PRINCETON UNIVERSITY FALL '13 COS 521:ADVANCED ALGORITHMS

Homework 4

Out: *Dec 2* Due: *Nov 12*

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.

2. Suppose we have a set of n images and for some multiset E of image pairs we have been told whether they are *similar* (denote +edges in E) or *dissimilar* (denoted −edges). These ratings were generated by different users and may not be mutually consistent (in fact the same pair may be rated as + as well as −). We wish to *partition* them into clusters $S_1, S_2, S_3, \ldots$ so as to maximise:

   (# of +edges that lie within clusters) + (# of −edges that lie between clusters).

   Show that the following SDP is an upperbound on this, where $w^+(ij)$ and $w^-(ij)$ are the number of times pair i,j has been rated + and − respectively.

$$\max \sum_{(i,j)\in E} w^+(ij)(x_i \cdot x_j) + w^-(ij)(1 - x_i \cdot x_j)$$
$$|x_i|_2^2 = 1 \qquad \forall i$$
$$x_i \cdot x_j \geq 0 \qquad \forall i \neq j.$$

3. For the problem in the previous question describe a clustering into 4 clusters that achieves an objective value 0.75 times the SDP value. (Hint: Use Goemans-Williamson style rounding but with two random hyperplanes instead of one. You may need a quick matlab calculation just like GW.)

4. Prove von Neumann's minimax theorem. (You can assume LP duality.)

5. Suppose you are given m halfspaces in $\Re^n$ with rational coefficients. Describe a polynomial-time algorithm to find the largest *sphere* that is contained inside the polyhedron defined by these halfspaces.

6. Let f be an n-variate convex function such that for every x, every eigenvalue of $\nabla^2 f(x)$ lies in $[m, M]$. Show that the optimum value of f is lowerbounded by $f(x) - \frac{1}{2m}|\nabla f(x)|_2^2$ and upperbounded by $f(x) - \frac{1}{2M}|\nabla f(x)|_2^2$, where x is any point. In other words, if the gradient at x is small, then the value of f at x is near-optimal. (Hint: By the mean value theorem, $f(y) = f(x) + \nabla f(x)^T(y-x) + \frac{1}{2}(y-x)^T\nabla^2 f(z)(y-x)$, where z is some point on the line segment joining x, y.)

7. (Extra credit) Show that approximation the number of simple cycles within a factor 100 in a directed graph is NP-hard. (Hint: Show that if there is a polynomial-time algorithm for this task, then we can solve the Hamiltonian cycle problem in directed graphs, which is NP-hard. Here the exact constant 100 is not important, and can even be replaced by, say, n.)

8. (Extra credit) (*Sudan's list decoding*) Let $(a_1, b_1), (a_2, b_2), \ldots, (a_n, b_n) \in F^2$ where $F = GF(q)$ and $q \gg n$. We say that a polynomial $p(x)$ *describes* k of these pairs if $p(a_i) = b_i$ for k values of i. This question concerns an algorithm that recovers p even if $k < n/2$ (in other words, a majority of the values are wrong).

   (a) Show that there exists a bivariate polynomial $Q(z, x)$ of degree at most $\lceil \sqrt{n} \rceil + 1$ in z and x such that $Q(b_i, a_i) = 0$ for each $i = 1, \ldots, n$. Show also that there is an efficient (poly(n) time) algorithm to construct such a Q.

   (b) Show that if $R(z, x)$ is a bivariate polynomial and $g(x)$ a univariate polynomial then $z - g(x)$ divides $R(z, x)$ iff $R(g(x), x)$ is the 0 polynomial.

   (c) Suppose $p(x)$ is a degree d polynomial that describes k of the points. Show that if d is an integer and $k > (d + 1)(\lceil \sqrt{n} \rceil + 1)$ then $z - p(x)$ divides the bivariate polynomial $Q(z, x)$ described in part (a). (Aside: Note that this places an upperbound on the number of such polynomials. Can you improve this upperbound by other methods?)

   (There is a randomized polynomial time algorithm due to Berlekamp that factors a bivariate polynomial. Using this we can efficiently recover all the polynomials p of the type described in (c). This completes the description of Sudan's algorithm for *list decoding*.)