PRINCETON UNIVERSITY FALL '19 COS 521:ADVANCED ALGORITHMS

Homework 4

Out: *Nov 4* Due: *Nov 18*

Instructions:

- Upload your solutions (to the non-extra-credit) as *a single* PDF file (one PDF total) to Mechanical TA. Please anonymize your submission (do not list your name in the PDF title or in the document itself). If you forget, it's OK.
- If you choose to do extra credit, upload your solution to the extra credits as a single PDF file to Mechanical TA. Please again anonymize your submission.
- You may collaborate with any classmates, textbooks, the Internet, etc. Please attach a brief "collaboration statement" listing any collaborators at the end of your PDF (if you forget, it's OK). You should write up your solutions individually.
- For each problem, you should aim to keep your writeup below one page. For some problems, this may be infeasible, and for some problems you may write significantly less than a page. This is not a hard constraint, but part of the assignment is figuring out how to easily convince the grader of correctness, and to do so concisely. "One page" is just a guideline: if your solution is longer because you chose to use figures (or large margins, display math, etc.) that's fine.
- Each problem is worth twenty points (even those with multiple subparts).

§1 Consider a set of n objects (images, songs, etc.) and suppose somebody has designed a *distance* function $d(\cdot)$ among them where $d(i,j)$ is the distance between objects i and j. We are trying to find a geometric realization of these distances. Of course, exact realization may be impossible and we are willing to tolerate a factor 2 approximation. We want n vectors $u_1, u_2, \ldots, u_n$ such that $d(i,j) \leq \|u_i - u_j\|_2 \leq 2d(i,j)$ for all pairs i, j. Describe a polynomial-time algorithm that determines whether such u_i's exist (and outputs them in the event that they do).

§2 Given black-box access to a poly-time algorithm $\mathcal{A}_P$ that optimizes linear functions over the convex, compact region P, and poly-time $\mathcal{A}_Q$ that optimizes linear functions over the convex, compact region Q, design a poly-time algorithm that optimizes linear functions over the convex, compact region $P \cap Q$.

§3 Describe separation oracles for the following convex sets. Your oracles should run in linear time, assuming that the given oracles run in linear time (so you can make a constant number of black-box calls to the given oracles).

(a) The ℓ_1 ball, $\{x : \|x\|_1 \leq 1\}$. Recall that $\|x\|_1 = \sum_{i=1}^{d} |x_d|$.

(b) Any convex set A that we have a projection oracle for. I.e. we have an oracle to compute $\arg\min_{x \in A} \|x - y\|_2$ for any y.

(c) The ϵ-neighborhood, E of any convex set A:

$$E = \{x : \exists y \in A \text{ with } \|x - y\|_2 \leq \epsilon\},$$

given a projection oracle for A.

§4 Define a *corner* of a convex, compact region P to be any $x \in P$ that cannot be written as a convex combination of other points in P.[1] Given black-box access to a poly-time algorithm $\mathcal{S}_P$ that is a separation oracle for convex region $P \in \mathbb{R}^n$, design a poly-time algorithm that takes as input a point x and writes x as a convex combination of corners of P. That is, output a list $\{(c_1, y_1), \ldots, (c_{n+1}, y_{n+1})\}$ such that each y_i is a corner of P, each $c_i \geq 0$, and $\sum_i c_i = 1$.

§5 The maximum cut problem asks us to cluster the nodes of a graph $G = (V, E)$ into two disjoint sets X, Y so as to maximize the number of edges between these sets:

$$\max_{X,Y} \sum_{(i,j)\in E} \mathbb{1}[(i \in X, j \in Y) \vee (i \in Y, j \in X)]$$

Consider instead clustering the nodes into **three** disjoint sets X, Y, Z. Our goal is to maximize the number of edges between different sets:

$$\max_{X,Y,Z} \sum_{(i,j)\in E} \mathbb{1}[(i \in X, j \in Y \cup Z) \vee (i \in Y, j \in X \cup Z) \vee (i \in Z, j \in X \cup Y)]$$

Design an algorithm based on SDP relaxation that solves this problem with approximation ration greater then .7.

Note: In the Goemans-Williamson algorithm for maximum cut, we claimed that $\frac{2\theta}{\pi(1-\cos\theta)} \geq 0.878,\ \forall\theta \in [0, \pi]$. This is much easier to verify analytically (e.g. with a plot in MATLAB) than to prove formally. If similar quantities appear in your proof, feel free to bound them analytically, without proof.

Obtain the highest object value you can – partial credit will be given to any non-trivial solution, even if it obtains a weaker bound than .7.

Extra Credit:

§1 (Extra Credit, follows "The maximum cut problem,...") Obtain an algorithm with approximation factor $> .8$.

§2 (Extra Credit) Consider the following variant on the secretary problem: an adversary puts the elements into any order they desire. Then, instead of being randomly permuted, the elements are revealed either in order, or in reverse order, each with probability 1/2 (everything else is the same: upon seeing an element, you must immediately and irrevocably accept or reject). Prove that no algorithm can guarantee acceptance of the heaviest element with probability $> 1/n$ when there are n elements.

[1] So for example, if P is a triangle, P has three corners. If P is a circle, it has infinitely many.

§3 (Extra Credit) Consider the following variant on prophet inequalities: instead of each X_i being independently drawn, there is a joint distribution over $(X_1, \ldots, X_n)$ (everything else is the same: you know the joint distribution, the random variables X_i are revealed to you in order, and you must immediately accept/reject upon seeing). Prove that no algorithm can guarantee better than $\mathbb{E}[\max_i X_i]/n$.