Intro to Algo HW #7

Thursday, April 30, 2020 5:29 PM

Problem 8.2

In the search procedure there is a sub procedure that takes 2 inputs. It takes in an instance and the proposed solution at that instance. The search procedure returns true if C(I,S) returns true.

The algorithm is as follows:

```
First run the solving procedure on the graph G
C(I,G)
If return true then continue with algo
    Define instance in I
    For all edges (e) in E:
        I = I+1
        G = G-e \\ make a new graph without that specific edge
        If C(I,G):
            Return G
        Else:
            G=G+e \\ return the edge if no Rudrata path

Else:
    No path possible
```

So what happens here is we take away edges until we end up with a Rudrata path.

Analysis: The procedure C(I,S) takes polynomial time. We run it effectively n more times. Because a polynomial times a polynomial is a polynomial the algorithm is polynomial time.

Problem 8.4

a)

Proof:

- Assume clique in graph.
- We must verify that each node has degree at most 3
  - We can do this by running through the edge set and verifying that there is an edge between every pair of vertices.
    - This is done in polynomial time by definition.
  - We can also verify that nodes are degree at most 3 in polynomial time.
- Thus the CLIQUE-3 problem is in NP

b)

The proof is false because there is a wrong assumption of problem difficulty. It must be reduced from CLIQUE to CLIQUE-3 to show that CLIQUE-3 is at least as hard as CLIQUE.

Problem 8.13

a)

Run Prims algorithm on the graph.

If the algorithm finds a node in L then that node is no longer able to be searched from

The final result will have all of L as leaves and be a spanning tree due to the nature of Prims
This is NP complete and can be solved in polynomial time.

b)

We can reduce the NP complete Rudrata problem to this one.

We take the graph G
We take 2 vertices u and v in L
If we claim that there is a tree between u and v then we are claiming there is a path between u and v that does not branch out anywhere. If we claim that the tree we stated is a spanning tree then it has to include every node in G. This is the Rudrata path problem. If we can solve the Rudrata path problem on G that also meets these requirements we have the answer.

Thus the problem is NP complete and can be solved in polynomial time.

c)

This is the same kind of reduction of the Rudrata path as above. This time we run the Rudrata Path problem on G. Because any tree must have at least two leaves we see that the same situation occurs for any 2 nodes u,v in L. Therefore the result of the previous reduction on this problem shows that the set of leaves must be the set L.

Thus the problem is NP complete and can be solved in polynomial time.