = W.

Proof: Suppose G has a hamiltonian path P. Then this path must start from s (as its indegree is zero) and end at t (as its outdegree is zero). $|V| = W+1$. Since P covers entire V, it has $|V|-1$ edges, i.e. $(W+1)-1 = W$ edges. Now, we constructed

means weight in graph G is W as each edge was given a wt of 1 initially.

G' in such a manner that $\sum_{G}$ weights $= \sum_{G'}$ weights. Also P is simple, so in the G', P' is also simple. (P' is the path that we obtain from P directly as follows: If the sequence of vertices in path P is $\{v_1 \to v_2 \to v_3 \to \cdots \to v_n\}$ then sequence of vertices in P' is $\{v_{i_1}, v_{i_2}, v_{i_3}, \ldots, v_{i_k}\}$ such that $i_1 < i_2 < i_3 < \cdots < i_k$ and neither of v_{i_j}'s, $j \in [1, k]$ were annihilated.)

Clearly no v_{i_j}'s were repeated in P'. So P' is simple.

Also, $\sum_{G}$ weights $= W$ (initially each edge has weight 1 in G and num of edges ~~are~~ is W)

$\sum_{G}$ weights $= \sum_{G'}$ weights $= W$. So forward direction is proved.

Suppose G' has a simple path P' between s and t such that sum of wts is W.
Consider any adjacents nodes in G' u and v. Let $w(u,v) = w_{u,v}$.
By construction of G', there must be $(w_{u,v} - 1)$ nodes b/w u and v in G (Here wlog, assume the direction of edge (u,v) is from u to v) and the edges intermediary to them must be having the same direction as (u,v) in G.
So, number of nodes generated $= w_{u,v} - 1$ and number of intermediate edges $= w_{u,v}$.
If we do this expansion-kind-of operation, then each edge $e \in P'$ with weight $w(e)$ will give $w(e)$ intermediate edges.
~~So~~ Let's write P' as sequence of edges. $P' = \{e_1, e_2, \ldots, e_k\}$ with weights $w(e_i)$.
Also, $\sum_{i=1 \text{ to } k} w(e_i) = W$. Each e_i gives $w(e_i)$ num of single-weighted edges. So, in together, $\sum w(e_i)$ $(= W)$ number of edges (single weighted) are generated.

Now, G has ~~no~~ no. of vertices $= W+1$. Clearly the above generated edges ~~form a~~ along with the endpoints form a subgraph of G (by construction of G'). As calculated, number of edges in " is W, so no. of vertices in the subgraph $= W+1$ as the subgraph is just a linear chain. But no. of vertices in $G = W+1$. It means this subgraph's ~~edges~~ covers entire vertex space of G. As the subgraph was just a linear chain, no vertex in that subgraph was visited twice. So, it is a hamiltonian path.

(~~Claim 1: If G~~) This was done for only sum of weights $= W$. For sum of weights $\geq W$, the problem could be fed into black box that solves LONG PATH for sum of weights $> W$ and $\leq$ max sum weight. This is polynomial times of call. So above proof of NP completeness still hold.

4. (25 points) Consider the following problem:

SPAN-TREE: Given an undirected graph $G = (V, E)$, determine if there is a spanning tree of G that has at most 10 leaves.

Either show that this problem is NP-complete or give a polynomial time algorithm (along with correctness proof and running time).

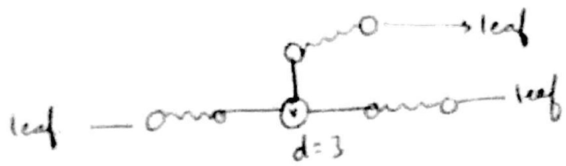
This problem is NP-complete.

(i) Given a spanning tree of G, it can be checked in polynomial time if it has at most 10 leaves. (by simple tree traversal $O(n+m)$ algorithm)

(ii) Claim 1: ~~If~~ G has a hamiltonian path iff G has a spanning tree with 2 leaves.

Proof: Suppose G has a hamiltonian path P. Then this path P (i) covers all vertices in G and (ii) since it is a chain, the endpoints have degree 1 ~~and on~~, that is the 2 endpoints are leaves.
So this P is actually a spanning tree with 2 leaves.

Suppose G has a spanning tree, S with 2 leaves.
Claim 1.1: S has ~~ha~~ all nodes (except the 2 leaves) with degree 2.
Proof: For the sake of contradiction, assume there is a node, v with degree $d > 2$. ~~Since v is in S and is not a leaf~~ So, there are 'd' edges sprouting out of ~~d~~ v. Since there are no cycles in a tree, these d edges have to terminate at some leaf. So, at least d leaves

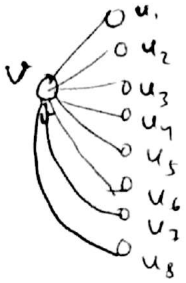


should be present (d leaves exactly would be present if those d edges further don't divide at some nodes in their respective chain). But S has only 2 leaves and $2 < d$. This is a contradiction. Hence claim 1.1 holds.
Note that I didn't take the case when $d=1$ as it directly

implies that v is another leaf, contradicting that there are only 2 leaves.

So, S has all nodes except leaves with degree being 2. This implies that S is a linear chain of nodes or a path that covers entire vertex space of G; implying that S is a hamiltonian path.

Let's create a graph G' in following way. Pick a vertex $v \in V$ and add 8 new nodes to it via 8 edges.



Let $V' = V \cup \{u_i \mid i \in [1,8]\}$, $E' = E \cup \{(v, u_i) \mid i \in [1,8]\}$ and $G' = G'(V', E')$ is the new graph.

Claim 2: G has a hamiltonian path iff G' has a spanning tree with at most 10 leaves

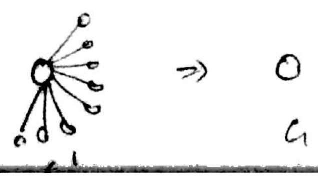
Suppose G has a hamiltonian path P. Then this P is also a spanning tree with 2 leaves for G. Consider the set of edges $(E' - E)$ and set of vertices $(V' - V)$ and include them in P to form a tree P'. Clearly P' is such that $V(P') = V'$ and $E(P') = E(P) \cup (E' - E)$. Clearly P' is a spanning tree of G' as P' covers entire V'. Since $|E' - E| = 8$ new pendant edges were added, P' has $2 + 8 = 10$ leaves. So G' has a spanning tree with 10 leaves always.

Suppose G' has a spanning tree with at most 10 leaves.

Claim 2.1: G' can't have a spanning tree with 7 or less leaves.

Proof: G' has at least 8 leaves (all the u_i's). These 8 leaves can't be spanned with a spanning tree ~~less~~ having less than 8 leaves, as all the leaf-nodes ~~appea~~ in G' appear as leaf nodes in the spanning tree.

Case 1: G' has a spanning tree with 8 leaves. Then G is just a single node graph and then hamiltonian path trivially exist for G.



Case 2: G' has a spanning tree with 9 leaves or 10 leaves. Then remove the u_i's from the spanning tree. Clearly all the u_i's were the leaf nodes in the spanning tree. Removing them leaves ~~us~~ with a tree with at most 2 leaves.

A tree with 1 leaf means a single node; so it is trivially a hamiltonian path.

A tree with 2 leaf means it is a chain and also which covers all nodes in G. So it is hamiltonian path in G

Hence Hamiltonian path $\leq_p$ SPANTREE.

So SPAN-TREE is NP complete.

5. (25 points) A cycle cover of a given directed graph $G = (V, E)$ is a set of vertex disjoint cycles that cover all the vertices. Consider the following problem:

CYCLE-COVER: Given a directed graph $G = (V, E)$, determine if there is a cycle cover of G.

Either show that this problem is NP-complete or give a polynomial time algorithm (along with correctness proof and running time).

The problem is not NP-complete.

Algorithm:

- for all $v \in V$, let there be two vertices v_L & v_R. The set of all v_L's be called V_L and that of v_R's be called V_R
- for all $(u,v) \in E$, let there be an edge (u_L, v_R). The set of all such edges be called E'.
- return MaximumMatching (V_L, V_R, E')

// i.e., for the bipartite graph (V_L, V_R, E'), if there exists a maximum matching, then there exists a cycle cover of G, else not.

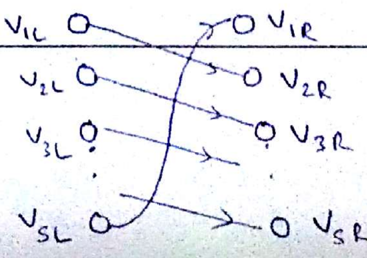
Running time

- Step 1 takes $O(|V|)$ times
- Step 2 takes $O(|E|)$ times
- Using Network flow, step 3 takes $O(|V||E|)$ time using Ford Fulkerson method.
- So running time is $O(|V||E|)$.

Proof of correctness

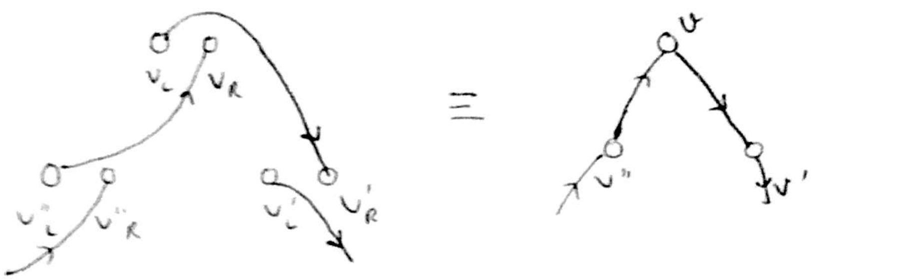
Claim: There is a cycle cover ~~iff~~ for G iff there is a maximum matching in $G' = (V_L, V_R, E')$.

Proof: Suppose there is a cycle cover $C = \{C_1, C_2, \dots, C_k\}$ for G, where C_i is a set of vertices present in i^{th} cycle. Let $C_i = \{v_1, v_2, \dots, v_s\}$. W.L.O.G., assume the cycle is like: $v_1 \to v_2 \to \dots \to v_s \to v_1$. In G', for C_i, we have the subgraph ~~is~~ like $V_{jL} \to V_{(j+1)R}$ for $j \in [1, s-1]$ and $v_{sL} \to v_{1R}$. Clearly this is a maximum matching.



This was for some $C_i \in C$. This can be done analogously for every element of C and we get a maximum matching in every cycle. So overall, for G, we get a maximum matching too.

($\Leftarrow$) Suppose there is a maximum matching in G'. Let the edge set in the max matching be E''.
It means for every $u_L \in V_L$, there is a matching to $v_R \in V_R$;
~~Since~~ so for every vertex v, it has an incoming and outgoing edge because in G', v_L is matched to some v'_R, which means that there is an outgoing edge from v to v' in G. Similarly v_R is matched from som v''_R which means that there is v_R in G which has an incoming edge towards v in G



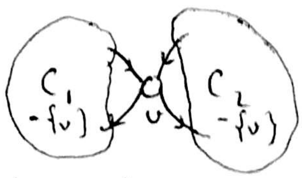
Also, every vertex $v \in G$ there is only one incoming and one outgoing edges, where the edges $\in E''$. It is because every vertex in V_L G'', after max matching, is matched with only one vertex in V_R. ~~So every~~

Claim: There is atleast one cycle in $G''(V, E'')$.

Proof: For the sake of contradiction, suppose there are not any cycle in $G''(V, E'')$. Then there is a node v, which indegree is zero. But it would mean that, there is no edges towards v_R ~~and~~ in G' and hence no maximum matching is possible. Hence contradiction arises and claim stands

Claim: G'' consists of disjoint cycles.

Proof: Every $v \in G$ is part of some cycle in $G''(V, E'')$ because for every $v \in G$, there exist v' and v'' s.t. in G', v_L is matched to v_R' and v_L'' is matched to v_R. So there is no v which indegree or outdegree is zero in G''.
Thus G'' consists of cycles and no acyclic components.
Now we prove that all cycles are vertex-disjoint in G''.
For the sake of contradiction, assume that cycle C_1 and C_2 have a common vertex v, i.e., $C_1 \cap C_2 = \{v\}$.
It implies that v has 2 incoming and 2 outgoing edges, 1 set of incoming and outgoing edges from some vertices in C_1 and other from C_2.



So, v_L has two edges going out of it. But it is not possible as edges are those from maximum matching set E'' and every $v_L' \in V_L$ has only one edge going out from it.
Similar argument holds for v_R.
So its a contradiction that $C_1 \cap C_2 = \{v\}$. So every cycle is vertex disjoint.

Claim: The set of vertex disjoint cycles cover the entire vertex space of G.

Proof: Note that $V(G) = V(G'')$. ~~Since every $v \in G$ is part of~~
And from claim that G'' consists of disjoint cycles, the claim that the set of vertex disjoint cycle cover the $V(G) (= V(G''))$ holds.

Scanned by CamScanner