Advanced Mechanism Design

Lecture 17

Lecture Overview

VCG

Definition (VCG mechanism)

$$\chi(\hat{v}) = \arg\max_x \sum_i \hat{v}_i(x)$$

$$p_i(\hat{v}) = \sum_{j \neq i} \hat{v}_j\left(\chi(\hat{v}_{-i})\right) - \sum_{j \neq i} \hat{v}_j(\chi(\hat{v}))$$

- You get paid everyone's utility under the allocation that is actually chosen
  - except your own, but you get that directly as utility
- Then you get charged everyone's utility in the world where you don't participate
- Thus you pay your social cost

VCG and Individual Rationality

Definition (Choice-set monotonicity)

An environment exhibits choice-set monotonicity if $\forall i$, $X_{-i} \subseteq X$.

Definition (No negative externalities)

An environment exhibits no negative externalities if $\forall i \forall x \in X_{-i},\ v_i(x) \geq 0$.

Theorem

The VCG mechanism is ex-post individual rational when the choice set monotonicity and no negative externalities properties hold.

VCG and Budget Balance

Definition (No single-agent effect)

An environment exhibits no single-agent effect if $\forall i$, $\forall v_{-i}$, $\forall x \in \arg\max_y \sum_j v_j(y)$ there exists a choice x' that is feasible without i and that has $\sum_{j \neq i} v_j(x') \geq \sum_{j \neq i} v_j(x)$.

Theorem

The VCG mechanism is weakly budget-balanced when the no single-agent effect property holds.

More good news

Theorem (Krishna & Perry, 1998)

In any Bayesian game setting in which VCG is ex post individually rational, VCG collects at least as much revenue as any other efficient and ex interim individually-rational mechanism.

- This is somewhat surprising: does not require dominant strategies, and hence compares VCG to all Bayes–Nash mechanisms.
- A useful corollary: VCG is as budget balanced as any efficient mechanism can be
  - it satisfies weak budget balance in every case where *any* dominant strategy, efficient and *ex interim* IR mechanism would be able to do so.

Bad news

Theorem (Green–Laffont; Hurwicz)

No dominant-strategy incentive-compatible mechanism is always both efficient and weakly budget balanced, even if agents are restricted to the simple exchange setting.

Theorem (Myerson–Satterthwaite)

No Bayes-Nash incentive-compatible mechanism is always simultaneously efficient, weakly budget balanced and ex-interim individual rational, even if agents are restricted to quasilinear utility functions.

Lecture Overview

1. Privacy

- VCG requires agents to fully reveal their private information
- this private information may have value to agents that extends beyond the current interaction
  - for example, the agents may know that they will compete with each other again in the future
- it is often preferable to elicit only as much information from agents as is required to determine the social welfare maximizing choice and compute the VCG payments.

2. Susceptibility to Collusion

Example

Agent	U(build road)	U(do not build road)	Payment
1	200	0	150
2	100	0	50
3	0	250	0

- What happens if agents 1 and 2 both increase their declared valuations by $50?

2. Susceptibility to Collusion

Example

Agent	U(build road)	U(do not build road)	Payment
1	250	0	
2	150	0	
3	0	250	0

- What happens if agents 1 and 2 both increase their declared valuations by $50?

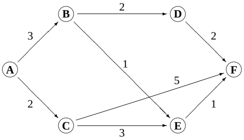
2. Susceptibility to Collusion

Example

Agent	U(build road)	U(do not build road)	Payment
1	250	0	100
2	150	0	0
3	0	250	0

- What happens if agents 1 and 2 both increase their declared valuations by $50?
- The choice is unchanged, but both of their payments are reduced.
- Thus, while no agent can gain by changing his declaration, groups *can*.

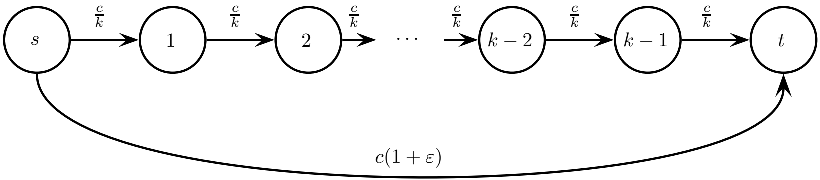
3. VCG is not Frugal



- VCG can end up paying arbitrarily more than an agent is willing to accept (or equivalently charging arbitrarily less than an agent is willing to pay)
- Consider AC, which is not part of the shortest path.
  - If the cost of this edge increased to 8, our payment to AB would increase to $p_{AB} = (-12) - (-2) = -10$.
  - If the cost were any $x \geq 2$, we would select the path $ABEF$ and would have to make a payment to AB of $p_{AB} = (-4 - x) - (-2) = -(x + 2)$.
  - The gap between agents' true costs and the payments that they could receive under VCG is unbounded.

3. VCG is not Frugal

Are VCG's payments at least close to the cost of the *second* shortest disjoint path?



- The top path has a total cost of c.
- VCG picks it, pays each of the k agents $c(1+\varepsilon) - (k-1)\frac{c}{k}$.
- Hence VCG's total payment is $c(1+k\varepsilon)$.
- For fixed ε, VCG's payment is $\Theta(k)$ times the cost of the second shortest disjoint path.

4. Revenue Monotonicity Violated

Revenue monotonicity: revenue always weakly increases as agents are added.

Example

Agent	U(build road)	U(do not build road)	Payment
1	0	90	0
2	100	0	90

4. Revenue Monotonicity Violated

Revenue monotonicity: revenue always weakly increases as agents are added.

Example

Agent	U(build road)	U(do not build road)	Payment
1	0	90	0
2	100	0	0
3	100	0	0

4. Revenue Monotonicity Violated

Revenue monotonicity: revenue always weakly increases as agents are added.

Example

Agent	U(build road)	U(do not build road)	Payment
1	0	90	0
2	100	0	0
3	100	0	0

- Adding agent 3 causes VCG to pick the same choice but to collect zero revenue!
- Agent 2 could pretend to be two agents and eliminate his payment.

5. Cannot Return All Revenue to Agents

- we may want to use VCG to induce agents to report their valuations honestly, but may not want to make a profit by collecting money from the agents.
- Thus, we might want to find some way of returning the mechanism's profits back the agents.
- However, the possibility of receiving a rebate after the mechanism has been run changes the agents' incentives.
- In fact, even if profits are given to a charity that the agents care about, or spent in a way that benefits the local economy and hence benefits the agents, the VCG mechanism is undermined.
- It *is* possible to return at least *some* of the revenues to the agents, but it must be done very carefully, and in general not all the money can be returned.

Lecture Overview

AGV Mechanism

- AGV is efficient, strictly budget balanced, but gives up dominant strategies and trades *ex post* for *ex ante* individual rationality.
- Explained in detail in the book, but the highlights:
  - allocation rule is the same as Groves
  - each agent i paid the *expected* utility of the other agents
    - based on the type distribution, not others' declarations
  - each agent charged $\frac{1}{n-1}$ of the payments made to the other agents

Lecture Overview

Besides VCG, what DS mechanisms can we build?

Let $\mathcal{X}_i(\widehat{v}_{-i}) \subseteq X$ denote the set of choices that can be selected by the choice rule χ given the declaration $\widehat{v}_{-i}$ by the agents other than i (i.e., the range of $\chi(\cdot, \widehat{v}_{-i})$).

Theorem

A deterministic mechanism is dominant-strategy truthful if and only if, for every $i \in N$ and every $\widehat{v}_{-i} \in V_{-i}$:

1. *The payment function $p_i(\widehat{v})$ can be written as $p_i(\widehat{v}_{-i}, \chi(\widehat{v}))$;*
2. $\forall v_i \in V_i$, $\chi(v_i, \widehat{v}_{-i}) \in \arg\max_{x \in \mathcal{X}_i(\widehat{v}_{-i})}(v_i(x) - p_i(\widehat{v}_{-i}, x))$.

- an agent's payment can only depend on other agents' declarations and the selected choice
  - it *cannot* depend otherwise on the agent's own declaration.
- taking the other agent's declarations and the payment function into account, from every player's point of view the mechanism selects the most preferable choice.

Which *Choice Rules* can be Implemented?

Definition (Affine maximizer)

A social choice function is an affine maximizer if it has the form

$$\arg\max_{x\in X}\left(\gamma_x + \sum_{i\in N} w_i v_i(x)\right),$$

where each γ_x is an arbitrary constant (may be $-\infty$) and each $w_i \in \mathbb{R}_+$.

Theorem (Roberts)

If there are at least three choices that a social choice function will choose given some input, and if agents have general quasilinear preferences, then the set of (deterministic) social choice functions implementable in dominant strategies is precisely the set of affine maximizers.

Understanding Roberts

- In the case of general quasilinear preferences (i.e., when each agent can have any valuation for each choice $x \in X$) and where the choice function selects from more than two alternatives, affine maximizers are the only DS-implementable social choice functions.

Understanding Roberts

- In the case of general quasilinear preferences (i.e., when each agent can have any valuation for each choice $x \in X$) and where the choice function selects from more than two alternatives, affine maximizers are the only DS-implementable social choice functions.
- Efficiency is an affine-maximizing social choice function for which $\forall x \in X, \gamma_x = 0$ and $\forall i \in N, w_i = 1$.
  - Affine maximizing mechanisms are weighted Groves mechanisms
    - They transform both the choices and the agents' valuations by applying linear weights, then effectively run a Groves mechanism in the transformed space
  - Thus, we cannot stray very far from Groves mechanisms even if we give up on efficiency

Understanding Roberts

- In the case of general quasilinear preferences (i.e., when each agent can have any valuation for each choice $x \in X$) and where the choice function selects from more than two alternatives, affine maximizers are the only DS-implementable social choice functions.
- Efficiency is an affine-maximizing social choice function for which $\forall x \in X, \gamma_x = 0$ and $\forall i \in N, w_i = 1$.
  - Affine maximizing mechanisms are weighted Groves mechanisms
    - They transform both the choices and the agents' valuations by applying linear weights, then effectively run a Groves mechanism in the transformed space
  - Thus, we cannot stray very far from Groves mechanisms even if we give up on efficiency
- It is possible to implement a richer set of functions when agents' preferences are restricted further

Lecture Overview

Computational applications of mechanism design

1. Task scheduling
    - allocate tasks among agents to minimize makespan

Computational applications of mechanism design

1. Task scheduling
   - allocate tasks among agents to minimize makespan
2. Bandwidth allocation in computer networks
   - allocate the real-valued capacity of a single network link among users with different demand curves

Computational applications of mechanism design

1. Task scheduling
   - allocate tasks among agents to minimize makespan
2. Bandwidth allocation in computer networks
   - allocate the real-valued capacity of a single network link among users with different demand curves
3. Multicast cost sharing
   - share the cost of a multicast transmission among the users who receive it

Computational applications of mechanism design

1. Task scheduling
   - allocate tasks among agents to minimize makespan
2. Bandwidth allocation in computer networks
   - allocate the real-valued capacity of a single network link among users with different demand curves
3. Multicast cost sharing
   - share the cost of a multicast transmission among the users who receive it
4. Two-sided matching
   - pair up members of two groups according to their preferences, without imposing any payments
   - e.g., students and advisors; hospitals and interns; kidney donors and recipients

Constrained Mechanism Design

Imagine that the mechanism designer isn't allowed to change the agents' strategy spaces *or* utility functions.

- e.g., improve traffic flow without building new roads or closing existing ones
- e.g., broker peace between two warring countries without changing their military capabilities

Constrained Mechanism Design

Imagine that the mechanism designer isn't allowed to change the agents' strategy spaces *or* utility functions.

What can be done in such settings?

1. Contracts
2. Bribes
3. Mediators

Constrained Mechanism Design

Imagine that the mechanism designer isn't allowed to change the agents' strategy spaces *or* utility functions.

What can be done in such settings?

1. Contracts
   - agents agree to a contract before playing the game
   - if anyone deviates, he is punished
   - folk-theorem-like result: can implement any social choice function that the agents prefer to the punishment
2. Bribes
3. Mediators

Constrained Mechanism Design

Imagine that the mechanism designer isn't allowed to change the agents' strategy spaces *or* utility functions.

What can be done in such settings?

1. Contracts
2. Bribes
   - promise to pay agents if a given joint outcome is reached
   - these promises can change the game's payoffs so that agents have dominant strategies
   - any Nash equilibrium can be transformed into a dominant strategy equilibrium in this way, and the mechanism designer does not even have to pay the agents anything in equilibrium!
3. Mediators

Constrained Mechanism Design

Imagine that the mechanism designer isn't allowed to change the agents' strategy spaces *or* utility functions.

What can be done in such settings?

1. Contracts
2. Bribes
3. Mediators
   - Offer to play on behalf of agents
   - e.g., in Prisoner's Dilemma, play C if the other agent also uses the mediator; otherwise play D
   - using the mediator's services is weakly dominant, and leads to both agents cooperating in equilibrium
   - in a broad class of other games, mediators can be used to implement the optimal-surplus outcome