ECE 606, Fall 2019, Assignment 11

Zhijie Wang, Student ID number: 20856733

zhijie.wang@uwaterloo.ca

November 26, 2019

1. *Proof.*

   For the **NP**-hard part, we reduce from VertexCover. Given an instance $< G =< V, E >, k >$ of VertexCover, we output our instance of proposed problem DistinctSkills, $< |E|, |V|, k >$, where every vertex in V represent an application, and every edge in E represent a skill. Hence, each applicant have a skill if and only if the corresponding edge incident on the corresponding vertex. Then we need to prove VertexCover $\leq_k$ DistinctSkills.

   For the only-if direction, suppose G has a vertex cover of size k, then, there must exist a subset of applicants of size $\leq k$ for DistinctSkills because all the edges which represent skills are incident of the vertex cover. For the if direction, suppose G doesn't have a vertex cover of size k, then there exists a edge which is not incident on any vertexes of a vertex set which has a size $\leq k$, then, there must exists a skill which is not included.

   For the **NP** part, given a true instance $< n, m, k >$, we could adopt as certificate $< r >$, where r is a subset of m applications. Our verification algorithm first check that if all n skills are included in these applications, then check if the size of the subset is $\leq k$. The certificate is linear in the size of the instance, and the verification algorithm is at worst polynomial-time.

   Therefore, DistinctSkills is a **NP**-complete problem.

2. *Proof.*

   For the **NP**-hard part, we reduce from IndependentSet. Given a instance $< G =< V, E >, k >$ of IndependentSet, we output our instance for DisjointPath as following. Let's said our new graph H. For every edge $e_i \in E$, we give H a vertex q_i. Then, for every vertex $v_j \in E$, we create a path $p_j \in P$ which constructed by $q_n \to \cdots \to q_m$, where $e_n, \ldots, e_m$ are edges incident on v_j. Also, we give all these paths' edges to H. Hence, we output $< H, P, k >$ as our instance of DisjointPath. Then we need to prove IndependentSet $\leq_k$ DisjointPath.

   For the only-if direction, suppose G has a independent set with a size of k, then, any two of these k vertexes don't have a edge, therefore, there must be k paths in P that no vertex appears in more than one path. For the if direction, suppose G doesn't have a independent set with a size of k, then, for any subset of V with size of k, there must be two distinct vertexes have a edge, let's said e', incident on both of them. Therefore, there must be two paths in H share one vertex q' which is corresponding to e'.

   For the **NP** part, given a true instance $< G, P, k >$, we could adopt as certificate $< r >$, where r is a subset of P. Our verification algorithm first check that if no vertex appears more than one time in paths of r, then check if the size of the subset r is k. The certificate is linear in the size of the instance, and the verification algorithm is at worst polynomial-time.

   Therefore, DisjointPath is a **NP**-complete problem.

3. *Proof.*

   For the **NP**-hard part, we reduce from HamCycle. Given an instance $< G =< V, E >>$ of HamCycle, we first turn all the edges in G into two edges with different directions, then, for a vertex $a \in V$, we weight all the edge incident on a as $(|V| - 2)/2$, and -1 for all the other edges. Let's say this new graph $< H =< V, E' >, w >$, which is our instance for ZeroWeightCycle. Then we need to prove HamCycle $\leq_k$ ZeroWeightCycle.

   For the only-if direction, suppose G has a hamiltonian cycle, then, there must exists a zero weight cycle for H because the hamiltonian cycle in G must have vertex a, therefore this cycle in H must have 2 edges weighted as $(|V|-2)/2$ and $(|V|-2)$ edges weighted as -1, which could construct a zero weight cycle. For the if direction, suppose G doesn't have a hamiltonian cycle, for any cycle in H, here we do a case analysis. 1) If this cycle include vertex a, it is zero weight cycle if and only if this cycle has another $(|V|-2)$ edges weighted as -1, then, for a simple cycle it must have another $|V|-1$ vertexes, which means this is a hamiltonian cycle. Hence, this is a contradiction. 2) If this cycle doesn't include vertex a, then, it cannot be zero weight cycle because any cycle would have a weight < 0.

For the **NP** part, given a true instance $< G =< V, E >, w >$, we could adopt as certificate $< m >$, where m is a cycle in G. Our verification algorithm will check that if cycle is zero-weight. The certificate is linear in the size of the instance, and the verification algorithm is at worst polynomial-time.

Therefore, ZeroWeightCycle is a **NP**-complete problem.

4. *Proof.*

For the **NP**-hard part, we reduce from VertexCover. Given an instance $< G =< V, E >, k >$ of VertexCover, for $e_m \in E$, where $m \in \{1, \ldots, |E|\}$, we introduce a new vertex q_m, then connect it to v_i, v_j, where $e_m =< v_i, v_j >$. Let's say this new graph $H =< V', E' >$, and our instance for NoCycles is $< H, k >$. Then we need to prove VertexCover $\leq_k$ NoCycles.

For the only-if direction, suppose G have a vertex cover with a size of k, then, if we remove all the vertexes and edges incident on them in this vertex cover in H, all the edges of G has been removed. We notice that every $< q_m, v_i, v_j >$ is a cycle, however, $e_m =< v_i, v_j >$ has been removed, therefore, there must be no cycles in H after removal. For the if direction, suppose G doesn't have a vertex cover with a size of k, then, we remove any subset of vertexes with a size $< k$, there must exists a edge $e_m =< v_i, v_j >$ from G, then, there must exists a cycle $< q_m, v_i, v_j >$.

For the **NP** part, given a true instance $< G =< V, E >, k >$, we could adopt as certificate $< X >$, where X is a set $\subset V$. Our verification algorithm will first check that if the size of X is k, then check if remove all the vertexes and edges incident on vertexes in X, there is still a cycle in G. The certificate is linear in the size of the instance, and the verification algorithm is at worst polynomial-time.

Therefore, NoCycles is a **NP**-complete problem.