Spring 2011, CS 598CSC: Approximation Algorithms

Homework 1
Due: 02/11/2011 in class

Instructions and Policy: Each student should write up their own solutions independently. You need to indicate the names of the people you discussed a problem with; ideally you should discuss with no more than two other people.

Solve as many problems as you can. I expect at least three and four (or more) is ideal.

Please write clearly and concisely - clarity and brevity will be rewarded. Refer to known facts as necessary. Your job is to convince me that you know the solution, as quickly as possible.

Problem 1 Let $G = (V, A)$ be a directed graph with arc weights $c : A \rightarrow \mathcal{R}^+$. Define the density of a directed cycle C as $\sum_{a \in C} c(a)/|V(C)|$ where $V(C)$ is set of vertices in C.

1. A cycle with the minimum density is called a minimum mean cycle and such a cycle can be computed in polynomial time. How?

   Hint 1: Given density λ, give a polynomial-time algorithm to test if G contains a cycle of density $< \lambda$. Now use binary search.

   Hint 2: There is a polynomial time algorithm to detect if a graph has a negative cycle (a cycle with sum of arc lengths negative).

2. Consider the following algorithm for ATSP. Given G (with c satisfying asymmetric triangle inequality), compute a minimum mean cycle C. Pick an arbitrary vertex v from C and recurse on the graph $G' = G[V - C \cup \{v\}]$. A solution to the problem on G can be computed by patching C with a tour in the graph G'. Prove that the approximation ratio for this heuristic is at most $2H_n$ where $H_n = 1 + 1/2 + \ldots + 1/n$ is the nth harmonic number.

Problem 2 Consider the k-dimensional knapsack problem. We are given n non-negative k-dimensional vectors $\bar{v}_1, \bar{v}_2, \ldots, \bar{v}_n$. Each vector has a non-negative weight, w_i for $\bar{v}_i$. We are also given a k-dimensional knapsack $\bar{V}$ and the goal is to find a maximum weight subset of vectors that pack in to V. A subset of vectors pack into $\bar{V}$ if their vector addition is less than $\bar{V}$ (co-ordinate wise). Prove that there is a pseudo-polynomial time algorithm for this problem when k is a fixed constant independent of n. What is the running time of your algorithm? Prove that even for $k = 2$ and *unit weights* the problem does not admit an FPTAS. (Hint: use a reduction from the Partition problem.) **Extra credit:** Obtain a PTAS for this problem when k is a fixed constant independent of n (might need to use LP techniques).

Problem 3 Consider a budgeted version of the maximum coverage problem. We are given m sets $S_1, S_2, \ldots, S_m$, each a subset of a set $\mathcal{U}$. Each set S_i has a non-negative cost c_i and we are also given a budget B. The goal is to pick sets of total cost at most B so as to maximize the number of elements covered. Show that if $c_i \leq \epsilon B$ for $1 \leq i \leq m$ then the Greedy algorithm yields a $1 - 1/e - \epsilon$ approximation (for sufficiently small ϵ). For any fixed $\epsilon > 0$ obtain a $1 - 1/e - \epsilon$ approximation. (Hint: consider the PTAS for knapsack from the lectures. You might find the inequality $1 - x \leq e^{-x}$ useful.) **Extra credit:** Obtain a $1 - 1/e$ approximation for this problem.

Problem 4 In the submodular set cover problem, we are given a universe $\mathcal{U}$ of elements, and a monotone submodular function $f : 2^{\mathcal{U}} \to \mathcal{R}^+$. The goal is to pick a smallest $U' \subseteq \mathcal{U}$ such that $f(U') = f(\mathcal{U})$. In the submodular maximum coverage problem, we are also given an integer k, and the goal is to pick a set $U' \subseteq \mathcal{U}$ of size at most k to maximize $f(U')$.

1. Prove that the greedy algorithm gives a $1 - 1/e$ approximation for submodular maximum coverage.
2. Consider a $k \times \ell$ matrix M where each entry $M_{i,j}$ is a subset of a universe $\mathcal{U}$ of size $n \geq \ell$. We say a row M_i of M is *covered* by $U' \subseteq \mathcal{U}$ if there are ℓ *distinct* elements $e_1, e_2, \ldots e_\ell$ in U' such that $e_j \in M_{i,j}$ for each $1 \leq j \leq \ell$.

   Prove that the problem of finding a smallest $U' \subseteq \mathcal{U}$ such that each row is covered by U' is a special case of the submodular set cover problem.

Problem 5 For Metric-TSP consider the nearest neighbour heuristic discussed in class. Prove that the heuristic yields an $O(\log n)$ approximation. (Hint: use the basic idea in the online greedy algorithm for the Steiner tree problem from Lecture 1). **Extra Credit:** Give an example to show that there is no constant c such that the heuristic is a c-approximation algorithm.

Problem 6 Multi-processor scheduling: given n jobs $J_1, \ldots, J_n$ with processing times $p_1, p_2, \ldots, p_n$ and m machines $M_1, M_2, \ldots, M_m$. For identical machines greedy list scheduling that orders the jobs in non-increasing sizes has an approximation ratio of 4/3. See Section 2.3 in the Shmoys-Williamson book.

Now consider the problem where the machines are not identical. Machines M_j has a speed s_j. Job J_i with processing time p_i takes p_i/s_j time to complete on machine M_j. Give a constant factor approximation for scheduling in this setting to minimize makespan (the maximum completion time). (Hint: consider jobs in decreasing sizes. Assuming $p_1 \geq p_2 \geq \ldots \geq p_n$ and $s_1 \geq s_2 \geq \ldots s_m$, show that $OPT \geq \max_{i \leq m} (\sum_{j \leq i} p_j / \sum_{j \leq i} s_j$.)