CS364B: Exercise Set #4

Due by the beginning of class on Wednesday, February 5, 2014

Instructions:

(1) Turn in your solutions to all of the following exercises directly to one of the TA (Okke). Please type your solutions if possible and feel free to use the LaTeX template provided on the course home page. Email your solutions to `cs364b-win1314-submissions@cs.stanford.edu`. If you prefer to hand-write your solutions, you can give it to Okke in person at the start of the lecture.

(2) Your solutions will be graded on a "check/minus/zero" system, with "check" meaning satisfactory and "minus" meaning needs improvement.

(3) Solve the non-optimal exercises and write up your solutions on your own. You may, however, discuss the exercises verbally at a high level with other students. You may also consult any books, papers, and Internet resources that you wish. And of course, you are encouraged to contact the course staff (via Piazza or office hours) to clarify the questions and the course material.

(4) No late assignments will be accepted.

Lecture 7 Exercises

Exercise 22 (Optional – Do Not Hand In)

This exercise outlines a proof that there is no ascending auction that is guaranteed to converge to the VCG outcome for every profile of gross substitutes valuations. For this exercise, we define an *ascending auction* as a procedure that asks demand queries at a sequence $\mathbf{p}^0, \mathbf{p}^1, \ldots,$ of nondecreasing price vectors, where every demand query posed depends only on the answers to previous demand queries. (Thus, for each $\mathbf{p}^i$, the auction may as well pose a demand query at these prices to every bidder.) The final allocation and payments can be an arbitrary function of all demand query answers over the course of the auction (cf., the clinching auction for identical goods).

Now consider a set $U = \{a, b, c, d\}$ of four items. There are three bidders:

- The first bidder's valuation satisfies $v_1(a) = v_1(b) = v_1(ab) = 2$, $v_1(c) = v_1(d) = v_1(cd) = 2$, and $v_1(S) = v_1(S \cap \{a, b\}) + v_1(S \cap \{c, d\})$ for $S \subseteq U$.
- The second bidder's valuation satisfies $v_2(c) = x$, $v_2(a) = v_2(ac) = 1$, $v_2(d) = 0$, $v_2(b) = v_2(bd) = 1$, and $v_2(S) = v_2(S \cap \{a, c\}) + v_2(S \cap \{b, d\})$ for $S \subseteq U$. Here x is an unknown parameter between $\frac{1}{4}$ and $\frac{1}{2}$.
- The third bidder's valuation satisfies $v_3(a) = y$, $v_3(c) = v_3(ac) = 1$, $v_3(b) = 0$, $v_3(d) = v_3(bd) = 1$, and $v_3(S) = v_3(S \cap \{a, c\}) + v_3(S \cap \{b, d\})$ for $S \subseteq U$. Here y is an unknown parameter between $\frac{1}{4}$ and $\frac{1}{2}$.

(a) Argue that v_1, v_2, v_3 satisfy the gross substitutes condition.

[Hint: prove that the "direct sum" of two G.S. valuations is again G.S.]

(b) Argue that the VCG payments are 2 for bidder 1, y for bidder 2, and x for bidder 3.

(c) Argue that to compute the correct VCG payment x for bidder 3 (for every $x \in (\frac{1}{4}, \frac{1}{2})$), there must exist a price vector $\mathbf{p}^j$ with $p^j(c) \leq x$ and $p^j(a) - p^j(c) \geq 1 - x$.

(d) Argue that to compute the correct VCG payment y for bidder 2 (for every $y \in (\frac{1}{4}, \frac{1}{2})$), there must exist a price vector $\mathbf{p}^\ell$ with $p^\ell(a) \le y$ and $p^\ell(c) - p^\ell(a) \ge 1 - y$.

(e) Conclude from (d) and (e) that no ascending auction correctly computes the VCG payments for every value of x and y.

Exercise 23

Prove that the following are equivalent for a valuation v on a set $U = \{A, B\}$ of two items:

(i) v satisfies the gross substitutes condition;

(ii) v is submodular;

(iii) v is subadditive, meaning $v(\{A, B\}) \le v(\{A\}) + v(\{B\})$.

Exercise 24 (Optional – Do Not Hand In)

Recall that a *budgeted additive valuation* has the form $v(S) = \min\{B, \sum_{j \in S} v_j\}$, where $v_1, \ldots, v_m$ and B are nonnegative numbers. Note that every such valuation is submodular and can be described using a linear (in m) number of parameters.

Prove that computing a welfare-maximizing allocation for bidders with budgeted additive valuations is NP-hard, even when there are only two bidders.

[Hint: reduce from Partition.]

Exercise 25 (Optional – Do Not Hand In)

This exercise gives a second $\frac{1}{2}$-approximation algorithm for computing a welfare-maximizing allocation for bidders with monotone submodular valuations. It does not have the "auction-like" flavor of the approximation algorithm from lecture, but it is still very simple, and has the added benefits of needing only value queries (as opposed to demand queries) and running in polynomial (rather than pseudopolynomial) time.

The algorithm is greedy, and makes one pass through the items:

1. Order the items U arbitrarily as $1, 2, \ldots, m$.
2. Initialize $S_i = \emptyset$ for every i.
3. For $j = 1, 2, \ldots, m$:
   (a) Add item j to the bundle i that maximizes the marginal value $v_i(S_i \cup \{j\}) - v_i(S_i)$ (breaking ties arbitrarily).

Prove that this algorithm computes an allocation that has welfare at least $\frac{1}{2}$ times the maximum possible.

[Hint: let Δ_j denote the increase in welfare of the greedy algorithm's allocation when it assigns item j. Use the greedy criterion and submodularity to argue that, for each bidder i, $v_i(S_i^* \cup S_i) - v_i(S_i) \le \sum_{j \in S_i^* \setminus S_i} \Delta_j$. Finish by summing over all bidders and rearranging.]

Lecture 8 Exercises

Exercise 26 (Optional – Do Not Hand In)

Recall scenario #7, where there are m identical items and each bidder i has a (not necessarily downward sloping) valuation given by nonnegative marginal values $\mu_i(1), \ldots, \mu_i(m)$. Prove that a welfare-maximizing allocation can be computed in time polynomial in n and m.

[Hint: dynamic programming.]

Exercise 27

Let $\mathbf{x}$ be a maximal-in-distributional-range (MIDR) allocation rule. Recall this means that there is a set $\mathcal{D}$ of distributions over Ω — the *distributional range* — such that $\mathbf{x}$ has the following form:

1. Given (reported) valuations $\mathbf{b}$ on Ω, compute the distribution D^* that maximizes the expected welfare $\mathbf{E}_{\omega \in D}[\sum_{i=1}^{n} b_i(\omega)]$ over $D \in \mathcal{D}$.[1]

2. Return an outcome ω drawn at random from the distribution D^*.

The key points are: (i) $\mathcal{D}$ is defined up front, independent of the reported valuations $\mathbf{b}$; (ii) given $\mathcal{D}$, the allocation rule is uniquely defined (up to tie-breaking) — it samples an outcome from the distribution that maximizes expected welfare with respect to the reported valuations.

Now define a payment rule by

$$p_i(\mathbf{b}) = \max_{D \in \mathcal{D}} \mathbf{E}_{\omega \in D}\left[\sum_{k \neq i} b_i(\omega)\right] - \mathbf{E}_{\omega \in D^*}\left[\sum_{k \neq i} b_i(\omega)\right];$$

that is, bidder i pays the loss of expected welfare to others caused by its participation in the mechanism.

Prove that $(\mathbf{x}, \mathbf{p})$ is a DSIC (randomized) mechanism, meaning that for every bidder i, valuation v_i, and reported valuations $\mathbf{b}_{-i}$ by the others, i maximizes its expected quasi-linear utility by revealing its true valuation:

$$v_i \in \operatorname*{argmax}_{b_i} \{\mathbf{E}_{\omega \sim \mathbf{x}(b_i, \mathbf{b}_{-i})}[v_i(\omega) - p_i(b_i, \mathbf{b}_{-i})]\}.$$

[1] To ensure that this maximum exists, it is a good idea to take $\mathcal{D}$ to be a compact set.