Due: Monday, April 3, 2017, 11am on MarkUs

You will receive 20% of the points for any (sub)problem for which you write "I do not know how to answer this question." You will receive 10% if you leave a question blank. If instead you submit irrelevant or erroneous answers you will receive 0 points. You may receive partial credit for the work that is clearly "on the right track."

1. (20 pts) **Integer Programming**

   Integer programming (IP) problem is a linear programming problem with the additional constraint that variables have to take on integer values. The standard form for an integer program is

$$\begin{array}{lrl} \text{Maximize} & c^T x & \\ \text{Subject to} & Ax & \leq b \\ & x & \geq 0 \\ & x & \in \mathbb{Z}^n \end{array}$$

   In the above $c \in \mathbb{R}^n$, $A \in \mathbb{R}^{m \times n}$ and $b \in \mathbb{R}^m$. Thus, the only difference with the regular LP standard form is the additional constraint $x \in \mathbb{Z}^n$.

   (a) Define IP dual of the IP problem in standard form. Prove that the principle of weak duality still holds for IP.

   (b) Show that strong duality does not always hold for IP, i.e., find IP primal such that its optimal is strictly less than the optimal of its IP dual.

   (c) $\{0,1\}$-IP feasibility problem: given $A \in \mathbb{Z}^{m \times n}$ and $b \in \mathbb{Z}^m$ determine if there exists $\widehat{x} \in \{0,1\}^n$ such that $A\widehat{x} \leq b$. State IP feasibility problem as a language and prove that it is NP-complete.

2. (20 pts) Consider the following decision problem. You are given a directed graph G, k starting vertices $s_1, \ldots, s_k$, and k finishing vertices $t_1, \ldots, t_k$. You need to decide if there exist k node-disjoint paths connecting s_i to the corresponding t_i for each $i \in [k]$.

   (a) Formulate the above problem as a language.

   (b) Show that 3-SAT reduces to this language.

   (c) Derive the corollary that this language is NP-complete.

   (d) **Short answer question.** Suppose instead of trying to decide existence of node-disjoint paths, we were trying to decide existence of edge-disjoint paths (now, the paths are allowed to share nodes, but not edges). Is this problem in P, or is it NP-complete? State your answer clearly, and give a brief justification (at most 5 short sentences).

3. (20 pts) Given an undirected graph $G = (V, E)$, a k-coloring of G is a function $c : V \to [k]$ such that for every edge $\{u, v\} \in E$ we have $c(u) \neq c(v)$. If G has a k-coloring, G is called k-colorable.

   (a) Describe a polynomial time algorithm for deciding if a graph G is 2-colorable in plain English. Provide a concise argument of correctness. State and justify the running time.

(b) Consider the following language

$$L_{\text{COL}} = \{\langle G, k\rangle \mid\ G \text{ is } k\text{-colorable}\}.$$

It is known that L_{COL} is NP-complete. Use this fact to prove that the language corresponding to the following scheduling decision problem is NP-complete.

Given a list of final exams $F_1, \ldots, F_k$ to be scheduled, and a list of students $S_1, \ldots, S_\ell$. For each student you are also given the subset of exams that the student is taking. In addition you are given a natural number h. You must schedule the exams into time slots so that no student is required to take two exams in the same slot. The problem is to determine if such schedule exists that uses at most h time slots. State this problem as a language and prove that it is NP-complete.

4. (20 pts) Given two languages L_1 and L_2, the concatenated language, denoted by L_1L_2, is defined as

$$L_1L_2 = \{w_1w_2 \mid w_1 \in L_1 \wedge w_2 \in L_2\}.$$

This allows us to define powers of a language L recursively as

$$L^1 = L \text{ and } L^i = L^{i-1}L \text{ for } i \geq 2.$$

By convention we have $L^0 = \{\varepsilon\}$, where ε is the empty string. The dagger of language L is the language

$$L^\dagger = \bigcup_{i=0}^{\infty} L^i.$$

For this question you need to show that the class P is closed under the dagger operation. That is you need to prove: if L is in P then $L^\dagger$ is in P.

Let A be a polytime algorithm deciding L. You need to design a polytime algorithm for deciding $L^\dagger$. Let $w[1..n]$ be the input to your algorithm deciding $L^\dagger$. Use dynamic programming.

(a) Describe the semantic array.

(b) Describe the computational array.

(c) Justify why the above two arrays are equivalent.

(d) What is the running time of your algorithm? State it in terms of the input length and the running time of the algorithm A. Justify the stated runtime.

5. (20 pts) **Minesweeper on Graphs**

Let G be an undirected graph. Consider the following version of the game Minesweeper. Each node in G is either empty or contains a single hidden mine. If the player clicks on a node with a mine, the player loses the game. If the player clicks on a node without a mine, the node is labeled with the number of mines contained in the adjacent (neighboring) nodes. The regular Minesweeper game is a special case played on the grid graph.

Now, consider *mine consistency problem.* You are given a graph G, in which some nodes are labeled with numbers and other nodes are unlabeled. The goal is to decide if it is possible to place mines on some of the unlabeled nodes such that all labels are correct, that is if node v is labeled with number k then it has exactly k neighbors with mines.

(a) Formulate mine consistency problem as a language.

(b) Show that 3-SAT reduces to this language.

(c) Derive the corollary that mine consistency problem is NP-complete.