6.854J / 18.415J Advanced Algorithms
Fall 2008

18.415/6.854 Advanced Algorithms

Problem Set 4

Problem 1. In a 0-*sum 2-player game*, Alice has a choice of n so-called *pure* strategies and Bob has a choice of m *pure* strategies. If Alice picks strategy i and Bob picks strategy j, then the *payoff* is a_{ij}, meaning a_{ij} dollars are transfered from Alice to Bob. So Bob makes money if a_{ij} is positive, but Alice makes money if a_{ij} is negative. Thus, Alice wants to pick a strategy that minimizes the payoff while Bob wants a strategy that maximizes the payoff. The matrix $A = (a_{ij})$ is called the *payoff matrix*.

It is well known that to play these games well, you need to use a *mixed* strategy—a random choice from among pure strategies. A mixed strategy is just a particular probability distribution over pure strategies: you flip coins and then play the selected pure strategy. If Alice has mixed strategy x, meaning he plays strategy i with probability x_i, and Bob has mixed strategy y, then it is easy to prove that the expected payoff in the resulting game is $x^T Ay$. Alice wants to minimize this expected payoff while Bob wants to maximize it. Our goal is to understand what strategies each player should play.

We'll start by making the pessimal assumption for Alice that whichever strategy she picks, Bob will play best possible strategy against her. In other words, given Alice's strategy x, Bob will pick a strategy y that achieves $\max_y x^T Ay$. Thus, Alice wants to find a distribution x that minimizes $\max_y x^T Ay$. Similarly, Bob wants a y to maximize $\min_x x^T Ay$. So we are interested in solving the following 2 problems:

$$\min_{x:\sum x_i=1, x\geq 0} \max_{y:\sum y_j=1, y\geq 0} x^T Ay$$

$$\max_{y:\sum y_j=1, y\geq 0} \min_{x:\sum x_i=1, x\geq 0} x^T Ay$$

Unfortunately, these look like nonlinear programs!

1. Show that if Alice's mixed strategy is known, then Bob has a pure strategy serving as his best response.

2. Show how to convert each program above into a linear program, and thus find an optimal strategy for both players in polynomial time.

3. Use strong duality (applied to the LP you built in the previous part) to argue that the above two quantities are *equal*.

The second statement shows that the strategies x and y, besides being optimal, are in *Nash Equilibrium*: even if each player knows the other's strategy, there is no point in changing strategies. This was proven by Von Neumann and was actually one of the ideas that led to the discovery of strong duality.

Problem 2. Consider the linear program

$$\min \sum_j x_j$$

subject to

$$\sum_j a_{ij} x_j \geq 1 \quad \forall i$$

$$x_j \geq 0 \quad \forall j$$

and its dual

$$\max \sum_i y_i$$

subject to

$$\sum_i a_{ij} y_i \leq 1$$

$$x_i \geq 0.$$

Assume that $A = [a_{ij}]$ is $m \times n$ and has only *nonnegative* entries.

In this problem, you'll have to show that a continuous algorithm solves (almost miraculously) the above pair of dual linear programs. We shall define a series of functions whose argument is the "time" and you'll show that some of these functions tend to the optimal solution as time goes to infinity. (For simplicity of notation, we drop the dependence on the time.)

- Initially, we let $s_j = 0$ for $j = 1, \ldots, n$ and $LB = 0$. The vector s will (sort of) play the role of primal solution, and LB the role of a lower bound on the objective function.

- At any time, let

  $$t_i = e^{-\sum_j a_{ij} s_j}$$

  for $i = 1, \ldots, m$. Also, let $d_j = \sum_i a_{ij} t_i$ for $j = 1, \ldots, n$, $D = \max_j d_j$ and k be an index j attaining the maximum in the definition of D. The algorithm continuously increases s_k.

Observe that when s_k is increased, the vectors t and d as well as D change also, implying that the index k changes over time.

1. Let $\alpha = \min_i(\sum_j a_{ij}s_j)$. Let $x_j = s_j/\alpha$ for $j = 1, \ldots, n$, $y_i = t_i/D$ for $i = 1, \ldots, m$ and $LB = \max(LB, \frac{\sum t_i}{D})$. Show that x is primal feasible, y is dual feasible and LB is a lower bound on the optimal value of both primal and dual.

2. Prove that
$$\sum_{i=1}^{m} t_i \leq me^{-\sum_{j=1}^{n} s_j/LB}.$$
Hint: Show that initially the inequality holds and that it is also maintained whenever we have equality.

3. Deduce from (b) that $\sum_i t_i$ tends to 0 as time goes to infinity.

4. Using (b), give an upper bound on the value of the primal solution x, and using (c), show that this upper bound tends to LB as time goes to infinity. This shows that as time goes to infinity, both x and y tend to primal and dual optimal solutions!

Problem 3. We would like to find a function $f(n)$ such that, given any set of n (possibly negative) numbers, $c_1, \cdots, c_n$, one cannot find more than $f(n)$ subsums of these numbers which decrease in absolute value by a factor of at least 2. More formally:

Lemma 1 *Let $c \in \mathbb{R}^n$ and $y_k \in \{0,1\}^n$ for $k = 1, \ldots, q$ such that $2|y_{k+1}^T c| \leq |y_k^T c|$ for $k = 1, \ldots, q-1$. Assume that $y_q^T c = 1$. Then $q \leq f(n)$.*

Using linear programming, you are asked to prove that $f(n) = O(n \log n)$.

1. Given a vector c and a set of q subsums satisfying the hypothesis of the Lemma, write a set of inequalities in the variables $x_i \geq 0, i = 1 \ldots n$, such that $x_i = |c_i|$ is a feasible vector, and for any feasible vector x' there is a corresponding vector c' satisfying the hypothesis of the Lemma for the same set of subsums.

2. Prove that there must exist a vector c' satisfying the hypothesis of the Lemma, with c' of the form $(d_1/d, d_2/d, \ldots, d_n/d)$ for some integers $|d|, |d_1|, \ldots, |d_n| = 2^{O(n \log n)}$.

3. Deduce that $f(n) = O(n \log n)$.

4. (Not part of the problem set; only for those who find the problem sets too easy...) Show that $f(n) = \Omega(n \log n)$ (as a tiny step, can you find a set of numbers such that $f(n) > n$?).

Problem 4. Let K be a bounded convex set in $\mathbb{R}^n$. In this problem, you'll prove that there exists an ellipsoid E contained within K such that if you blow it up by a factor of n (the dimension) then the corresponding ellipoid contains K; in short, $E \subseteq K$ and $K \subseteq nE$.

1. Suppose that we have an ellipsoid $E(a, A) = \{x \in \mathbb{R}^n : (x-a)^T A^{-1}(x-a) \leq 1\}$ and we have a point $b \notin nE(a, A)$. Argue that the convex hull of b and $E(a, A)$, $conv(\{b\}, E(a, A))$, contains an ellipsoid E' of larger volume than $E(a, A)$.

   (You do not need to explicitly give a' and A' corresponding to $E' = E(a', A')$, if that helps. It might be easier to deal with a particular case for a, A and b, and argue why you can.)

2. Argue that the maximum volume ellipsoid E contained in K (it is actually unique, although you do not need this) is such that $nE \supseteq K$.

3. (Optional. Assume that $K = \{x \in \mathbb{R}^n : Cx \leq d\}$ is bounded, where C is $m \times n$ and $d \in \mathbb{R}^m$. Formulate the problem of finding the largest volume ellipsoid contained within K as a convex program (minimizing a convex function over a convex set, or maximizing a concave function over a convex set. One could therefore use the ellipsoid algorithm to find (a close approximation to) this maximum volume ellipsoid.)

Problem 6. Given an undirected graph $G = (V, E)$, a set $T \subseteq V$ with $|T|$ even and a weight function $w : E \to \mathbb{Q}_+$, the minimum (weight) T-cut problem is to find $S \subseteq$ with $|S \cap T|$ odd[1] such that $d(S) := w(S : \bar{S}) := \sum_{e \in (S:\bar{S})} w_e$ is minimized. Here $(S : \bar{S})$ denotes the set of edges of E with exactly one endpoint in S (since our graph is undirected, observe that $d(S) = d(\bar{S})$). For $T = \{s, t\}$, the minimum T-cut problem reduces to the minimum $s - t$ cut problem (in an undirected graph). In this problem, you will show that the minimum T-cut problem can be solved efficiently.

1. Argue that the minimum $s - t$ cut problem in an *undirected* graph G can be solved efficiently by using an algorithm for a minimum $s - t$ cut problem in a *directed* graph H.

2. A $T - T$ cut is a cut $(S : \bar{S})$ with $S \cap T \neq \emptyset$ and $\bar{S} \cap T \neq \emptyset$. Show that the minimum weight $T - T$ cut can be obtained by solving a polynomial number of minimum $s - t$ cut problems. Can you do it with $O(|T|)$ such minimum $s - t$ cut computations?

3. Prove that for any $A, B \subseteq V$, we have

$$d(A) + d(B) \geq d(A \cap B) + d(A \cup B).$$

[1] thus, $\bar{S} \cap T| = |T \setminus S|$ is also odd.

4. To solve the minimum T-cut problem, suppose we first solve the minimum $T - T$ cut problem and obtain the cut $(S : \bar{S})$. If $|S \cap T|$ is odd, we are done (right?). If $|S \cap T|$ is even, use the previous inequality to argue that there exists a minimum T-cut $(C : \bar{C})$ such that $C \subseteq S$ or $C \subseteq \bar{S}$. Deduce from this an efficient algorithm for solving the minimum T-cut problem. How many calls to your minimum $T - T$ cut algorithm are you using?