Algorithmic Game Theory, Summer 2015	Lecture 10 (5 pages)

VCG Mechanisms

Instructor: Thomas Kesselheim

In previous lectures we introduced the mechanism design problem, and used the revelation principle to show that every social choice function that can be implemented in dominant strategies also has an incentive compatible implementation. Next we proved the Gobbard-Sutterthwarte theorem which tells us that if we do not have any restrictions on the environment, only dictatorial choice functions can be implemented in dominant strategies. Hence people turn to look at special classes of utility functions.

A particular class that we considered is the quasilinear environment, which intuitively can also be viewed as mechanisms with money. In this setting the mechanism needs to decide on an outcome as well as the payment that each agent needs to pay. And the goal is to implement an efficient social choice function that gives an outcome to maximize the social welfare. In previous lectures we looked at a very simple example: the single-item auction, and showed that there does exist an incentive compatible mechanism for this problem: the second-price auction. It turns out that this is not only a coincidence. In this lecture we will talk about a cornerstone result of mechanism design theory, saying that for every efficient social choice function in the quasilinear environment, there is always an incentive compatible implementation.

1 Model

Recall that a quasilinear environment is defined as following:

- the set of outcomes has form $O = S \times \mathbb{R}^n$, where S is the set of possible solutions, and $\mathbb{R}^n$ represents the payments that agents have to pay (or receive) for a given solution,
- all possible utility functions are of form $u_i(s,(p_1,\ldots,p_n)) = v_i(s) - p_i$ for any $(s,(p_1,\ldots,p_n)) \in O$, where $v_i(s)$ is the value of solution s to agent i, and p_i is the amount of money that agent i needs to pay (or receive if p_i is negative).
- the social choice function $C(\mathbf{u}) = (s,(p_1,\ldots,p_n))$ maps utility functions $\mathbf{u}$ to the solution s that maximize $\sum_i v_i(s)$. Such social choice functions are called *efficient*.

We simplify notation a bit. As we consider direct mechanisms, each agent effectively reports a valuation function. To make the distinction clear, we denote this reported valuation by b_i, whereas the true (only privately known) valuation is denoted by v_i. Both belong to V_i, the set of possible valuation functions for agent i. The cartesian product of all V_i, we denote by $\mathbf{V}$; the cartesian product of all V_j except V_i by $\mathbf{V}_{-i}$.

Slightly abusing notation, we will denote a mechanism in quasilinear environments as $(f,p_1,\ldots,p_n)$, where $f\colon \mathbf{V} \to S$ is a social choice function and $p_i\colon \mathbf{V} \to \mathbb{R}$ indicates the amount of money that player i needs to pay.

2 Groves Mechanisms

Next we show the main positive result in this setting.

Definition 10.1 (Groves Mechanism)**.** *A* **Groves mechanism** *is a mechanism* $(f,p_1,\ldots,p_n)$ *in a quasilinear environment in which:*

- $f(\mathbf{b}) \in \arg\max_{s \in S} \sum_i b_i(s)$, *and*
- *for every* i, $p_i(\mathbf{b}) = h_i(\mathbf{b}_{-i}) - \sum_{j \neq i} b_j(f(\mathbf{b}))$, *where* $h_i \colon \mathbf{V}_{-i} \to \mathbb{R}$ *is an arbitrary function that does not depend on* b_i *(let alone* v_i*).*

Theorem 10.2 (Vickrey-Clarke-Groves)**.** *Every Groves Mechanism is an incentive compatible mechanism.*

Proof. Observe that for all b_i, $\mathbf{b}_{-i}$,

$$u_i(b_i, \mathbf{b}_{-i}) = v_i(f(b_i, \mathbf{b}_{-i})) - p_i(b_i, \mathbf{b}_{-i}) = v_i(f(b_i, \mathbf{b}_{-i})) - h_i(\mathbf{b}_{-i}) + \sum_{j \neq i} b_j(f(b_i, \mathbf{b}_{-i})) \ .$$

On input $(v_i, \mathbf{b}_{-i})$, the function f returns a solution s^*, which maximizes $v_i(s^*) + \sum_{j \neq i} b_j(s^*)$. That is, for any $s \in S$, we have $v_i(s^*) + \sum_{j \neq i} b_j(s^*) \geq v_i(s) + \sum_{j \neq i} b_j(s)$. In particular, this holds for $s = f(b_i, \mathbf{b}_{-i})$ for all possible b_i.

Consequently,

$$v_i(f(v_i, \mathbf{b}_{-i})) + \sum_{j \neq i} b_j(f(v_i, \mathbf{b}_{-i})) \geq v_i(f(b_i, \mathbf{b}_{-i})) + \sum_{j \neq i} b_j(f(b_i, \mathbf{b}_{-i}))$$

and therefore

$$u_i(v_i, \mathbf{b}_{-i}) \geq u_i(b_i, \mathbf{b}_{-i}) \ .$$

□

3 Clarke Pivot Rule

Next we look at how to find appropriate functions $h_1, \ldots, h_n$. Although every function can guarantee incentive compatibility of the mechanism, not all of them would make sense in other aspects. For example, the most simple case is to assume that $h_i(\mathbf{u}_{-i}) \equiv 0$ for all i. This would result every agent receiving a large amount to money from the mechanism. But in many quasilinear environment scenarios such as auctions, we would want agents to pay money to the mechanism. Setting $h_i(\mathbf{u}_{-i})$ to be an arbitrarily large value is also not good. In this case, agents would probably receive negative net utilities, and in that case they would rather choose to not participate in the mechanism. Fortunately, there is a sweet spot in between these two extremes, given by the Clarke Pivot Rule.

Definition 10.3 (Clarke Pivot Rule)**.** *A Groves mechanism is said to have* Clarke Pivot Rule *if each* h_i *is of form*

$$h_i(\mathbf{b}_{-i}) = \max_{s \in S} \sum_{j \neq i} b_j(s) \ .$$

A Groves mechanism with Clarke pivot payments is also called a Vickrey-Clarke-Groves(VCG) mechanism*.*

Besides incentive compatibility, a VCG mechanism also enjoys the following nice properties:

- **Individual Rationality.** If $v_i(s) \geq 0$ for all s, then $u_i(v_i, b_{-i}) \geq 0$ for all $\mathbf{b}_{-i}$. The reason is that

$$\begin{aligned} u_i(v_i, b_{-i}) &= v_i(f(v_i, b_{-i})) + \sum_{j \neq i} b_j(f(v_i, \mathbf{b}_{-i})) - \max_{s \in S} \sum_{j \neq i} b_j(s) \\ &= \left(\max_{s \in S} \left(v_i(s) + \sum_{j \neq i} b_j(s) \right) \right) - \left(\max_{s \in S} \sum_{j \neq i} b_j(s) \right) \geq 0 \ . \end{aligned}$$

  The term is non-negative because $v_i(s) + \sum_{j \neq i} b_j(s) \geq \sum_{j \neq i} b_j(s)$ for all s. Therefore this also holds for the maximum.

- **No Positive Transfer.** For all $\mathbf{b}$, we have

$$p_i(\mathbf{b}) = \left(\max_{s \in S} \sum_{j \neq i} b_j(s) \right) - \left(\sum_{j \neq i} b_j(f(\mathbf{b})) \right) \geq 0 \ ,$$

because $\sum_{j \neq i} b_j(f(b)) \leq \max_{s \in S} \sum_{j \neq i} b_j(s)$: The left-hand side is just one possible value that this expression can take whereas it is maximized on the right-hand side.

4 Examples

4.1 Single-Item Auctions Revisited

As a first example for VCG, let us consider single-item auctions again. Remember that each agent's valuation function v_i given by

$$v_i(s) = \begin{cases} t_i & \text{if agent } i \text{ receives the item in } s \\ 0 & \text{otherwise} \ . \end{cases}$$

Given the vector $\mathbf{b}$, the function f selects the agent with the highest bid. Let this agent be denoted by i^*. For i^*, we now have

$$p_{i^*}(\mathbf{b}) = h_{i^*}(\mathbf{b}_{-i^*}) - \sum_{j \neq i^*} b_j(f(\mathbf{b})) \ .$$

For $j \neq i^*$, we have $b_j(f(\mathbf{b})) = 0$ because agent j does not get the item. Furthermore

$$h_{i^*}(\mathbf{b}_{-i^*}) = \max_{s \in S} \sum_{j \neq i^*} b_j(s) \ .$$

For all agents $i \neq i^*$

$$h_i(\mathbf{b}_{-i}) = \sum_{j \neq i} b_j(f(\mathbf{b})) = b_{i^*} \ .$$

That is, agent i^* pays the second highest bid, the other agents pay nothing. This is exactly the second-price auction.

4.2 Sponsored Search Auctions

In a sponsored search auction, we sell $k < n$ ad slots on a search results page. The higher the slot is displayed on the page, the more likely it will be clicked. For slots 1, ..., k, we assume click through rates of $\alpha_1 \geq \alpha_2 \geq \ldots \geq \alpha_m$. Agent i's valuation is expressed in terms of a single number v_i such that $v_i(s) = v_i \alpha_j$ if agent i gets slot j in s.

If $v_1 \geq v_2 \geq \ldots \geq v_n$, then the social-welfare optimize allocation gives slot j to bidder j. This results in social welfare $\sum_{j=1}^{k} v_j \alpha_j$. The optimal social welfare without agent i is $\sum_{j=1}^{i-1} v_j \alpha_j + \sum_{j=i+1}^{k+1} v_j \alpha_{j-1}$. Consequently, given truthful reports, agent i's VCG payment is

$$p_i(\mathbf{v}) = \sum_{j=1}^{i-1} v_j \alpha_j + \sum_{j=i+1}^{k+1} v_j \alpha_{j-1} - \left(\sum_{j=1}^{i-1} v_j \alpha_j + \sum_{j=i+1}^{k} v_j \alpha_j \right) = \sum_{j=i+1}^{k+1} v_j (\alpha_{j-1} - \alpha_j) \ .$$

Interestingly, for mysterious reasons in practice this scheme is not applied. Instead a rule called *generalized second price* is used: Agent i has to pay $v_{i+1}\alpha_{i+1}$. This is generally not incentive compatible.

4.3 Procurement Auction

In a procurement auction, one buyers wants to buy an item that is offered by multiple sellers. Each seller incurs a private cost c_i for furnishing the item. Indeed, this setting can be modeled as an ordinary single-item auction by assuming negative values for winning the item, i.e., $t_i = -c_i$. The VCG mechanism therefore buys the item from the seller reporting the lowest cost. However, with Clarke pivot rule, we end up with a weird payment scheme. Observe that the function h_i is always 0. Therefore, losers have to pay the highest cost. The winner does not have to pay anything. This is not individually rational.

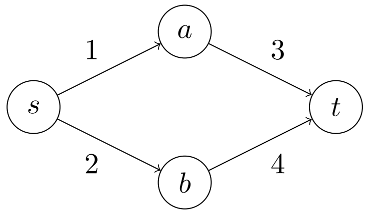
Choosing different functions h_i, namely restricting the maximum to only go over solutions in which agent i does not supply the item, we get a more reasonable payment rule. In this case, the agent with the lowest reported cost is compensated the second lowest cost; the payments for the other agents are 0. This way, we are again individually rational but have positive transfers. We would have incentive compatibility for any choice of the h_i functions.

4.4 Buying a Path in a Network

Consider a communication network, which is modeled by a directed graph G. There are two specified vertices s and t, which we would like to connect by a path. Each agent owns one edge e of the graph an incurs (private) cost c_e if the path is routed along his edge. Consequently, the mechanism has to compensate the agent for the cost by a payment. Using Clarke pivot rule, we again end up with unnatural payments because it penalizes losers. However, it is possible to generalize the functions from procurement auctions in precisely the same way.

Let $\mathcal{P}$ denote the set of all paths from s to t. We need to find the socially optimal solution p, which minimizes $\sum_{e \in p} c_e$. This can be done easily by computed the shortest path from s to t.

Consider the following example network. Numbers on the edges indicate the c_e values.



If all agents report truthfully, the path via a is selected. The agent in charge of (s, a) receives compensation $6 - 3 = 3$, the agent in charge of (a, t) receives $6 - 1 = 5$.

5 Limitations

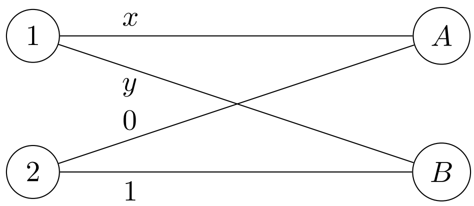
We have seen that VCG mechanisms work well in many environments. However, it does this now solve all questions regarding mechanism design with money. There are several limitations: First of all, to build a VCG mechanism, we have to solve the welfare-maximization problem optimally. In many cases, this problem is actually intractable. Below we will see that only approximating social welfare is not enough. VCG also does not optimize the payments in any sense. For example, it does not even try to maximize the revenue obtained by the payments. Also, agents only have a limited budget, but we do not ensure that they only spend a certain amount. Finally, it might be a problem that agents collude. Although each single agents cannot benefit from false reports themselves, other agents can.

Probably the biggest limitation from an algorithmic aspect is the fact that VCG requires a welfare-maximizing solution. It will be instructive to see that this is indeed necessary because there are approximation algorithms that cannot be turned into an incentive compatible mechanism.

Theorem 10.4. *There are functions f such that there is no incentive compatible mechanism $(f, p_1, \ldots, p_n)$, even for $\sum_i b_i(f(\mathbf{b})) \geq \frac{1}{2} \max_{s \in S} \sum_i b_i(s)$ for all $\mathbf{b}$.*

Proof. We consider the problem of selling m items to n agents. Each agent i has a separate valuation $v_{i,j}$ for each of the items j. To run VCG, we would need to find the maximum-weight matching of the bipartite allocation graph. This is possible in polynomial time. However, a fast way to find a reasonable matching is the greedy algorithm: Always take the maximum-weight edge whose both endpoints are still unmatched. It is easy to see that this algorithm is a 2-approximation. That is, we have $\sum_i b_i(f(\mathbf{b})) \geq \frac{1}{2} \max_{s \in S} \sum_i b_i(s)$ for all $\mathbf{b}$.

We consider this kind of instance to show that no payment scheme can render the mechanism incentive compatible.



There are two items A and B. Bidder 1 has values x and y; bidder 2 has values 0 and 1. From different values of x and y, we will conclude properties of the payments that an incentive compatible mechanism would need to fulfill. We keep bidder 2's valuation and report fixed at all times.

Step 1: In every report that bidder 1 can make that gets him item A, bidder 1 pays the same amount. Suppose there is a pair of reports b_1, b'_1 with different payments in which bidder 1 gets item A. Without loss of generality $p_1(b_1, v_2) < p_1(b'_1, v_2)$. If player 1's true valuation is b'_1 then he would be better off by reporting b_1 instead. The same argument also holds for item B; call the respective prices p_A and p_B.

Step 2: We now claim that $p_A = p_B$. Consider an arbitrarily small $\epsilon > 0$. If $x = 1 + 2\epsilon$, $y = 1 + \epsilon$, then bidder 1 could misreport values 0 and $1 + \epsilon$. We then have $u_1(v_1, v_2) = 1 + 2\epsilon - p_A$, $u_1(b'_1, v_2) = 1 + \epsilon - p_B$. As $u_1(v_1, v_2) \geq u_1(b'_1, v_2)$, we have $p_B \geq p_A - \epsilon$. Therefore $p_B \geq p_A$. We can show $p_B \leq p_A$, by considering $x = 1 + \epsilon$, $y = 1 + 2\epsilon$.

Step 3: Consider $x = \frac{1}{4}$, $y = \frac{1}{2}$. Truthful reporting gives bidder 1 a utility of $u_1(v_1, v_2) = \frac{1}{4} - p_A$. Claiming instead values 0 and 2 would give utility $\frac{1}{2} - p_B = \frac{1}{2} - p_A$. □

Recommended Literature

- Chapter 9.3 in the AGT book
- Tim Roughgarden's lecture notes `http://theory.stanford.edu/~tim/f13/l/l7.pdf` and lecture video `https://youtu.be/TLl3FVXPVIY`
- Eva Tardos's lecture notes `http://www.cs.cornell.edu/courses/cs6840/2012sp/lec18.pdf`