CS264: Homework #7

Due by midnight on Wednesday, November 12, 2014

Instructions:

(1) Students taking the course pass-fail should complete the exercises. Students taking the course for a letter grade should complete both the exercises and the problems.

(2) All other instructions are the same as in previous problem sets.

Lecture 13 Exercises

Exercise 44

The point of this exercise is to extend the analysis from lecture to the case of n points in d dimensions (in lecture, $d = 2$). Assume that every point's density function f_i has support in $[0,1]^d$, with $f_i(x) \leq \frac{1}{\sigma}$ for every $x \in [0,1]^d$. The objective function is again to minimize the total ℓ_1 distance of the tour, with the distance between two points x, y defined as $\|x - y\|_1 = \sum_{i=1}^{d} |x_i - y_i|$.

Prove that for every fixed constant d, the expected running time of the 2-OPT local search algorithm (defined as in lecture) is $O(\sigma^{-1} n^6 \log n)$. Roughly what is the dependence of your bound on d (polynomial, exponential, doubly exponential, etc.)?

Lecture 14 Exercises

Exercise 45

For a Knapsack instance with n items, let PC_i denote the Pareto curve of the first i items. That is, a subset $S \subseteq \{1, 2, \ldots, i\}$ belongs to PC_i if and only if it is not dominated by any other subset of $\{1, 2, \ldots, i\}$. Explain how to implement a subroutine that, given as input the set PC_i sorted by total weight, produces PC_{i+1} sorted by total weight, in time $O(|PC_i| + |PC_{i+1}|)$. Conclude that the Pareto curve of a Knapsack instance can be generated in time $O(\sum_{i=0}^{n} |PC_i|)$.

[Hint: how long does it take to merge two sorted lists?]

Exercise 46

Prove that, in the worst case, the size of the Pareto curve of a Knapsack instance with n items can be 2^n.

Problems

Problem 22

(15 points) Recall that in Lecture #14 we proved that the smoothed complexity of the Pareto curve of Knapsack solutions is $O(n^2/\sigma)$, where $1/\sigma$ is a density bound on the distributions over the item weights. Prove that there are distributions for which this upper bound is tight. You can focus on the case where $\sigma = \Theta(1)$ and get full credit; if you're feeling keen, consider also smaller values of σ.

Problem 23

(20 points) Recall the Max Cut problem, where the input is an undirected graph in which the edges have nonnegative weights, and the goal is to compute a cut that maximizes the total weight of the crossing edges. Given a cut, the allowable *local moves* are to pick one vertex and move it to the opposite side of the cut. (In addition, both sides of the cut must stay non-empty.) *Local search* means repeatedly making local moves that strictly increase the weight of the cut until a *local optimum* — a cut from which there are no improving local moves — is reached. It turns out that, in the worst case, local search can require an exponential number of iterations to reach a locally optimal cut.

Let's consider the smoothed complexity of local search for the Max Cut problem (analogous to that of the 2-OPT heuristic that we studied in Lecture #13). Suppose each edge weight is drawn independently from a probability distribution with a density function that is bounded everywhere by $1/\sigma$, where σ is a parameter. Suppose also that the graph has maximum degree $O(\log n)$, where n is the number of vertices. Prove that local search has polynomial smoothed complexity in such instances, meaning that the expected running time (over the random edge weights) to reach a locally optimal cut is polynomial in n and $1/\sigma$.

[Hints: Consider first a fixed local move — what is the probability that it is an improving move that makes very little progress? Second, given that the maximum degree is small, how many "fundamentally distinct" local moves are there?]