CS 573: Graduate Algorithms, Fall 2010
Homework 1

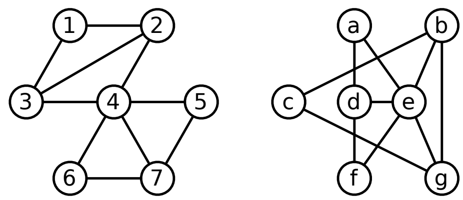
~~Due Friday, September 10, 2010 at 1pm~~

Due Monday, September 13, 2010 at 5pm
(in the homework drop boxes in the basement of Siebel)

For this and all future homeworks, groups of up to three students may submit a single, common solution. Please neatly print (or typeset) the full name and NetID on each page of your submission.

---

1. Two graphs are said to be ***isomorphic*** if one can be transformed into the other just by relabeling the vertices. For example, the graphs shown below are isomorphic; the left graph can be transformed into the right graph by the relabeling $(1,2,3,4,5,6,7) \mapsto (c,g,b,e,a,f,d)$.



Two isomorphic graphs.

Consider the following related decision problems:

- GRAPHISOMORPHISM: Given two graphs G and H, determine whether G and H are isomorphic.
- EVENGRAPHISOMORPHISM: Given two graphs G and H, such that every vertex in G and H has even degree, determine whether G and H are isomorphic.
- SUBGRAPHISOMORPHISM: Given two graphs G and H, determine whether G is isomorphic to a subgraph of H.

(a) Describe a polynomial-time reduction from EVENGRAPHISOMORPHISM to GRAPHISOMORPHISM.

(b) Describe a polynomial-time reduction from GRAPHISOMORPHISM to EVENGRAPHISOMORPHISM.

(c) Describe a polynomial-time reduction from GRAPHISOMORPHISM to SUBGRAPHISOMORPHISM.

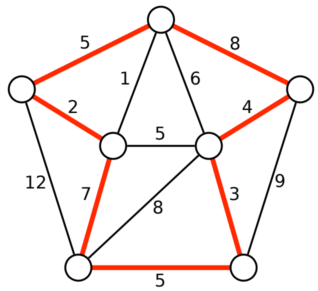
(d) Prove that SUBGRAPHISOMORPHISM is NP-complete.

(e) What can you conclude about the NP-hardness of GRAPHISOMORPHISM? Justify your answer.

[Hint: These are all easy!]

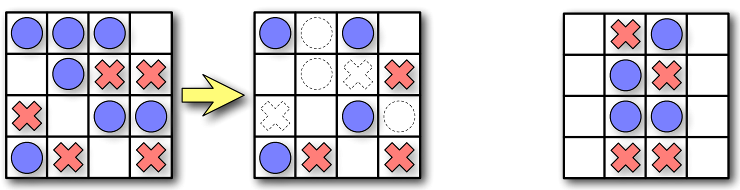
2. Suppose you are given a magic black box that can solve the 3COLORABLE problem ***in polynomial time***. That is, given an arbitrary graph G as input, the magic black box returns TRUE if G has a proper 3-coloring, and returns FALSE otherwise. Describe and analyze a ***polynomial-time*** algorithm that computes an actual proper 3-coloring of a given graph G, or correctly reports that no such coloring exists, using this magic black box as a subroutine. *[Hint: The input to the black box is a graph. Just a graph. Nothing else.]*

3. Let G be an undirected graph with weighted edges. A *heavy Hamiltonian cycle* is a cycle C that passes through each vertex of G exactly once, such that the total weight of the edges in C is at least half of the total weight of all edges in G. Prove that deciding whether a graph has a heavy Hamiltonian cycle is NP-complete.



A heavy Hamiltonian cycle. The cycle has total weight 34; the graph has total weight 67.

4. Consider the following solitaire game. The puzzle consists of an $n \times m$ grid of squares, where each square may be empty, occupied by a red stone, or occupied by a blue stone. The goal of the puzzle is to remove some of the given stones so that the remaining stones satisfy two conditions: (1) every row contains at least one stone, and (2) no column contains stones of both colors. For some initial configurations of stones, reaching this goal is impossible.



A solvable puzzle and one of its many solutions. An unsolvable puzzle.

Prove that it is NP-hard to determine, given an initial configuration of red and blue stones, whether the puzzle can be solved.

5. A boolean formula in ***exclusive-or conjunctive normal form*** (XCNF) is a conjunction (AND) of several *clauses*, each of which is the *exclusive*-or of one or more literals. For example:

$$(u \oplus v \oplus \bar{w} \oplus x) \wedge (\bar{u} \oplus \bar{w} \oplus y) \wedge (\bar{v} \oplus y) \wedge (\bar{u} \oplus \bar{v} \oplus x \oplus y) \wedge (w \oplus x) \wedge y$$

The XCNF-SAT problem asks whether a given XCNF boolean formula is satisfiable. Either describe a polynomial-time algorithm for XCNF-SAT or prove that it is NP-complete.