Lindsey Bieda and Joe Frambach
Dynamic Programming Problems
10.19.2011

7. Show that if one of the following three problems has a polynomial time algorithm then they all do.

- The input is two undirected graphs G and H. The problem is to determine if the graphs are isomorphic. (referred to as ISOGRAPH)
- The input is two directed graphs G and H. The problem is to determine if the graphs are isomorphic. (referred to as ISODIGRAPH)
- The input is two undirected graphs G and H, and an integer k. The problem is to determine if the graphs are isomorphic and all the vertices in each graph have degree k. (referred to as KISOGRAPH)

Intuitively, two graphs are isomorphic if on can name/label the vertices so that the graphs are identical. More formally, two undirected graphs G and H are isomorphic if there is a bijection f from the vertices of G to the vertices of H such that (v, w) is an edge in G if and only if $(f(v), f(w))$ is an edge in G. More formally, two directed graphs G and H are isomorphic if there is a bijection f from the vertices of G to the vertices of H such that (v, w) is a directed edge in G if and only if $(f(v), f(w))$ is a directed edge in G. The degree of a vertex is the number of edges incident to that vertex.

KISOGRAPH $\leq$ ISOGRAPH

```
program KISOGRAPH:
    read G, H, k
    g = # vertices in G
    h = # vertices in H
    If g ≠ h return 0
    For all vertices n = 1 to g:
        count = 0
        For all vertices m = 1 to g, m ≠ n:
            If edge (n,m) exists in G:
                count = count + 1
        if count ≠ k:
            return 0
    For all vertices n = 1 to g:
        count = 0
        For all vertices m = 1 to g, m ≠ n:
            If edge (n,m) exists in H:
                count = count + 1
        if count ≠ k:
            return 0
    return ISOGRAPH(G,H)
```

Just check the degree of the vertices in G and H and then if that is valid return the result of ISOGRAPH. The degree checking will take $O(n^2)$ time.

ISOGRAPH $\leq$ ISODIGRAPH

program ISOGRAPH:

read G, H
create G' and H' by replacing every edge with two directed edges in both directions ($O(n^2)$)
return ISODIGRAPH(G, H)

Clearly, since the new graphs generated from the input contain every edge going in opposite directions they are equivalent to the undirected graphs, since now the direction of the edge does not matter.

9. The input to the Hamiltonian Cycle Problem is an undirected graph G. The problem is to find a Hamiltonian cycle, if one exists. A Hamiltonian cycle is a simple cycle that spans G. Show that the Hamiltonian cycle problem is self reducible. That it, show that if there is a polynomial time algorithm that determines whether a graph has a Hamiltonian cycle, then there is a polynomial time algorithm to find Hamiltonian cycles.

Hamiltonian Cycle Optimization $\leq$ Hamiltonian Cycle Decision.

program hamiltonian cycle optimization:
read G
If G has a hamiltonian cycle: *call to hamiltonian cycle*
for each node, n in G:
$G' = G - n$
If G' has a hamiltonian cycle:
$G = G'$
output G

13. Show that the following problem is NP-hard:
INPUT: A graph G. Let n be the number of vertices in G.
OUTPUT: 1 if G contains a simple cycle with $\sqrt{n}$ edges, and 0 otherwise.
Use the fact the the following problem is NP-hard:
INPUT: A graph G.
OUTPUT: 1 if G contains a simple cycle that spans G, and 0 otherwise.
Note that a cycle is simple if it doesnt visit any vertex more than once. A cycle spans G if every vertex is included in the cycle.

Simple Cycle (SC) $\leq$ Simple Cycle Square Root (SCSR)

program SC:
read G
n = number verticies in G
$H = n$ disjoint copies of G
output SCSR(H)

We've created a graph, H, with n^2 vertices. Since the copies of G are disjoint the only case in which a simple cycle of length $\sqrt{n}$ will appear is when there is a simple cycle spanning G.