CS 170 Efficient Algorithms and Intractable Problems
Spring 2017 Prasad Raghavendra and Sanjam Garg **Final Review**

NP-Completeness

1. 4D Matching

Recall the 3-DIMENSIONAL MATCHING problem, asking you to match n girls, n boys, and n pets given a list of compatible triples. We know that it is **NP**-complete. In the 4-DIMENSIONAL MATCHING problem you are given compatible *quadruples* of n boys, n girls, n pets, and n *homes*, and again you want to create n harmonious households accommodating them all. Prove 4-DIMENSIONAL MATCHING is **NP-complete**.

Proof: *A solution to 4D matching can be verified in polynomial time by verifying that every boy, girl, pet, home is assigned to exactly one household and vice-versa.*

We will reduce the problem 3D matching to the problem 4D matching.

Given an instance of the problem 3D matching we construct an instance of the problem 4D matching as follows: Create n homes, which are compatible with every triple in our 3D Matching instance.

A common error was to add a home to each triple that was compatible with that triple. □

2. Strongly Connected Subgraph

You are given a strongly connected directed graph $G = (V,E)$ and a budget B, and you are asked to find a subgraph (V,E') of G with $E' \subseteq E$ such that (1) (V,E') is strongly connected, and (2) $|E'| \leq B$. Prove that this problem is **NP**-complete.

Proof: *A solution to Strongly Connected Subgraph can be verified in polynomial time by verifying that the graph is strongly connected using DFS and that the budget constarint is met.*

The above problem is a generalization of the Rudrata Cycle problem, because we can set $B = n$. We know that a strongly connected graph on n vertices with n edges is a cycle on the vertices. For every pair of search problems A and B, if problem A is a generalization of problem B then clearly problem B reduces to problem A. Since we know that the problem Rudrata Cycle is NP-complete, the above problem is NP-complete. □

3. $\frac{1}{3}$ Independent Set

In the $\frac{1}{3}$-INDEPENDENT SET problem you are given a graph (V,E) and you are asked to find an independent set of the graph of size exactly $\frac{|V|}{3}$. In other words, the target size g is not part of the input, but it is always $\frac{|V|}{3}$. Prove this special case of INDEPENDENT SET is **NP**-complete.

Proof: A solution to $\frac{1}{3}$ Independent Set can be verified in polynomial time by checking that there are no edges between the nodes of the independent set and checking its size.

Use a reduction from Independent Set to $\frac{1}{3}$-Independent Set. Consider an instance of Independent Set with a graph G and target size g. If $g \geq |V|/3$, then we can add $3g - |V|$ vertices to G such that the new vertices all have edges to each other and the original vertices. We can then solve our modified G with $\frac{1}{3}$-Independent Set.

If $g < |V|/3$, then we add $(|V| - 3g)/2$ isolated vertices to G. We then solve our modified G with $\frac{1}{3}$-Independent Set and receive an independent set as an answer. From that independent set, we return only the vertices that were in our original graph. □

4. Non-Partisan Traveling Senator Problem

The NON-PARTISAN TRAVELING SENATOR PROBLEM (NPTSP) is a variant of TSP. Each vertex now has one of two colorings (red, blue), and a path cannot include more than three consecutive vertices with the same color. Formulate a reduction from TSP to NPTSP.

Proof: We reduce the problem TSP to the problem NPTSP. Given an instance $G = (V, E)$ of TSP, we construct an instance of NPTSP as follows:

For each vertex $v \in V$, we construct a red vertex v_r and a blue vertex v_b. The edge (v_r, v_b) has cost 0, the edge (v_r, w_b) has cost infinity whenever v and w are distinct vertices in V, the edges (v_r, w_r) and (v_b, w_b) have the same cost as (v, w) in E.

Note that for any TSP path

$$v_1, v_2, \ldots, v_n$$

in G, we can construct a corresponding NPTSP path

$$v_{1,b}, v_{1,r}, v_{2,r}, v_{2,b}, v_{3,b}, v_{3,r}, \ldots, v_{n,r}, v_{n,b}$$

(depending on the parity of n, the last two vertices in the path may be $v_{n,b}, v_{n,r}$). By construction, we never have more than 3 vertices of the same color in a row. This NPTSP has the same cost as our TSP path, so the optimal NPTSP path has cost less than or equal to the cost of the optimal TSP path.

Now consider an optimal NPTSP path. If two consecutive vertices in this path have different colors, then they must be of the form v_r, v_b or v_b, v_r or else they would contribute a cost of infinity to the path. We know that three consecutive vertices cannot all have the same color. I claim that for any vertex $v \in V$, the vertices v_r and v_b must be consecutive vertices in the path. Suppose otherwise and without loss of generality say v_r appears first. From our above arguments, we know that the path from v_r to v_b must take the form

$$v_r, w_{1,r}, w_{1,b}, w_{2,b}, w_{2,r}, \ldots, w_{k,r}, w_{k,b}, v_b$$

Note that v_r and v_b must be the first and last vertices in our optimal path, because otherwise there would be three consecutive red vertices at the beginning or three consecutive blue vertices at the end. But then I could construct a cheaper path by moving v_b to the front. Thus, the optimal NPTSP path takes the form

$$v_{1,b}, v_{1,r}, v_{2,r}, v_{2,b}, v_{3,b}, v_{3,r}, \ldots, v_{n,r}, v_{n,b}$$

and there exists a corresponding TSP path

$$v_1, v_2, \ldots, v_n$$

in G with the same cost. It follows that the optimal TSP path has the same cost as the optimal NPTSP path.

□