Algorithms, Section 2
Spring 2003
Homework 12b
Due Tuesday, April 29

This homework set is quite challenging. Please start early and please come to office hours. In this homework set you will prove that several problems are NP-complete. Recall that proving NP-completeness has several parts: First you will need to prove that the problem is in NP (which is usually easy). Next, you must describe a reduction from a known NP-complete problem to this problem and explain why the reduction can be performed in polynomial time. Finally, you must show that the answer to the problem on the left-hand side of the reduction operator $\leq_p$ is true if and only if the answer to the problem on the right-hand side of the reduction operator is true. Be sure to write out your NP-completeness proofs carefully and precisely!

1. [**20 Points**] **More on ILP!** In class we proved that Integer Linear Programming (ILP) is NP-complete. Our reduction was from 3SAT. In this problem you will prove that ILP is NP-complete using two different reductions. You do not need to show that ILP is in NP (we already argued that in class). You need only give a reduction, prove its correctness (remember, this is an "if and only if" proof), and explain why the reduction can be done in polynomial time.

   (a) Show that ILP is NP-complete using a reduction from Vertex Cover.

   (b) Show that ILP is NP-complete using a reduction from Subset Sum.

2. [**20 Points**] **SPAM is NP-Complete!** It's true! Here's the story: Millisoft has just released it's new product "Portholes 1987" and Gill Bates has decided to celebrate by throwing a big party for his employees. Unfortunately, many of the employees at Millisoft do not get along with one another[1]. Gill would like to invite as many employees to the party as possible such that no two partygoers dislike one another. This is known as the **S**electing **P**artygoers **A**t **M**illisoft (**SPAM**) Problem.

   Given is a set $E = \{e_1, \ldots, e_n\}$ of n employees and a set H which consists of pairs of employees (e_i, e_j) such that employees e_i and e_j do not get along. In the optimization problem, we wish to find the largest subset $S \subseteq E$ such that for every pair of employees $e_k, e_\ell \in S$, $(e_k, e_\ell) \notin H$.

   (a) Formulate the SPAM problem as a decision problem.

   (b) Prove that the SPAM decision problem is NP-complete. (*Note:* Use one of 3SAT, Vertex Cover, Independent Set, or Clique in your reduction.)

3. [**15 Points**] **The Kernel!** Consider a directed graph $G = (V, E)$. A *kernel* for graph G is a subset K of the vertices such that:

[1]The famous social satirist Tom Lehrer once said "There are people in this world who do not love one another and I **HATE** people like that!"

(a) Every vertex in V is either in the kernel K or has an incoming edge from some vertex in the kernel. (Formally, "$\forall v \in V$, $v \in K$ or $\exists u \in K$ s.t. $(u, v) \in E$.")

(b) There do not exist two vertices in the kernel K with a directed edge between them. (Formally, "$\forall u, v \in K$, $(u, v) \notin E$ and $(v, u) \notin E$.")

Observe that the graph below has a kernel. (A set containing just one of the two vertices is a kernel.)

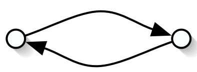

Figure 1: This directed cycle has a kernel.

On the other hand, the graph in Figure 2 has no kernel. (Take a look at the definition of the kernel to make sure you see why.)

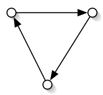

Figure 2: This directed cycle has no kernel.

The **Kernel Problem** is the following: Given a directed graph $G = (V, E)$, does G have a kernel? Prove that the Kernel Problem is NP-complete. (Hint: Perform a reduction from 3SAT and use the graphs in Figures 1 and 2 as the gadgets.)

4. **[25 Points] 2SAT!** Recall that 2SAT is just like 3SAT except that every clause in a 2SAT instance contains the disjunction of exactly two variables. While 3SAT is NP-complete, 2SAT can be solved by efficient algorithms. In class, we mentioned that the *resolution* algorithm from CS 80 can be used to solve 2SAT. In this problem, we will see that 2SAT can be solved using graph algorithms instead. Amazing but true! (*Note:* We use the notation $\overline{x}$ to denote the negation of variable x.)

(a) Consider the following instance of 2SAT:

$$(x_1 \vee x_2) \wedge (x_1 \vee \overline{x_2}) \wedge (\overline{x_1} \vee x_2)$$

Show that this instance is satisfiable by giving a satisfying assignment for the variables.

(b) Now consider the following instance of 2SAT:

$$(x_1 \vee x_2) \wedge (x_1 \vee \overline{x_2}) \wedge (\overline{x_1} \vee x_2) \wedge (\overline{x_1} \vee \overline{x_2})$$

Explain briefly why this instance of 2SAT is not satisfiable.

(c) Given an instance of 2SAT, let's construct a corresponding directed graph as follows: For each variable x_i that appears in the 2SAT instance, construct a *pair* of vertices, one labeled x_i and the other labeled $\overline{x_i}$. For every clause of the form $(a \vee b)$ in the 2SAT instance (where a and b are variables which may or may not be negated), place a directed edge from vertex $\overline{a}$ to vertex b and also a directed edge from vertex $\overline{b}$ to vertex a. For example, if we had a clause $(x_1 \vee \overline{x_3})$ then $a = x_1$ and $b = \overline{x_3}$. Therefore, we would place a directed edge from vertex $\overline{x_1}$ to vertex $\overline{x_3}$ as well as a directed edge from vertex x_3 to vertex x_1. In this example, you should interpret the edge from vertex $\overline{x_1}$ to vertex $\overline{x_3}$ to mean "if $\overline{x_1}$ is true (that is, x_1 is false) then $\overline{x_3}$ must be true." Similarly, the edge from vertex x_3 to the vertex x_1 is interpreted as "if x_3 is true then x_1 must be true."

Construct this directed graph for the 2SAT instance in part (a). This graph should contain 4 vertices 6 edges. Notice that there is no path from vertex x_1 to vertex $\overline{x_1}$.

(d) Notice that in the graph you constructed there is a path from vertex $\overline{x_1}$ to vertex x_1. Clearly explain why *the existence of this path* implies that a satisfying assignment for this instance of 2SAT cannot have $\overline{x_1}$ be **true** (that is, x_1 cannot be **false**). Be very clear and precise about your reasoning here.

(e) Notice that there does *not* exist a path in this graph from vertex x_1 to vertex $\overline{x_1}$. Notice also that in your satisfying assignment for this instance, x_1 was set to **true**. Clearly explain why this graph tells us that x_1 should be assigned **true** if the instance has any hope of being satisfied.

(f) What does the graph tell you about the value that should be assigned to variable x_2? Explain.

(g) Now, construct the graph for the 2SAT problem in part (b). What property does this graph have that forces you to conclude that the 2SAT instance is not satisfiable? Be precise.

(h) Next, write down a conjecture that starts as follows: "A 2SAT instance is satisfiable if and only if the corresponding directed graph has the property that blah, blah, blah." Fill in the "blah, blah, blah" with mathematically precise language.

(i) Now, prove your conjecture. This is the main part of this problem (it's also worth most of the points!) and will take a few paragraphs to prove. Note that it is an "if and only if" proof, which means that there are two things to show. In this proof, the fact that two edges were introduced for each clause will be absolutely critical. You'll need to use this fact!

(j) Now, describe an algorithm to determine whether or not a 2SAT instance is satisfiable. What is the running time of your algorithm?