Massachusetts Institute of Technology
6.854J/18.415J: Advanced Algorithms
Erik Demaine and David Karger

Handout 17
Monday, October 27, 2003

Problem Set 9

Due: Monday, November 3, 2003.

NONCOLLABORATIVE Problem 1. You are given a collection of n points in some metric space (i.e., the distances between the points satisfy the triangle inequality). Consider the problem of dividing the points into k clusters so as to minimize the maximum diameter of (distance between any two points in) a cluster.

(a) Suppose the optimum diameter d is known. Devise a greedy 2-approximation algorithm. **Hint:** consider any point and all points within distance d of it.

(b) Consider the algorithm that (k times) chooses as a "center" the point at maximum distance from all previously chosen centers, then assigns each point to the nearest center. By relating this algorithm to the previous algorithm, show that you get a 2-approximation.

Problem 2. The *Steiner Tree Problem* presents a graph G with edge costs c_e, and a subset T of the vertices called *terminals*. The goal is to construct a minimum cost tree spanning all the terminals (it may also include any desired subset of the non-terminals; these included vertices are called *Steiner points*).

(a) Suppose you compute all-pairs shortest paths in G, and create a complete graph G' on the terminals T where each edge cost is equal to the shortest path between its (terminal) endpoints in G. Relate the cost of the minimum spanning tree in this graph to the cost of the optimum steiner tree in G.

(b) Give a 2-approximation algorithm for the Steiner tree problem.

Problem 3. Suppose you are given a set of jobs with processing times p_j and *release dates* r_j, before which job j cannot be processed. Consider scheduling the jobs so as to minimize the *average completion time* of the jobs. Let C_j denote the completion time of job j, and note the average is proportional to $\sum C_j$.

(a) Suppose that it is permissible to *preempt* a job, setting it aside midway to work on another job, and then returning where one left off on the first job. Argue that the optimum (greedy) strategy is to work on whichever job is available (released) with the shortest remaining processing time (after subtracting any work already done on the job) **Hint:** consider the consequences of swapping two (pieces of) jobs that occur next to each other in the order.

(b) Show that scheduling the jobs *non-preemptively* in order of their completion time in the preemptive schedule increases each completion time only by a factor of 2.

(c) Argue that this yields a 2-approximation to (nonpremptive) average-completion-time scheduling with release dates.

OPTIONAL (d) There is actually a PTAS for this problem. Can you find it?

Problem 4. In class we saw a greedy algorithm for set cover that achieved an approximation ratio of $O(\log n)$ given n sets. We can achieve the same ratio with a linear programming/randomized rounding solution.

(a) Devise a linear programming relaxation for set cover, generalizing the one we saw in class for vertex cover.

(b) Suppose we apply randomized rounding to the fractional set-usage variables. Argue that each element is covered with constant probability. (You will need to upper bound $\prod(1 - x_i)$ when $\sum x_i \geq 1$.)

(c) By applying the rounding experiment repeatedly, arrange for every element to be covered with high probability while using only $O(\log n)$ times the optimum number of sets.

(d) Devise a different scheme that multiplies all the variables by $O(\log n)$ before rounding, then uses the Chernoff bound to prove that it works.

Problem 5. Consider the *set basis* problem: given a finite collection C of finite sets, and given a parameter k, is there a finite collection B of k sets such that every set $A \in C$ equals the union of some subset of B?

(a) If the answer to a set-basis instance is "yes", what can you say about the number of sets in C?

(b) If two elements x and y appear in precisely the same family of sets in C (i.e., for all $S \in C$, $x \in S$ if and only if $y \in S$), show that removing y from all sets in C preserves the answer to set basis.

(c) Show that set basis is fixed-parameter tractable by reducing to a problem kernel of size $f(k)$.