🔟

HW10

▼ Question 1

1. (20 pts) 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.

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

QUESTION 1

1. Show 3-SAT(15/16) is NP-co

Certificate : whether a truth-assignment exists for how many clauses is it satisfied & compare it to (15/16)·k.

Certifier : depending on the number of literals, we need to find truth alignment such that the clause is satisfied. We can count the number of such clauses & compare it the 15k/16 as specified in the certificate

2. Select a problem that can be reduced to 15k/16 3-sat
We will choose the 3-sat problem

3. Prove that 3-SAT $\leq_P$ 3-SAT(15/16)
We expand the given original 8 clauses to cover 16 clauses by introducing 3 more variables. These 3 variables will allow for 8 new clauses.
Now if the number of clauses k is a multiple of 8, then a truth alignment exists. Any assignment will satisfy 7/8 clauses & we must satisfy all original 8 clauses. If k is not a multiple of 8, we will satisfy more than 15/16 of the clauses, but even if one of the original clauses are not satisfied, we would have satisfied less than 15/16 clauses.

So, 3-SAT(15/16) is an NP-Hard problem & an NP problem
So, it is an NP-complete problem.

▼ **Question 2**

2. (20 pts) Consider modified SAT problem, SAT' in which given a CNF formula having m clauses in 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 also NP-Complete.

$\boxed{\text{QUESTION - 2}}$

① Prove SAT' ∈ NP

<u>Certificate</u>: If the assignment values of SAT' are given, we can calculate the value of SAT' to check if it is satisfied.

Certifier: We calculate the value of SAT' & compare it with $m-2$

② Select a problem

We choose the SAT problem for reducability.

③ Prove SAT' ∈ NP-complete → Add 4 more clauses: $x, \bar{x}, y, \bar{y}$

<u>Claim</u>: CNF formula for SAT', F' has an assignment that satisfies SAT', iff, there exists an assignment that satisfies SAT

→ <u>Forward claim</u>: If CNF formula has an assignment that satisfies SAT', there is an assignment that satisfies SAT
If an assignment $x_1 \ldots x_n$ of F exists, then it satisfies 2 out of the 4 extra clauses of SAT, ie $m+2$ ie. $m'-2$ for SAT'

→ <u>Backward claim</u>: If F' has an assignment in SAT', then F has an assignment that satisfies SAT
By construction only, one of $(x, \bar{x})$ & one of $(y, \bar{y})$ will be satisfied so all original m clauses are classified.

Hence, proved.

▼ **Question 3**

3. (20 pts) 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.

Solution

1. Prove that Dense Subgraph Problem is an NP problem

   Certificate: Given a graph G(V, E) and 2 integers k, m

   Certifier: We can find if a subgraph exists with k vertices and m edges in polynomial time

2. Chose a problem to reduce it to Dense Subgraph Problem

   We will make use of the Independent set problem

3. Prove that Independency Set <p Dense Subgraph Problem

   Construction: Take the complement of the graph G, which will be Gc.

   Claim: There exists an independent set of size k in graph G iff there exists a subgraph of graph Gc with nodes k and edges m = k(k - 1)/2

   Forward Claim: If there exists a clique of graph Gc of size k, then there exists a subgraph of Gc of size atleast k and edges at most k(k - 1)/2 → By definition, a clique of size k will automatically have edges at most k(k - 1)/2

   Backward Claim: If there exists a subgraph of Gc with vertices k and edges atmost k(k - 1)/2, then there exists a clique of size atleast k → for a subset to have k(k - 1) edges, there has to be k edges. So this subset of k vertices forms a clique of size k in Gc.