Introduction to approximation and randomized algorithms

2nd home assignment

Deadline: 12th January 2020 11:59PM
You can hand your solutions in during the exercise sessions or send them to me by email to `mberg@kam.mff.cuni.cz`.

Exercise 1 (4 points)**.** Consider a network with n vertices (computers) and circular topology (the underlying graph is a cycle). On input we get a list of messages where each message has a source vertex and a target vertex. To every message we want to assign one of the two possible paths such that the load (the total number of uses) of the most loaded edge is minimized. Construct a 2-approximation algorithm for this problem. I recommend using a rounding of linear relaxation, but of course you can use whatever method you like.

Exercise 2 (6 points)**.** Remember the LP-SAT algorithm you've seen in the lecture. This algorithm has an approximation ratio $1 - 1/e$. You've also seen that when we combine this algorithm with RAND-SAT, then we get a 3/4-approximation. Our goal now is to get a 3/4-approximation without combining those two algorithms. More specifically we want to modify LP-SAT algorithm to get a 3/4-approximation. In LP-SAT algorithm we solve a linear program with variables y_i (for logical variable x_i we have a variable y_i in the program) and z_j (for clause C_j we have variable z_j). Suppose we have an optimal solution y^*,z^* of this program and we set variable x_i to *true* with probability y_i^*. Now we modify this step and we set variable x_i to *true* with probability $y_i^*/2 + 1/4$. Your task is to show that the algorithm with this modification is a 3/4-approximation.

Exercise 3 (5 points)**.** Now we'll look at the k-supplier problem. In this problem we get a metric space with $m+n$ points, such that m of them are suppliers and n of them are customers. Our goal

is to select k suppliers such that the maximal distance between a customer and its closest supplier is minimized. Construct and analyze 3-approximation algorithm.

Note: This problem is similar to k-center problem which was supposed to be covered during one of the exercise sessions. So it might be useful to read about this k-center problem in the book The Design of Approximation Algorithms by Williamson and Shmoys, which is freely available online. It's chapter 2.2.

Exercise 4 (6 points)**.** Consider a scheduling on machines with speeds. We have m machines with speeds $s_1 \geq s_2 \geq \cdots \geq s_m$ and n jobs with lengths $p_1, \ldots, p_n$, where all of these numbers are positive integers. The time needed to process a job j on machine i is p_j/s_i. We want to schedule these jobs on the machines such that the length of the schedule is minimized. Our goal is to get a 2-approximation algorithm.

a) First show that from a c-relaxed decision procedure we can construct a c-approximation algorithm. A c-relaxed decision procedure is a procedure which gets an input of the problem together with a number D, runs in the polynomial time and either returns a schedule of length at most cD or correctly stated that no schedule of length at most D exists.

b) Then we want to show that the following procedure is a 2-relaxed decision procedure. First to each job we assign its type, which is the index of the slowest machine which can process the job in time at most D. That is the type of job j is the maximal i such that $p_j/s_i \leq D$. If there exists a job that cannot be processed in time D even on the fastest machine we return that no schedule exists. Otherwise suppose we have a partial schedule and we want to schedule another job on machine i. If the current schedule of machine i ends at time D or later, we do not schedule another job to this machine. Otherwise we choose an unscheduled job of type i (if it exists) or a job of the lowest type from $i+1, i+2, \ldots, m$ a schedule it on this machine. That means that machine i first processes jobs of type i, then of type $i+1$, type $i+2$,

etc. If we can schedule all jobs by this procedure, we return this schedule. Otherwise we state that no schedule of length at most D exists.