

CS 570 (Spring 2009) — First Midterm (Takehome)

Unless you are specifically asked to reprove a known fact, you can use all facts from class, homeworks and the textbook without proof. There are four questions on this exam. The only legitimate sources in solving this exam are (1) your textbook and course notes, (2) any handouts provided by us, (3) discussions with the TA or instructor. You cannot discuss the problems with anyone else (whether in the class or not), nor can you look for solutions in other textbooks, on the Internet, or in other sources. The exam is due in David Kempe's office no later than 12:00 noon on Friday, 03/13/2009. (You are very welcome to submit in class on Thursday, 03/12/2009, instead.) If you submit after that, your exam will automatically be graded as 0.

G O O D L U C K

(1) [10 points]

Suppose that you are consulting for a company that has n employees and the same number n of projects. Each employee works for 40 hours per week, and each project requires 40 hours per week of work. Unfortunately, for legacy reasons, projects are currently divided between multiple people. Specifically, employee i spends a_{ij} hours per week on project j , for each i, j . While $\sum_j a_{ij} = 40$ for all i , and $\sum_i a_{ij} = 40$ for all j (that is, each project gets all its attention, and each employee works the full 40 hours), this is a suboptimal situation. So the company wants to instead reassign the employees, so each employee works full-time on only one project. In addition, each employee i should work on a project they are already familiar with, i.e., a project j such that currently, $a_{ij} > 0$. Prove that such a reassignment is always possible.

(2) [10 points]

Let E be a set of elements, and \mathcal{M} a nonempty collection of subsets of E . \mathcal{M} is called a *matroid* if it satisfies the following two properties:

- (a) *Downward closure*: If $S \in \mathcal{M}$, and $T \subseteq S$, then $T \in \mathcal{M}$.
- (b) *Exchange Property*: If $S, T \in \mathcal{M}$, and $|T| > |S|$, then there exists at least one element $e \in T \setminus S$ such that $S \cup \{e\} \in \mathcal{M}$. In words, if T is a strictly larger set than S , then at least one element of T can be added to S and result in another set of the matroid.

The sets $S \in \mathcal{M}$ are also called *independent sets*¹

- (a) [5 points] Prove that for any given undirected graph G with edges E , the set of all forests of G forms a matroid. That is, if we define the set $\mathcal{M} = \{E' \subseteq E \mid E' \text{ contains no cycles}\}$, then \mathcal{M} is a matroid.
- (b) [5 points] We are now returning to arbitrary matroids (not necessarily forests). Assume that each element $e \in E$ of the matroid has a weight $w_e \geq 0$. Show how to modify Kruskal's algorithm to find an independent set S of maximum total weight $\sum_{e \in S} w_e$. Prove that your modified algorithm is correct, and runs in polynomial time.

¹The reason for this name is that one particular matroid is obtained by taking E to be any set of vectors, and \mathcal{M} the set of all subsets of E that are linearly independent.

(3) [10 points]

Suppose that we are worried about the outbreak of a computer virus in a network. The network is modeled by an undirected graph $G = (V, E)$. We have reason to suspect that the outbreak will happen from a given node s . In order to deal with the outbreak, we can do two things:

- (a) Disable some connections of the network. Specifically, to disable edge e costs $c_e \geq 0$. Once an edge has been disabled, the virus cannot propagate along that edge any more. On the other hand, for any edge $e = (u, v)$ that is *not* disabled, once u becomes infected, the virus will always manage to propagate along e and infect v as well.
- (b) Spend time and money to clean up the infected machines after the fact. For each node v , the quantity $d_v \geq 0$ captures the cost of cleaning up v . Of course, we only need to clean up machines that actually got infected.

The goal is to find a strategy $E' \subseteq E$ minimizing the total cost, i.e., the sum of the disabling costs $\sum_{e \in E'} c_e$, plus the sum of cleanup costs for nodes v reachable from s without using edges in E' .

Give a polynomial-time algorithm for finding such a strategy, and prove that your algorithm is correct (and runs in polynomial time).

(4) [10 points]

You are given two strings x, y . Both contain ordinary characters, but can also contain the two wildcards '?' and '*'. Just as in your standard operating system shell, '?' can match any one character, while '*' can match any sequence of 0 or more characters (including '?' or other '*', of course). Your goal is to decide if x and y match in this sense, i.e., whether true characters can be substituted for all '?', and sequences of characters for all '*', such that the resulting strings match exactly.

Give a polynomial-time algorithm for deciding if the strings match in this sense, and prove that it is correct (and runs in polynomial time).

E N D O F E X A M