Remove Watermark Wondershare PDFelement

Denis R[redacted]

[redacted]

01/04/21

Question 1)

It belongs to P. Consider the following algorithm:

- Iterate over the clause of length L.
  - Save every new term you encounter in its current form.
  - If you encounter a previously saved term's negation, then return false.
- If have terminated the loop, then return true.

If the algorithm terminates while reading ϕ then we know the clause has at least one pair of opposite terms, call them X_n and $!X_n$. Clearly this means that at least on clause will be true. This algorithm runs on poly time on the length of ϕ, thus it is O(L).

Question 2)

Run a certifier through a truth assignment. It can verify if half the terms of each clause are set to true, it runs on poly time over the length of the input and returns yes or no accordingly.

Now, we need to prove NP-completeness. We will reduce 3SAT to this.

Take an input for 3SAT, we will construct a CNF of the form (x1 or x2 or x3 or z1m or z2m or z3m). If the original clause is satisfied then 1, 2 or 3 values of x1, x2 and x3 are set to true. Respectively we would want 0, 1 or 2 terms of z1m, z2m and z3m to be true in these scenarios.

Build a new formula f made of n 3SAT clauses C_i. For each clause add 3 terms as shown above and set the last one to false, name this formula f'. Our clauses look like this:

C_i = (x1 or x2 or x3 or z1m or z2m or false)

If f is satisfiable by a true assignment, then we can set exactly half of the terms in each clause to true by setting either or both z1m and z2m to true. Therefore, is f is satisfiable then the certifier returns yes on f'. We need to prove this in the other direction too.

If the certifier returns yes on f' then exactly half of the terms on each 6 terms clause if true. We know the last one is false thus at least one of x1, x2 or x3 must be true. Therefore, f is satisfiable. If we negate the whole clause, we still have half the terms being true, z=true in this case).

We proved f is satisfiable iff the certifier returns yes for it as an input

We can use the oracle to determine if f' is yes or no and determine if f is satisfiable.

Remove Watermark Wondershare PDFelement

Question 3)

I don't know how to do it :'(

Question 4)

It belongs to P. Consider the following idea, if at most 10 vertices can be of the 3rd color then by removing them, we should get a bipartite graph. The algorithm is as follows:

- For a graph G we calculate every possible subset of vertices size 1 through 10. Name this subset G_s.
    - For every subset, remove it from G to produce $G - G_s$ and run DFS on it to determine if it is bipartite. If any is found to be bipartite then return true.
- If we executed the whole loop without finding any bipartite $G - G_s$ then we return false.

Calculating the subsets takes (n choose k) where $1<=k<=10$. Recall (n choose k) $<= n^k$. Factoring in DFS and its m + n run time we see that the algorithm is $O(n^{10})$.

Question 5)

Let a graph G have n vertices. There are (n choose 100) possible subsets of size 100, we need to verify them until we find an independent set. Consider the following algorithm:

- For a graph G with n vertices calculate every possible subset of size 100. This takes (n choose 100).
    - For each subset check if it is independent. If one is then return true.
- If we finished executing the loop, then we didn't find any. Return false.

Checking if a subset is independent can be done using an adjacency list, since there are n vertices then the list for any vertex is at most size n-1. Recall also (n choose 100) $<= n^{100}$. Thus we have $O(100 * (n-1) * n^{100})$ which is equivalent to $O(n^{100})$ and thus is in P.

Question 6)

Run a certifier through a list of edges to verify it is a Hamiltonian cycle. This takes at most O(m) if the graph G has m edges. Thus, it is NP.

Reduce the Hamiltonian Cycle (NP-Complete) problem to this. Consider the following oracle algorithm:

- For a graph G, iterate through every edge.
    - For each edge, run the oracle on it. If it says yes, then return true and exit.
- If we executed the whole loop, then return false.

We reduced the Hamiltonian cycle problem to this thus showing it is NP-hard too. Therefore, Hamiltonian cycle through an edge e is NP complete.