Homework 10

1 Graded Problems

1. True/False Questions (9pt)
   State True or False for the following sentences and give a brief explanation.

   (a) If someone proves P=NP, then it would imply that every decision problem can be solved in polynomial time.

   (b) Assume A is a decision problem, If $A \leq_p B$ and $B \in$ NP, then $A \in$ NP.

   (c) Assume $P \notin NP$. Let A and B be decision problems. If $A \in NPC$ and $A \leq_p B$, then $B \in P$.

   (a) **False**. Halting is a counter example.

   (b) **True**. The important observation is that you can construct a certifier for A by composing the polynomial time reduction map and the certifier for B. A proof sketch for Karp reductions follows if you are interested.

   Proof: $B \in NP$ implies there exists a polynomial time certifier C_B. That is, there exists a constant k such that, if x is a "no" instance for B then $\forall c, C_B(x, c) = 0$ and if $x \in B$ then there exists a c_x such that $||cx|| \leq ||x||^k$ and $C_B(x, cx) = 1$. Here, $||y||$ denotes the number of bits used to write y.

   Let f be a polynomial time reduction from A to B with running time bounded by a polynomial of degree d. That is, f is a polynomial time algorithm that maps instances of A to instances of B, such that x is a "yes" instance of A if and only if $f(x)$ is a "yes" instance of B. Compose the algorithms C_B and f to obtain the polynomial time algorithm C_A, that is

   $$\forall x, \forall c, C_A(x, c) := C_B(f(x), c)$$

   We claim that C_A is a polynomial time certifier for A with certificate size bounded by kd bits.

   (c) **False**. If B were in P, then $A \leq_p B$ would imply $A \in P$. Since $A \in NPC$, $\forall D \in NP, D \leq_p A$. Since $A \leq_p B$, this implies $\forall D \in NP, D \in P$ which contradicts $P \notin NP$.

   Rubric (3pt for each):

   - 2 pt: True or False.
   - 1 pt: Reasonable explanation.

2. Show that vertex cover remains NP-Complete even if the instances are restricted to graphs with only even degree vertices. (15pts)
   First we need to prove even degree vertex cover problem is in NP.

   The certifier takes an undirected graph (the VC instance) and a set of vertices as input. It verifies the union of all edges belong to the each vertex cover all the original graph's edges. Thus problem is in NP.(the same certifier to the original vertex cover problem)

   Then we need to prove finding the even degree vertex cover is NP-Hard.

We claim that vertex cover with only even degree vertices remains NP-Complete by showing it is polynomial time reducible from the original vertex cover 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.
Let U be the subset of vertices with odd degrees in G.
Construct a new instance $(\overline{G} = (V_0,\ E_0),\ k{+}2)$ of Vertex Cover, where $V_0 = V \cup \{x, y, z\}$ and $E_0 = E \cup \{(x, y), (y, z), (z, x)\} \cup \{(x, v) | v \subset U\}$. That is, we make a triangle with three new vertices, and then connect one of them (say x) to all the vertices in U. The degree of every vertex in V_0 is even. Since a vertex cover for a triangle is of (minimum) size 2, it is clear that V_0 has a vertex cover of size k + 2 if and only if G has a vertex cover of size k.
Hence, vertex cover with only even degree vertices is NP Complete.

Rubric (15pt):

- 6 pt: state the problem is in NP and mention how to verify it in polynomial time by checking if it is a valid vertex cover with even degree.
- 3 pt: Choose a NP-complete problem and say that we need to prove that chosen problem $\leq_c$ even degree vertex cover. Other reduction can also get 3 points. Note: wrong direction will get 0.
- 6 pt: Showing the polynomial reduction with details and its proof.

3. Consider the partial satisfiability problem, denoted as 3-Sat(α). We are given a collection of k clauses, each of which contains exactly three literals, and we are asked to determine whether there is an assignment of true/false values to the literals such that at least αk clauses will be true. Note that 3-Sat(1) is exactly the 3-SAT problem from lecture.
Prove that 3-Sat(15/16)is NP-complete. (20 points)
Hint: If x, y, and z are literals, there are eight possible clauses containing them:

$$(x \vee y \vee z), (!x \vee y \vee z), (x \vee !y \vee z), (x \vee y \vee !z), (!x \vee !y \vee z), (!x \vee y \vee !z), (x \vee !y \vee !z), (!x \vee !y \vee !z)$$

To prove it's in NP: given a truth value assignment, we can count how many clauses are satisfied and compare it to 15k / 16.
To prove it's NP-hard:
We will show that 3-SAT p 3-SAT(15/16). For each set of 8 original clauses, create 8 new clauses using 3 new variables, and construct the clauses considering the collection of all possible clauses on the 3 new variables. If the number of clauses is a multiple of 8, then we are done: any assignment will satisfy 7/8 of the new clauses, so we must satisfy all of the original clauses in a valid solution to satisfy exactly 15/16 of the clauses. If the number of clauses is not a multiple of 8, we will satisfy more than 15/16 of the clauses, but if even one of the original clauses is not satisfied, we will have satisfied less than 15/16 of the clauses.
So, 3-Sat(15/16) is in the intersection of NP and NP-hard which is the class NP-Complete.
Example: If original given formula is
$(a \vee b \vee c) \wedge (!a \vee b \vee c) \wedge (a \vee !b \vee c) \wedge (a \vee b \vee !c) \wedge (!a \vee !b \vee c) \wedge (d \vee e \vee f) \wedge (g \vee !b \vee !c) \wedge (!a \vee !h \vee !c)$
So we add our 8 new clauses so that formula now contains total 16 clauses.i.e. :
$(a \vee b \vee c) \wedge (!a \vee b \vee c) \wedge (a \vee !b \vee c) \wedge (a \vee b \vee !c) \wedge (!a \vee !b \vee c) \wedge (d \vee e \vee f) \wedge (g \vee !b \vee !c) \wedge (!a \vee !h \vee !c) \wedge (x \vee y \vee z) \wedge (!x \vee y \vee z) \wedge (x \vee !y \vee z) \wedge (x \vee y \vee !z) \wedge (!x \vee !y \vee z) \wedge (!x \vee y \vee !z) \wedge (x \vee !y \vee !z) \wedge (!x \vee !y \vee !z)$

And as described above there will be 15/16 satisfied clauses. Note we add for every 8 original clauses, new 8 clauses. But if original number of clauses in a formula is not a multiple of 8, then there will be more than 15/16 clauses satisfied. Which is okay for us, since we want AT LEAST 15/16.

Rubric (15pt):

- 5 pt: state the problem is in NP and mention how to verify it in polynomial time by checking if it is a valid vertex cover with even degree.
- 5 pt: Choose a NP-complete problem and say that we need to prove that chosen problem $\leq_c$ 3-SAT (15/16). Other reduction can also get 3 points. Note: wrong direction will get 0.
- 10 pt: Showing the polynomial reduction with details and its proof including:
  5 pt: instance construction 5 pt: using the construction proof the polynomial reduction.

4. Given a graph $G = (V, E)$ and two integers k, m, the *Dense Subgraph* Problem is to find a subset V' of V, whose size is at most k and are connected by at least m edges. Prove that the *Dense Subgraph* Problem is NP-Complete.(20 pts)

Proving this problem NP is trivial, So here we show how to prove it's NP-Completeness.
We prove that the Independent set problem $\leq_P$Dense Subgraph Problem. Given a graph G(V, E) and an integer k, an

independent set decision problem outputs yes, if the graph contains an independent set of size k. For an arbitrary graph $G = (V, E)$ of n vertices, we first get the complementary graph G_c of G.
A clique is a subset of vertices of an undirected graph G such that every two distinct vertices in the clique are adjacent; that is, its induced subgraph is complete. We know that a clique will always contain $k(k-1)/2$ edges if there're k vertices in G, and that an independent set in G is a clique in G_c (the complement graph of G) and vice versa.
Then we set m to $k(k-1)/2$ and test with the dense subgraph problem.

Claim: There exists an independent set of size k in G (equivalently, a clique in G_c of size k), iff there exists a subgraph of G_c with at most k vertices and at least $m = k(k-1)/2$ edges. $\rightarrow$) If there exists a clique in G_c of size at least k, then there exists a subgraph of G_c with at most k vertices and at least $k(k-1)/2$ edges.
If there is a clique of size at least k then there is a clique of size exactly k. Moreover, by definition, a clique of size k would have k * (k-1)/2 edges. $\leftarrow$) If there exists a subgraph of G_c with at most k vertices and at least $k(k-1)/2$ edges, then there is a clique of size at least k.
For a subgraph to have $k(k-1)$ edges, implies there are k vertices. So this subset with k vertices forms a clique in G_c of size k.
Example: Consider a graph $G = (V, E)$ where $V = \{1, 2, 3, 4\}$ and $E = \{(1, 2), (2, 3)\}$. To determine if there's an independent set in G of size 3, we first generate G_c, where $E_c = \{(1, 3), (1, 4), (2, 4), (3, 4\}$. Since there's a dense graph with 3 vertices and 3*2/2=3 edges in G_c , which is $V' = \{1, 3, 4\}$, $E' = \{(1, 3), (1, 4), (3, 4)\}$, we can say there's an independent set in G of size 3. On the other hand, if we set $k = 4$ and $m = 4 * 3/2 = 6$, then there's no such dense graph in G_c , thus no independent set of size 4 in G.

Rubric (15pt):

- 3 pt: state the problem is in NP and mention how to verify it in polynomial time by checking if it is a valid vertex cover with even degree.
- 5 pt: Choose a NP-complete problem and say that we need to prove that chosen problem $\leq_c$ Dense Subgraph problem. Other reduction can also get 3 points. Note: wrong direction will get 0.
- 12 pt: Showing the polynomial reduction with details and its proof including:
  4 pt: instance construction 8 pt: proving the claim. Each direction is 4 pts.

2 Ungraded Problems

1. There are N cities, and there are some undirected roads connecting them, so they form an undirected graph $G(V, E)$. You want to know, given K and M, if there exists a subset of cities of size K, and the total number of roads between these cities is larger or equal to M. Prove that the problem is NP-Complete.

   Solution:

   We can formalize the problem as, given an undirected graph $G(V, E)$ and asks for if there exist a subset S of K vertices such that the number of edges in the induced graph $G[S]$ is at least M.

   (a) (Showing Problem in NP, 5 points) The solution of the problem can be easily verified in polynomial time, just check the total number of roads between these cities. Thus it is in NP.

   (b) (Showing Problem in NP-Hard, 15 points) Given a decision version of the independent set problem, suppose it asks for whether an independent set exists of size K. Then we ask whether the complement of the graph has a subset S of K vertices such that number of edges in the induced graph $G[S]$ is at least $K(K-1)/2$. If so, we find an independent set of size K in the original graph. For the other direction, if there is an independent set of size K in the original graph, we can find a subset S of K vertices such that number of edges in the induced graph $G[S]$ is at least $K(K-1)/2$. Thus we reduced the independent set problem to this problem in polynomial time. Since independent set problem is NP-Complete, so this problem is in NP-Hard.

   Thus this problem is NP-Complete.

2. Suppose we have a variation on the 3-SAT problem called Min-3-SAT, where the literals are never negated. Of course, in this case it it possible to satisfy all clauses by simply setting all literals to true. But, we are additionally given a number k, and are asked to determine whether we can satisfy all clauses while setting at most k literals to be true. Prove that Min-3-SAT is NP-Complete.

   Solution:

(a) For a truth assignment, we can simply count the number of literals set to true. Then evaluate each clause with the truth assignment. If all clauses equal to true and at most k literals are set to true, then answer yes. So Min-3-SAT $\in$ NP.

(b) We reduce from vertex cover to Min-3-SAT. For any given instance of the vertex cover problem, we can construct an equivalent Min-3-SAT problem with variables for each vertex of a graph. Each edge (u, v) of the graph can be represented by a clause $(u \vee u \vee v)$ or $(u \vee v \vee v)$ which can be satisfied only by including either u or v among the true variables of the solution. For the constructed Min-3-SAT problem, there is a satisfying assignment within k true variables if and only if there is a vertex cover within k vertices to the corresponding vertex cover problem. Therefore, Min-3-SAT is NP-Hard.

Thus this problem is NP-Complete.