

# CS599 (Spring 2007) - Takehome Midterm

Due in class on Thursday, 03/22

No late submission will be granted, so you may want to start early. Contrary to the policy on homeworks, you cannot discuss problems, ideas, or solutions to this midterm with anyone (in or outside of class) except myself. Also, as a reminder, you are not allowed to seek solutions to these problems online or elsewhere. If you are stuck on a problem and need hints, or you need help understanding a problem, you are welcome to e-mail me, or talk to me in person.

## Problem 1

You are given a universe  $U$  of  $n$  elements. For each  $i = 1, \dots, m$ , you are given two subsets  $S_i, T_i \subseteq U$ . Your goal is to select exactly one of  $S_i, T_i$  for each  $i$ , while minimizing the size of the union of the sets selected. As an application, imagine that you need to do  $m$  jobs, and job  $i$  can be performed either by team  $S_i$  or by team  $T_i$ . Your goal is to hire as few people total as possible, while ensuring that all jobs can be performed with subsets of the teams you hired.

Give (and analyze) a polynomial time 2-approximation algorithm for this problem. (Hint: formulate an LP and round it appropriately.)

## Problem 2

You are given a directed acyclic graph  $G = (V, E)$ , in which each edge  $e$  has an integer cost  $c_e \geq 0$  (think highway tolls) and an integer delay  $d_e \geq 0$ . You only have  $C$  dollars total, and want to pay for an  $s$ - $t$  path, while minimizing the delay subject to the constraint of not paying more than  $C$ . Give (and analyze) an FPTAS for this problem.

## Problem 3

Suppose that we have a SET COVER problem with costs  $c_S$  for sets, but are somewhat tolerant, in that we don't need to cover all elements. Instead, the requirement is to cover at least 90% of all elements, but we get to choose which elements exactly are covered. (In other words, the problem is "Find the cheapest way to cover at least any 90% of the elements".)

Prove that by appropriately modifying the greedy algorithm from class and its analysis, we get an algorithm that pays at most  $\alpha \cdot \text{OPT}$ , for some constant  $\alpha$ , where OPT is the optimum solution covering *all* elements. That is, the algorithm has a two-fold advantage: it has to cover fewer elements, but gets to compare its cost against an optimum solution covering all elements.

## Problem 4

Formulate the shortest  $s$ - $t$  path problem as a cut-based LP (similar to your Steiner Tree LP). Use primal-dual techniques and reverse deletion to obtain a polynomial time 1-approximation algorithm, i.e., a polynomial time exact algorithm. (Hints: (1) Think about the order in which you raise duals. (2) Your algorithm might end up looking a lot like Dijkstra's algorithm; you should still use duality theory to prove optimality.)