Advanced Algorithms 2020 11/06, 2020

Exercise 08

Lecturer: Mohsen Ghaffari *Teaching Assistang: Vaclav Rozhon*

1 Fractional Set Cover via Multiplicative Weights

Recall that during the lecture you have seen an algorithm that (approximately) solves an LP given by $c^\top x = K, Ax \geq b, x \geq 0$ for a nonnegative K and a vector c and a matrix A with nonnegative entries. The solution $\tilde{x}$ the algorithm found satisfied the constraint only approximately, i.e., $A\tilde{x} \geq b - \delta\mathbb{1}$.

The algorithm used multiplicative weights: in step t, it uses a nonnegative weight vector p_t with one entry for each row of the matrix A. At the beginning, $p_1 = \mathbb{1}$. In the step t, we solve a simpler problem $c^\top x = K, p_t^\top Ax \geq p_t^\top b, x \geq 0$. Then, we update the weight vector p_t with the multiplicative update step: we multiply the i-th entry of p_t by $1 - \varepsilon(A_i\tilde{x}_t - b_i)/\rho$ for $\varepsilon = \delta/4\rho$ and $\rho = \max_{i,x}(1, |A_i x - b_i|)$ where the maximum is taken over all rows of the matrix A and all x satisfying $c^\top x = K, x \geq 0$. Intuitively, whenever we satisfied the condition $A_i x \geq b_i$ for the i-th row, we make its weight smaller in the next step, while the weight of unsatisfied rows is getting bigger. Finally, we output the average solution $\tilde{x} = (\sum_{t=1}^T \tilde{x}_t)/T$. The number of steps needed is $T = O(\rho^2(\log m)/\delta^2)$.

In this exercise, we want to understand what exactly happens in the case of fractional set cover problem, where you are given some set of elements E and a family of its subsets $\{S_1, S_2, \ldots, S_m\}$. You have to pick a fraction $0 \leq x_S \leq 1$ of each set S so that for each element e we have $\sum_{S, e\in S} x_S \geq 1$, i.e., each element is in total covered by at least one set. We wish to minimize the sum $\sum x_S$, i.e., the total weight of fractional sets needed.

1. Write down the LP for the fractional set cover. Are all constraints in your LP needed? Also, give an example where the solution of the fractional set cover problem has smaller total weight $\sum x_S$ than the respective solution of the set cover problem.
2. LP for fractional set cover is a special case of covering LP that we can solve by the multiplicative weights algorithm. What can you say about the solution to the simpler problem that is solved in each step of the algorithm?
3. The final algorithm uses binary search to find the best value of K. What is a reasonable upper bound on K? How many steps of the algorithm are needed?
4. We would like to get a $(1+\varepsilon)$-approximation algorithm to the fractional set cover problem. The solution that our algorithm outputs is not such a solution. How can you fix it?

2 Smallest Enclosing Circle

In the smallest enclosing circle problem, we are given n points in the plane and want to find the smallest circle that encloses all of them. Here is a randomized algorithm for this problem: in each step, we sample a subset R of 20 points uniformly at random and find their minimum enclosing circle (e.g. by bruteforce). If the circle encloses all the points, we finish. Otherwise, we duplicate each point outside that circle and repeat the algorithm on the new (enlarged) set of points.

Your task is to analyze how many steps the algorithm needs in expectation. You can use the following geometric lemma:

Lemma 1 *Suppose we have a set S of n (not necessarily distinct) points and we choose a subset R of them uniformly at random. Then the expected number of points from S that lie outside the smallest enclosing circle of R is at most $3n/(|R|+1)$.*

Another potentially helpful fact is that for a random variable T that takes positive integer values we have $\mathbb{E}[T] = \sum_{t=1}^{\infty} P[X \geq t]$.