CS 886: Multiagent Sysytems
Assignment 2

November 13, 2008

Due Monday, November 10 in class. In this assignment you are expected to work individually. You may use any sources that you want, but you must cite them. You can email me or drop by my office if you have questions.

1. (30 points) (Thanks to K Leyton-Brown) Suppose you have some object that each of n agents desires, but which you do not value. Assume that each agent i values it at v_i with v_i's drawn independently and uniformly from some real line positive interval, say $(0,10^{100}]$. Although you do not desire the object and do not care about the actual values of the v_i's, you need to compute $\sqrt{v_i}$ for each i.

   Unfortunately, you face two problems. First, agents are not inclined to just reveal to you anything about their v_i's. Second, your computer is costly to operate. It costs you 1 unit to determine the greater of two values, 2 units to perform any basic arithmetic operation $(+, -, \times,)$, and anything more complicated (say $\sqrt{x}$) costs 20. The (accurate) current time of day can be observed without cost.

   (a) How much does it cost to compute $\sqrt{v_i}$ for each i using a straightforward VCG mechanism? When computing cost, ignore the revenue that the auction will generate.

   There are a couple of ways to do this, but the cheapest is to make the observation that we can run a Vickrey auction, which means that all non-winning agents pay zero. This means that we do not need to compute payments for the $n-1$ lowest bidders.

   To run the mechanism we need to find the highest and second highest bids. To do this in a naive way we need to do $n-1$ comparisons to find the highest bid. Once we have the higest bid, we need to do $n-2$ comparisons to find the second highest bid. (There are more efficient ways to doing this). We also need to compute the square root of each valuation, which costs us $20n$. Therefore, the total cost is

$$20n + (n-1) + (n-2) = 22n - 3.$$

(b) Your answer above gives an upper bound on the cost of computing the square roots of the valuations. Design an incentive-compatible, dominant strategy ("strategy-proof") direct mechanism that will allow you to compute all $\sqrt{v_i}$ at minimal cost. Assume that the agents can do computations for free. Make sure that you specify all the components of the mechanism: players, actions, outcomes, mappings from actions to outcomes. Explain why your mechanism is strategy-proof. Specify the algorithms that you will use to implement those mappings. Give your mechanism's total computation cost (or an upper bound on it). You do not need to prove that your mechanism has minimal cost. (Hint: Think about the revelation principle.)

There are different ways to solve this problem, but one way is to note that the main cost is the square-root calculations. If you could make the agents report the square-root of their true valuations, then you would save $20n$.

One approach is to run an auction where the winner is the agent with the highest bid, but it must pay an amount equal to the square of the second highest bid. The players, actions, and outcomes would be the same as for the standard Vickrey auction except for this small change in the payment function. The proof that agents will reveal the true square root of their valuation is also similar to the proof of the strategy-proofness of the Vickrey auction.

If an agent i had not won the auction by revealing $\sqrt{v_i}$, then it still would have lost by announcing $b_i < \sqrt{v_i}$. If it had announced $b_i > \sqrt{v_i}$ and won, then it would have paid b^2 where $b_i > b > \sqrt{v_i}$. Thus, agent i would have had negative utility from bidding b_i. Similarly, if the agent would have won by announcing $\sqrt{v_i}$ then it can not improve its utility by bidding higher since this would not change the price that it paid. By bidding lower, it can also not change the price it pays, unless it bids below the second highest bid, at which point it loses the auction.

The total cost of running this auction is the search costs of finding the highest and second highest bids ($n-1$ and $n-2$) plus the cost of squaring the second highest bid (2). Therefore, the total cost is

$$n - 1 + n - 2 + 2 = 2n - 1.$$

(c) In the previous part you were restricted to direct mechanisms. Show that an *indirect* (multistage) mechanism can achieve even lower cost.

You can use the clock to run an *ascending auction*. Initialise the clock to zero. Tell all agents that they are assumed to be active bidders in the auction (that is, willing to pay the current price which is the current time on the clock) until they explicitly indicate that they withdraw from the auction. The winner is the last agent left in the auction, and will pay the square of the price (time) at which the second last bidder left. Then start the clock and wait. If agents are allowed to see each other exit then you may not get the value of the square root of the highest bidder. However, if you do not allow the agents to observe when others leave the auction you can just keep on running the clock until the last bidder exits (and still have them pay the price at which the second highest bidder left). Each agent is best off by staying in the auction until the clock goes past the square root of their valuation. The only cost that the auctioneer experiences is when it computes the square of the second highest price. That is, the cost is 2.

2. (30 points) Recall that the OR* bidding language adds *dummy goods* to bids in a combinatorial auction in order to express complex bidding languages within the OR language. Consider a combinatorial auction with n items. For each valuation function below, explain how to encode it in the OR* language using as few bids as possible. Explain how many dummy items your encoding requires.

   (a) $v(S) = k|S|$ for some constant k

   $$\text{Bid} = (\{x_1\}, k) OR (\{x_2\}, k) OR \ldots OR (\{x_n\}, k)$$

   No dummy items are required, and there are n atomic bids.

   (b) $v(S) = k|S|$ if $|S| \leq j$ for some j, otherwise $v(S) = 0$
   For each item x_i, create bids $(\{x_i, d_l\}, k)$ where $1 \leq l \leq j$. Then the bid to express the valuation is

   $$\text{OR}_{i=1}^{n} \text{OR}_{l=1}^{j} (\{x_i, d_l\}, k)$$

   There are j dummy items and $n \cdot j$ atomic bids.

   (c) $v(S) = 1$ is $|S| \geq n/2$ and $v(S) = 0$ otherwise
   We can look at the different cases in order to determine the atomic bids and when dummy items are required.
   If $|S| = \frac{n}{2}$, then the atomic bid has the form $(S \cup \{d\}, 1)$ where d is the only dummy item used. In addition to these bids, you also add in atomic bids $(\{x_i\}, 0)$ to allow for the formation of sets which are larger than $\frac{n}{2}$. That is, you use 1 dummy and $\binom{n}{\frac{n}{2}} + n$ atomic bids.
   You could also do it as follows (not as compact, but I also accepted it for fully marks)
   $|S| < \frac{n}{2}$: You would not place any atomic bids of this size.
   $|S| > \frac{n}{2}$: All S such that $|S| > \frac{n}{2}$ must have overlaping bids and so are mutually exclusive. Therefore, each atomic bid would take the form $(S, 1)$, and these would be ORed together.

$|S| = \frac{n}{2}$**:** Here you must add a single common dummy bid to all S such that $|S| = \frac{n}{2}$ since otherwise it is possible that two of these atomic bids are accepted when you really want to accept a single one.

That is, the bid looks like

$$(S_1, 1)OR \ldots, OR(S_m, 1)OR(T_1 \cup \{d\}, 1)OR \ldots OR(T_k \cup \{d\}, 1)$$

where $|S_i| \geq \frac{n}{2}$ and $|T_j| = \frac{n}{2}$, and d is the dummy item. Therefore, you would use a single dummy item, and submit

$$\sum_{i=\frac{n}{2}}^{n} \binom{n}{i}$$

atomic bids.

(d) Goods come in three colours, red, blue and green; there are $n/3$ goods of each colour. $v(S) = k_R|S|$ if it contains only red items, $v(S) = k_G|S|$ if it has only green items, and $v(S) = k_B|S|$ if it has only blue items (for some constants k_R, k_G, k_B). $v(S) = 0$ otherwise.

Let r_i, b_i, and g_i denote the ith red, blue and green item respectively. All red atomic bids take the form

$$(\{r_i, d_{r_i,b_1}, d_{r_i,b_2}, \ldots, d_{r_i,b_{n/3}}, d_{r_i,g_1}, \ldots, d_{r_i,g_{n/3}}\}, k_R)$$

All blue atomic bids take the form

$$(\{b_i, d_{b_i,r_1}, d_{b_i,r_2}, \ldots, d_{b_i,r_{n/3}}, d_{b_i,g_1}, \ldots, d_{b_i,g_{n/3}}\}, k_B)$$

All green atomic bids take the form

$$(\{g_i, d_{g_i,r_1}, d_{g_i,r_2}, \ldots, d_{g_i,r_{n/3}}, d_{g_i,b_1}, \ldots, d_{g_i,b_{n/3}}\}, k_G)$$

The number of *unique* dummy items is

$$(\frac{2n}{3})\frac{n}{3} + (\frac{n}{3})\frac{n}{3} = \frac{n^2}{3}.$$

The number of atomic bids is n.

(e) Goods come in pairs. $v(S) = |S|$ if it contains at most one item from every pair, and $v(S) = 0$ otherwise.

Let x_i, x'_i denote the ith pair of items. Let d_i denote the dummy item associated with pair i. The valuation is represented as

$$(\{x_1, d_1\}, 1)OR(\{x'_1, d_1\}, 1)OR \ldots OR(\{x_{n/2}, d_{n/2}\}, 1)OR(\{x'_{n/2}, d_{n/2}\})$$

A total of $\frac{n}{2}$ dummy items are used, and n atomic bids.