CS369N: Problem Set #4 (Aka Take-Home Final)

Due to the instructor by noon on Friday, December 16, 2011

Instructions: Same as the first homework.

Problem 16

This problem studies a way of combining algorithms that are optimal in different senses (e.g., one optimized for the worst case and another optimized for the average case). As a concrete application, suppose a sequence of jobs arrive one-by-one, online. When a job j shows up, we learn its processing time p_{ij} on each of m machines i. The goal is to schedule all of the jobs on the machines to minimize the makespan (the biggest load on a machine, where the load on a machine is the sum of processing times of jobs assigned to it).

The goal is to get the best of both worlds of two different online algorithms for the problem, A and B. The motivation is that A is an online algorithm with good worst-case competitive ratio (like $O(\log n)$) but no knowledge about the input; while the algorithm B has a guess as to what the jobs and their processing times will be and, if the guess is correct, will produce an accordingly excellent schedule (say within $O(1)$ of optimal). But B may have terrible performance if its guess is wrong.

(a) (10 points) Given two online algorithms A and B as above (deterministic, say), show how to combine them into a single "master algorithm" C with the following property: for every input z,

$$\text{cost}(C, z) \leq 2 \cdot \min\{\text{cost}(A, z), \text{cost}(B, z)\}.$$

[Hint: imagine running A and B in parallel, and consider very simple rules to decide which one C should pay attention to.]

(b) (5 points) Show by counterexample that the constant 2 in (a) cannot be improved in general for this problem.

Problem 17

(12 points) Recall from Lecture #16 that we proved the following (the Leftover Hash Lemma). Suppose X is a random variable with collision probability $cp(X)$ at most $1/K$. Suppose $\mathcal{H}$ is a (2-)universal family of hash functions (from the range of X to the range $\{0, 1, 2, \ldots, M-1\}$), and h is chosen uniformly at random from $\mathcal{H}$. Then the statistical distance between the joint distribution of $(h, h(X))$ and of the uniform distribution (on $\mathcal{H} \times \{0, 1, 2, \ldots, M-1\}$) is at most $\frac{1}{2}\sqrt{M/K}$.

For this problem, assume that you have a sequence $X_1, \ldots, X_T$ of random variables, with the property that for every i and fixed values of $X_1, \ldots, X_{i-1}$, the (conditional) collision probability of X_i is at most $1/K$ (i.e., a "block source"). Prove that the statistical distance between the joint distribution of $(h, h(X_1), \ldots, h(X_T))$ and of the uniform distribution is at most $\frac{T}{2}\sqrt{M/K}$.

[Hint: One high-level approach is to prove, by downward induction on i, a bound of $\frac{(T-i)}{2}\sqrt{M/K}$ on the statistical distance between $(h, h(X_{i+1}), \ldots, h(X_T))$ and the uniform distribution for every fixed value of $X_1, \ldots, X_i$. The increase in statistical distance in the inductive step should come from the Triangle Inequality.]

Problem 18

(20 points) You are given n points $x_1, \ldots, x_n$ in some bounded real interval ($[0, 1]$, if you like) and a parameter k. The goal is to partition the n points into k clusters $C_1, \ldots, C_k$ and designate points $m_1, \ldots, m_k \in \mathcal{R}$ as cluster centers to minimize $\Phi = \sum_{i=1}^{k} \sum_{x_j \in C_i} (x_j - m_i)^2$. One can easily check that, given the C_i's, the optimal thing to do is to set m_i equal to the average value of the points in C_i.

In this problem we will analyze a particular local search heuristic, which works as follows. Iteration 0 begins with an arbitrary clustering $C_1, \ldots, C_k$ with each C_i non-empty. In an odd iteration, we hold the C_i's fixed and re-compute m_i as the average value of the points in C_i. In an even iteration, we independently and simultaneously re-assign each point x_j to the cluster C_i that had mean m_i closest to x_j. You should check that every non-vacuous iteration (i.e., one that makes some change) strictly decreases Φ. Thus, this heuristic is guaranteed to terminate (with a "locally optimal" clustering). Prove that the heuristic has "almost polynomial smoothed complexity", meaning that for every point set $x_1, \ldots, x_n$, if an independent (one-dimensional) Gaussian with standard deviation σ is added to each x_i, then the running time of the local search heuristic is polynomial in n, k, and $1/\sigma$, with high probability (over the perturbation).

[Hint: You might look to the analysis of the 2-OPT heuristic for TSP for inspiration. Try to identify a sufficient condition on the input that guarantees that every improving local move makes a non-trivial improvement to Φ, and prove probability bounds on the likelihood that the condition is satisfied.]

Problem 19

(15 points) Recall that in Lecture #18 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 (or alternatively, if you go back and check, on the item values). 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 20

(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 #17). 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 $1/\sigma$.

[Hint: how many "fundamentally distinct" local moves are there?]

[**Extra credit:** extend the result to graphs with arbitrarily large maximum degree.]