# Homework 10

Jiahao Liu April 8, 2024

## 1. Clique and Dense Subgraph Problem

**a) Given a graph G, and a number $k \geq 0$, does G have a clique of size k, where a clique is a subset of vertices such that every two distinct vertices in the subset are adjacent. Show that this problem is NP-complete.**
**First show that the Clique problem is in NP:** Certificate: A subset of vertices of size k.
Certifier: Check the subset contains k distinct vertices and any two vertice in the subset are adjacent, i.e. there is an edge between them in G. This can be done in polynoimal time $O(k^2)$.
**Claim: Independent Set $\leq_p$ Clique Problem**
Start with any Independent Set problem G = (V, E), because we want to get a set of non-adjacent nodes in G by using the adjacent property of Clique, we have to construct a new graph G' = (V, E') where E' = (V x V) - E, i.e. there is an edge between two vertices in G' if and only if there is no edge between them in G. Now we need to prove that there is an Independent Set of size k in G iff there is a clique of size k in G'.

*Proof.* **If there is an Independent set of size k in G, then there is a clique of size k in G'.**
Becuase there is no edge between any two vertices in the Independent Set in G, there must be an edge between any two vertices in this set in G'. We get a Clique of size k.
**If there is a clique of size k in G', then there is an Independent set of size k in G.**
Because there is an edge between any two vertices in the Clique in G', there must be no edge between any two vertices in this set in G. We get an Independent Set of size k. □

**b) Dense Subgraph Problem: Given graph G = (V, E), numbers $k, m \geq 0$. Does there exist a subgraph G' = (V', E') of G, such that V' has at most k vertices and E' has at least m edges. Prove that the Dense Subgraph Problem is NP-Complete.**
**First show that the Dense Subgraph problem is in NP:**
Certificate: A subgraph of at most k vertices and at least m edges.
Certifier: Check the subgraph has at most k vertices and at least m edges. This can be done in polynoimal time O(k + m).
**Claim: Clique $\leq_p$ Dense Subgraph Problem**
Given a Blackbox that solves Dense Subgraph Problem, we can easily get if there is a clique of size k in G by calling the Dense Subgraph Blackbox to check if there is a subgraph of at most k vertices and at least k(k-1)/2 edges in G.

*Proof.* **Suppose we have a clique of size k in G,** then because every two vertices in G are adjacent, so there is a subgraph of at most k vertices and at least k(k-1)/2 edges in G. **Suppose there is a subgraph of at most k vertices and at least k(k-1)/2 edges in G,** then this subgraph must have k vertices and k(k-1)/2 edges, i.e. every two vertices in this subgraph are adjacent, so we get a clique of size k. □

## 2. Set Packing Problem

**There are n courses at USC, each of them scheduled in multiple disjoint time intervals. For example, a course may require the time from 9am to 11am and 2pm to 3pm and 4pm to 5pm (you can assume that there is a fixed set of possible intervals). You want to know, given n courses with their respective intervals, and a number K, whether it's possible to take at least K courses with no two overlapping (two courses overlap if they have at least one common time slot). Prove that the problem is NP-complete.**
This problem is just the same as Set Packing Problem talked in class. The same prove can be applied here.
**First show that the Course Selection Problem is in NP:**
Certificate: A set of K courses with no two overlapping.
Certifier: Check the set contains K distinct courses and any two courses in the set have no overlapping time slots. This can be done in polynoimal time.
**Claim: Independent Set $\leq_p$ The Course Scheduling Problem**
Start any instance of Independent Set problem in G, we can consider each node as a course $C_u$ and its connected edges as the lecture intervals $T_e$, and $C_u$ consist of all intervals $T_e$ such that edge $e$ is incident on node $u$. Then there is an Independent Set of size K in G iff there is K courses with no two overlapping.

*Proof.* **If there is an Independent Set of size K in G, then there is a set of K courses with no two overlapping.**
Because the K nodes in G that are not connected by any edge, the courses corresponding to these K nodes must have no overlapping time slots. We get a set of K courses with no two overlapping.
**If there is a set of K courses with no two overlapping, then there is an Independent Set of size K in G.**

Because there is no overlapping between any two courses in the set, just choose the corresponding nodes for these courses and there must be no edge connecting these nodes. We get an Independent Set of size K. □

## 3. 3-SAT(15/16) Problem

**Consider the partial satisfiability problem, denoted as $3SAT(\alpha)$ defined with a fixed parameter $\alpha$ where $0 \leq \alpha \leq 1$. As input, we are given a collection of k clauses, each of which contains exactly three literals (i.e. the same input as the 3-SAT problem from lecture). The goal is to determine whether there is an assignment of true/false values to the literals such that at least $\alpha k$ clauses will be true. Note that for $\alpha = 1$, we require all k clauses to be true, thus 3-Sat(1) is exactly the regular 3-SAT problem. Prove that 3-Sat(15/16)is NP-complete First show that the 3SAT(15/16) problem is in NP:**
Certificate: An assignment of true/false values to the literals such that at least $15k/16$ clauses will be true.
Certifier: Check the assignment satisfies at least $15k/16$ clauses. This can be done in polynoimal time.
**Claim: 3SAT $\leq_p$ 3SAT(15/16) Problem**
Given an instance of 3SAT with k clauses, we can construct a 3SAT(15/16) problem with 16*k clauses such that there is a satisfying assignment of 3SAT iff there is a satisfying assignment of 3SAT(15/16), i.e. at least 15*k clauses are true. For each clause $(x \vee y \vee z)$ in 3SAT, we can construct 16 clauses as follows: $(x \vee y \vee z) \wedge (\bar{x} \vee y \vee z) \wedge (x \vee \bar{y} \vee z) \wedge (x \vee y \vee \bar{z}) \wedge (\bar{x} \vee \bar{y} \vee z) \wedge (\bar{x} \vee y \vee \bar{z}) \wedge (x \vee \bar{y} \vee \bar{z}) \wedge (\bar{x} \vee \bar{y} \vee \bar{z}) \wedge 8 * (x \vee y \vee z)$.
If k % 8 = 0, we can add 3 new dummy variables for each set of 8 clauses, and create all the 8 possible clauses on these 3 new variables. In this way, we can add k new clauses to the original k clauses. And any assignment of these 3 dummy variables will satisfy only 7/8 of the new clauses. So all of the original k clauses in 3SAT can be satisfied if and only if 15/16 of all 2*k clauses in 3SAT(15/16) can be satisfied.
If k = 8m + b, b < 8. we can still add 3*m new dummy variables for m sets of 8 clauses with 8*m new clauses, leading to 16m + b clauses in total. Here the 3SAT(15/16) is satisfied with at least $\lceil 15m + 15b/16 \rceil$. Remember exactly 7*m of the dummy clauses can be satisfied. The original k clauses must be satisfied with at least $\lceil 8m + 15b/16 \rceil = 8m + b$. So the 3SAT(15/16) is satisfied with at least 15/16 of all clauses if and only if the original 3SAT is satisfied.

*Proof.* **If there is a satisfying assignment of 3SAT, then there is a satisfying assignment of 3SAT(15/16) with at least 15*k clauses are true.** Because if the $(x, y, z)$ is true in 3SAT, then only $(\bar{x}, \bar{y}, \bar{z})$ in 3SAT(15/16) is false, thus we can get 15 clauses are true for each group, thus 15*k clauses are true in total.
**If there is a satisfying assignment of 3SAT(15/16) with at least 15*k clauses are true, then there is a satisfying assignment of 3SAT.** Because if there are 15*k clauses are true in 3SAT(15/16), then $(x, y, z)$ in each group must be true, thus we can get a satisfying assignment of 3SAT. □

## 4. Vertex Cover on Even Degree Graph Problem

**Consider the vertex cover problem that restricts the input graphs to be those where all vertices have even degree. Call this problem VC-EDG. Show that VC-EDG is NP-complete.**
**First show that the VC-EDG problem is in NP:**
Certificate: A set of vertices that covers all edges.
Certifier: Check the set contains distinct vertices and covers all edges. This can be done in polynoimal time.
**Claim: Vertex Cover $\leq_p$ VC-EDG**
Start with any instance of Vertex Cover problem in G, we can try to construct a new graph G' with all vertices have even degree, thus convert the Vertex Cover problem to VC-EDG problem.
Let (G = (V, E), k) to be an input instance of Vertex Cover. Because each edge in E contributes a count of 1 to the degree of each of the vertices with which it connects, the sum of the degrees of the vertices is exactly 2|E|, an even number. Hence, there is an even number of vertices in G that have odd degrees. We can add a triangle with vertices $v, w_1, w_2$ with all the odd-degree vertices in the original graph G are connected to v. Precisely, let U be the subset of V with odd degrees in G, $G' = (V \cup \{v, w_1, w_2\}, E \cup \{(v, w_1), (v, w_2), (w_1, w_2)\} \cup \{(v, u) | u \in U\})$
Now all vertices in G' have even degree. Finally, since a vertex cover for a triangle is of minimum size is 2, there is a vertex cover of size b in G if and only if there is a vertex cover of size b+2 in G'.

*Proof.* **If we have a vertex cover of size b in G, then we have a vertex cover of size b+2 in G'.**
Just add v and $w_1$ to the cover of G, then we get a cover of size b+2 in G'.
**If we have a vertex cover of size b+2 in G', then we have a vertex cover of size b in G.**
Consider a vertex cover of G' does not include v, then it must include both $w_1$ and $w_2$ otherwise some triangle edge would be uncovered. If we take $w_2$ out of the cover and replace it with v then, since the triangle vertices remain covered, we still have a vertex cover. Thus any vertex cover of G' either includes v or, if it doesn't include v, it can be changed into a same-size vertex cover that does include v. So just remove v and $w_1$ from the cover of G' and we can get a cover of size b of G.
(Note: only add one vertex $v$ is not enough, because we can't directly find a vertex cover of size b in G given a vertex cover of size b+1 in G', since added vertex v may NOT be included in the given vertex cover of G'.) □

## 5. Transitivity of Polynomial Time Reduction

**Let S be an NP-complete problem, and Q and R be two problems whose classification is unknown (i.e. we don't know whether they are in NP, or NP-hard, etc.). We do know that Q is polynomial time reducible to S and S is polynomial time reducible to R. Mark the following statements True or False based only on the given information, and explain why.**

Because $Q \leq_p S \leq_p R$ and S is NP-complete, thus Q could in P or NP-Intermediate, and R is NP-hard.

**(i) Q is NP-complete (False)**
**(ii) Q is NP-hard (False)**
**(iii) R is NP-complete (False)**
**(iv) R is NP-hard (True)**