HomeWork 10

Ques 1. 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 3-SAT problem from lecture.

Prove that 3-Sat(15/16) is NP-Complete.

Hint: If x, y, and z are literals, there are eight possible clauses containing them: $(x \vee y \vee z)$, $(!x \vee y \vee z)$, $(x \vee !y \vee z)$, $(x \vee y \vee !z)$, $(!x \vee !y \vee z)$, $(!x \vee y \vee !z)$, $(x \vee !y \vee !z)$, $(!x \vee !y \vee !z)$

Ans. If the original formula is,

$(a \vee b \vee c) \wedge (!a \vee b \vee c) \wedge (a \vee !b \vee c) \wedge (a \vee b \vee !c) \wedge (!a \vee !b \vee c) \wedge (d \vee e \vee f) \wedge (g \vee !b \vee !c) \wedge (!a \vee !b \vee !c)$ — ①

we can add 8 new clauses,

$(x \vee y \vee z)$, $(!x \vee y \vee z)$, $(x \vee !y \vee z)$, $(x \vee y \vee !z)$, $(!x \vee !y \vee z)$, $(!x \vee y \vee !z)$, $(x \vee !y \vee !z)$, $(!x \vee !y \vee !z)$ — ②

$\Rightarrow$ $(a \vee b \vee c) \wedge (!a \vee b \vee c) \wedge (a \vee !b \vee c) \wedge (a \vee b \vee !c) \wedge (!a \vee !b \vee c) \wedge (d \vee e \vee f) \wedge (g \vee !b \vee !c) \wedge (!a \vee !b \vee !c) \wedge (x \vee y \vee z) \wedge (!x \vee y \vee z) \wedge (x \vee !y \vee z) \wedge (x \vee y \vee !z) \wedge (!x \vee !y \vee z) \wedge (!x \vee y \vee !z) \wedge (x \vee !y \vee !z) \wedge (!x \vee !y \vee !z)$ — ③

- There are 15/16 clauses which are satisfied, which means it is in NP (For a truth value assignment in NP, we can count how many clauses are satisfied and compare it to 15k/16.

- To prove its NP-hard, we will show that 3-SAT $\leq_p$ 3-SAT (15/16). For each set of 8 original clauses, we created 8 new clauses. Since, the no. of clauses is a multiple of 8, any assignment will satisfy 7/8 of new clauses, so we must satisfy all of the original clauses in a valid solution to satisfy exactly 15/16 of the clauses.
- A 3-Sat (15/16) is in intersection of NP and NP-Hard which is the class NP-Complete.

Ques 2. Given a graph $G = (V, E)$ and two ~~string~~ integers k, m ~~and~~ 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.

Ans.

- We prove that,
  Independent set problem $\leq_p$ Dense Subgraph Problem

- If a graph $G(V, E)$ contains an independent set of size k, an independent set decision problem outputs yes, given a graph $G(V, E)$ and an integer k.

- A clique always contain $k*(k-1)/2$ edges if there k vertices in G, and that an independent set in G is a clique in G_c (complement graph of G) and vice-versa. 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.

- Claim: There exists an independent set of size k in G, iff there exists a subgraph of G_c (complementary graph) with at most k vertices and at least $m = k*(k-1)/2$ edges.

- 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.

- 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 for a clique in G_c of size k.

Ques 3. Consider a modified SAT problem, SAT' in which given a CNF formula having m clauses and n variables $x_1, x_2, \ldots, x_n$, the output is YES if there is an assignment to the variables such that exactly $m-2$ clauses are satisfied, and NO otherwise. Prove that SAT' is NP-Complete.

Ans

- To show that SAT' is NP-Complete we will show that SAT' $\in$ NP and SAT $\leq_p$ SAT'.
- SAT' $\in$ NP,

  Given the assignment values as certificates we can evaluate the SAT' instance and verify if it is satisfied. This is same as the SAT-verification. Moreover, we can count the no. of satisfied clauses and check if it is equal to $m-2$ in linear time.
- SAT $\leq_p$ SAT',
  - Add four more clauses $x_1, x_2, \neg x_1, \neg x_2$ to original SAT instance.

    If given SAT instance is $(a \vee b \vee c) \wedge (d) \wedge (a \vee \neg b)$, then after adding more clauses it will be
    $(a \vee b \vee c) \wedge (d) \wedge (a \vee \neg b) \wedge (x_1) \wedge (\neg x_1) \wedge (x_2) \wedge (\neg x_2)$
  - Claim :

    CNF formula obtained for SAT', F' has an assignment which satisfies SAT' iff CNF formula of SAT, F has an assignment which satisfies SAT.
  - If F has an assignment which satisfies SAT, then F' has an assignment which satisfies SAT'.

    - If an assignment $x_1 \ldots x_n$ satisfies

F, then it satisfies exactly two of the four extra clauses, giving exactly $m+2$, which is nothing but $m'-2$ satisfied clauses for the F'.

- If F' has an assignment which satisfies SAT', then F has an assignment which satisfies SAT.
  - The only unsatisfied clauses for F' must be one of x_1 or $\neg x_1$ and one of x_2 or $\neg x_2$, so all the original m clauses are satisfied.

Ques 4. Show that vertex cover is still NP-Complete even when all vertices in the graph are restricted to have even degree.

Ans. • To prove that when all vertices in the graph are restricted to have even degree, vertex cover is still in NP-Complete, we can check in polynomial time

• In a graph, where every vertex has degree 2, every connected component is a cycle.

• We can decide for every component (cycle) separately, how many vertices we need to cover. If graph has m edges, the minimum vertex cover has size [M/2].

• We can start with an instance of vertex cover problem and will construct G' such that G has a vertex cover of size at most k iff all vertices of G' are restricted to have even degree.

• G' will have same set of nodes and edges in G plus a number of new nodes and edges. Add new nodes connected to those which have odd degrees. Newly added nodes will have odd degrees. Connect each of this nodes to another newly added node with odd degree. This will ensure all vertices in the graph have even degree.