CS787 Final Exam

Mikola Lysenko

May 11, 2009

1. We construct a stable matching using the method of Gale and Shapley. The algorithm proceeds in rounds, where in each round each unmatched boy propositions his highest ranked girl which he has not yet asked. If the girl he propositions is not currently engaged or if the boy she is currently engaged to is ranked lower in her preferences than the boy that asked her, the girl becomes engaged to the new boy. This process is repeated until all girls and boys are matched.

This construction assures that at the end all people are matched, since each girl will eventually be propositioned (there are equal boys and girls, so it will happen eventually) and once she is propositioned she is guaranteed a partner. The matching is also stable since for each boy there is no available partner he would prefer more than the one he ultimately selects – otherwise he would have selected that partner earlier and remained matched to her. So, we have shown that a stable perfect matching always exists.

Since the input must encode $2n$ lists of n unique ordered elements (one list of preferences per person), no algorithm may proceed faster than $\Omega(n \log(n!)) = \Omega(n^2 \log(n))$ or $\Omega(n^2)$ on a RAM machine. Since the algorithm considers the i^{th} preference of each person at most once because each boy propositions each girl at most once. As a result, the algorithm is optimal in the sense that it performs a constant amount of work per each bit of input.

2. To prove that S is a vertex cover, we perform a case-by-case analysis using the fact that the DFS gives a 4-partition of the edges in G:

Tree Edges: Tree edges are edges between nodes of the DFS, and so they are trivially covered by S.

Forward / Back Edges: Forward/back edges connect nodes to their descendants/ancestors. If a node has any descendants/is an ancestor, then it is not a leaf and therefore in S, so these edges are covered by S.

Cross Edges: Cross edges are edges between nonrelated nodes of the DFS tree. A cross edge may not appear between two leaf vertices in an undirected graph, because if this were the case then we would have taken the edge in the DFS tree upon reaching the earlier of the two vertices and thus one of them could not have been a leaf. Therefore, cross edges only occur between internal nodes and are thus covered.

In the worst case, the size of the depth first search tree is $O(|V|+|E|)$. This cost dominates the cost of the depth first search as well the construction of the cover, and so it bounds the total running time of the algorithm.

Finally, any vertex cover of G must also be a vertex cover of the tree edges, T, of the DFS tree (by definition). Since T is a bipartite graph, covering the tree requires at least K vertices, where K is the number of vertices in the smaller bipartite component (Konig's theorem). We also know that excluding the leaves, L of T, each vertex in T has at least one parent and one child so by the pidgeon hole principle the smaller bipartite component must be of size $K \geq \frac{|T|-|L|}{2}$. Because $|S| = |T| - |L| = 2K \geq |\text{Minimal Vertex Cover}|$, S is a factor-2 approximation of the minimal vertex cover.

3. Without loss of generality, assume that there are no equations of the form $0 = b$, (ie all coefficients $a_i = 0$). These equations are trivially (un)satisfiable and could be dealt with in linear time preprocessing.

To answer the first part, since $\mathbb{Z}_2$ is a field, we can use Gaussian elimination to solve for x whenever $m \leq n$ in $O(n^3)$ time[1] guaranteeing that all equations are satisfied.

For the general case, the problem becomes more difficult. Consider the expected number of equations satisfied by a uniform distribution of assignments X. Because the parity of a uniform collection of boolean variables is also a uniform boolean variable, for any equation A:

$$E(A|X) = \frac{1}{2}$$

Where $E(A|X)$ denotes the expectation of satisfying A given the distribution X (by abuse of notation). Let A_j denote the j^{th} equation with by linearity:

$$E(\sum_{j=1}^{m} A_j|X) = \sum_{j=1}^{m} E(A_j|X) = m/2$$

We also know that

$$m/2 \leq OPT \leq m$$

where OPT is the number of equations satisfied by the optimal assignment, thus uniform random assignment gives an approximation within $OPT/2$.

This method can be derandomized in a way analogous to the derandomized approximation of Max-SAT. For $i = 1$ to n, greedily assign to each variable x_i the value that maximizes the expectation of performing a uniform assignment on the remaining variables (equivalent to minimizing the number of equations which are rendered unsatisfiable by x_i). We argue inductively that this gives a guaranteed factor $1/2$ approximation based on the observation that it is at

[1]Technically $O(n^\omega)$, where $2 \leq \omega \leq \log_2 5$ is the matrix multiplication constant

least as good as the expectation of a uniform random assignment. For the sake of brevity, let $M = \sum_{i=1}^{j} A_j$

Initially, the choice of x_1 satisfies

$$E(M|x_1X') \geq E(M|\neg x_1X')$$

with X' denoting a uniform assignment of the remaining variables and $\neg x_1$ denoting the opposite assignment of x_1. By definition:

$$\frac{1}{2}E(M|x_1X') + \frac{1}{2}E(M|\neg x_1X') = E(M|X)$$

Which combined with the above gives:

$$E(M|x_1X') \geq E(M|X)$$

Now suppose at stage i,

$$E(M|x_1x_2..x_{i-1}X') \geq E(M|X)$$

Then we know

$$\begin{aligned} E(M|x_1x_2...x_iX') &\geq E(M|x_1x_2...\neg x_iX') \\ \frac{1}{2}E(M|x_1x_2...x_iX') + \frac{1}{2}E(M|x_1x_2...\neg x_iX') &= E(M|x_1x_2..x_{i-1}X') \end{aligned}$$

Substitution by the inductive hypothesis and the above equation gives

$$E(M|x_1x_2...x_iX') \geq E(M|X)$$

And so we are done.

4. There are several components to this question which must be dealt with individually:

i First, we show that given some R-strategy, x, there is a pure optimal C-strategy, y (that is there exists some i such that $y_i = 1$). To do this observe that the payoff function is linear and that the space of strategies for C is the convex polytope $S_C = \{y| \sum y_i = 1, y_i \geq 0\}$, with vertices unit Cartesian vectors in $\Re^n$. By a well known theorem from convex optimization[2], the extrema for a linear function over a convex polytope occur at the vertices (if they exist). Since the set S_C is non-empty and compact, such a strategy exists and therefore there exists some pure strategy for C which is optimal.

ii Recall the definition of R's optimal payoff

$$\max_{x \in S_R} \min_{y \in S_C} \sum_{i=1}^{m} \sum_{j=1}^{n} x_i a_{ij} y_j$$

[2]Stephen Boyd, Nonsmooth optimization lecture notes

But from part i, we know the extrema for $\min_y$ are Cartesian unit vectors, this simplifies to:

$$\max_{x \in S_R} \min_{1 \leq j \leq n} \sum_{i=1}^{m} a_{ij} x_i$$

iii Now we come to the crux of the problem, which is to show that R's optimal strategy is given by the solution to the following linear program:

$$\text{maximize } z$$

$$\text{subject to } \forall 1 \leq j \leq n : z - \sum_{i=1}^{m} a_{ij} x_i \leq 0$$

$$\sum_{i=1}^{m} x_i = 1$$

$$\forall 1 \leq i \leq m : x_i \geq 0$$

To do this, we first show that z is R's optimal payoff. Let v denote the optimal payoff obtained from part ii,

$$v = \max_{x \in S_R} \min_{1 \leq j \leq n} \sum_{i=1}^{m} a_{ij} x_i$$

If $z > v$, then there must exist some j such that $z > \max_x \sum_{i=1}^{m} a_{ij} x_i$. However, this would violate one of the constraints so $z \leq v$. Now, if $z < v$ we would conclude that for all constraints j, $z < \max_x \sum_{i=1}^{m} a_{ij} x_i$, and since there are no other upper constraints on z, z must not be maximal. Thus, we are forced to pick $z = v$. As a corollary, the value of x which maximizes z must be an optimal strategy. Therefore, we conclude that this linear program does indeed solve the game for player R.

iv The dual of the above program is given by:

$$\text{minimize } z'$$

$$\text{subject to } \forall 1 \leq i \leq m : z' - \sum_{j=1}^{n} a_{ij} y_j \geq 0$$

$$\sum_{j=1}^{n} y_j = 1$$

$$\forall 1 \leq j \leq n : y_j \geq 0$$

We argue that the value z' in this equation is the same as the value z attained in the primal. Consider a pair of optimal solutions $(x, z), (y, z')$

(which exist since both programs are feasible and bounded), then primal/dual complementary slackness tells us:

$$\forall 1 \le i \le m : x_i(z' - \sum_{j=1}^{n} a_{ij} y_j) = 0$$

$$\forall 1 \le j \le n : y_j(z - \sum_{i=1}^{m} a_{ij} x_i) = 0$$

Separating z, z' and taking sums over all i, j gives

$$z' = \sum_{i=1}^{m} x_i z' = \sum_{i=1}^{m} \sum_{j=1}^{n} x_i a_{ij} y_j = \sum_{j=1}^{n} y_j z = z$$

So $z = z'$, and by a symmetric argument to part iii y must be an optimal strategy for player C.

v The Minimax Theorem states:

> For every two person zero sum game, there exist strategies x, y for players R,C and some scalar z such that: for any column strategy, the payoff to R is at least z if R plays x; and for any row strategy, the cost to C is no more than z if C plays y.

This theorem exactly states the conclusion from parts iii and iv. The fact that x assures a payoff of at least z is equivalent to stating the optimality of x, and that y assures a payoff of no more than z is the same as stating the optimality of y. Existence conditions are trivial due to the feasibility of both programs.