Competition and Cooperation in Multi-Agent Systems

Lecture 2 - Design of Mechanisms for Agent Cooperation

Vincenzo Bonifaci[1]

Institute for Systems Analysis and Informatics (IASI) – CNR, Italy

October 31, 2012

[1] Most slides are from prof. K. Leyton-Brown's MAS class

Lecture Overview

1. Auctions
2. Canonical Single-Good Auctions
3. Comparing Auctions
4. Second-price auctions

Motivation

- Auctions are any mechanisms for allocating resources among self-interested agents
- Very widely used
  - government sale of resources
  - privatization
  - stock market
  - request for quote
  - FCC spectrum
  - real estate sales
  - eBay

CS Motivation

- resource allocation is a fundamental problem in CS
- increasing importance of studying distributed systems with heterogeneous agents
- markets for:
  - computational resources (JINI, etc.)
  - P2P systems
  - network bandwidth
- currency needn't be real money, just something scarce
  - that said, real money trading agents are also an important motivation

First-, Second-Price Auctions

First-Price Auction

- bidders write down bids on pieces of paper
- auctioneer awards the good to the bidder with the highest bid
- that bidder pays the amount of his bid

Second-Price Auction

- bidders write down bids on pieces of paper
- auctioneer awards the good to the bidder with the highest bid
- that bidder pays the amount bid by the second-highest bidder

Second-Price proof

Theorem

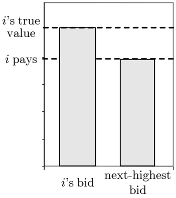
Truth-telling is a dominant strategy in a second-price auction.

Proof.

Assume that the other bidders bid in some arbitrary way. We must show that i's best response is always to bid truthfully. We'll break the proof into two cases:

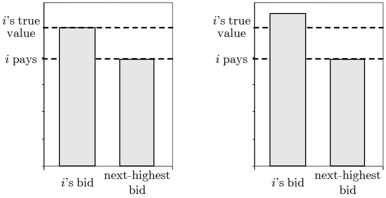
1. Bidding honestly, i would win the auction
2. Bidding honestly, i would lose the auction

Second-Price proof (2)



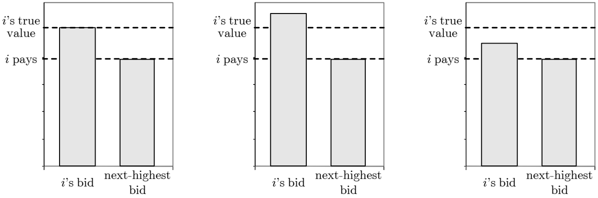
- Bidding honestly, i is the winner

Second-Price proof (2)



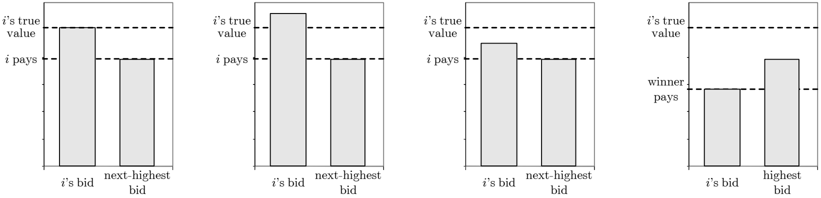
- Bidding honestly, i is the winner
- If i bids higher, he will still win and still pay the same amount

Second-Price proof (2)



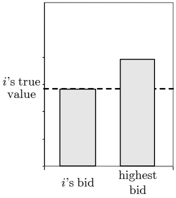
- Bidding honestly, i is the winner
- If i bids higher, he will still win and still pay the same amount
- If i bids lower, he will either still win and still pay the same amount...

Second-Price proof (2)



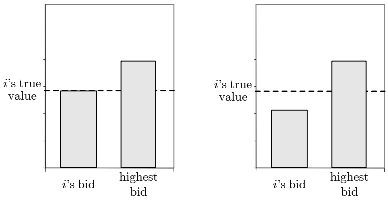
- Bidding honestly, i is the winner
- If i bids higher, he will still win and still pay the same amount
- If i bids lower, he will either still win and still pay the same amount. . . or lose and get utility of zero.

Second-Price proof (3)



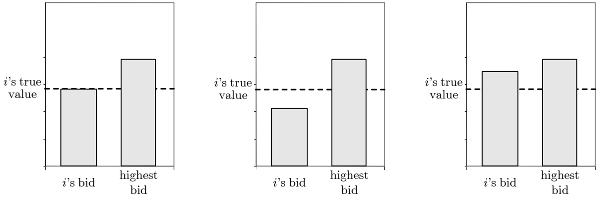
- Bidding honestly, i is not the winner

Second-Price proof (3)



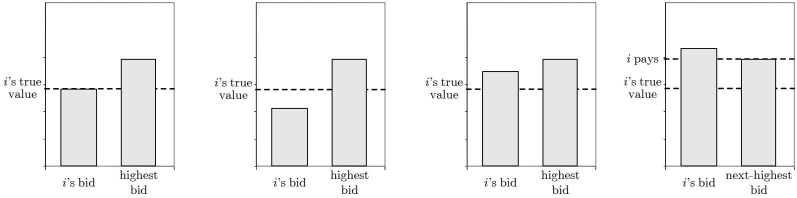
- Bidding honestly, i is not the winner
- If i bids lower, he will still lose and still pay nothing

Second-Price proof (3)



- Bidding honestly, i is not the winner
- If i bids lower, he will still lose and still pay nothing
- If i bids higher, he will either still lose and still pay nothing. . .

Second-Price proof (3)



- Bidding honestly, i is not the winner
- If i bids lower, he will still lose and still pay nothing
- If i bids higher, he will either still lose and still pay nothing. . . or win and pay more than his valuation.

Lecture Overview

Introduction

Our setting now:

- a set of outcomes
- agents have preferences across them
- for the moment, we won't consider incentive issues:
    - center knows agents' preferences, or they declare truthfully
- the goal: a social choice function: a mapping from everyone's preferences to a particular outcome, which is enforced
    - how to pick such functions with desirable properties?

Formal model

Definition (Social choice function)

Assume a set of agents $N = \{1, 2, \ldots, n\}$ and a set of outcomes (or candidates, etc.) O. Let L be the set of total orders on O. A social choice function (over N and O) is a function $C : L^n \to O$.

Definition (Social welfare function)

Let L, N, O as above. A social welfare function (over N and O) is a function $W : L^n \to L$.

A social choice function aggregates the preferences into an outcome
A social welfare function aggregates the preferences into a single preference ordering

Non-Ranking Voting Schemes

- Plurality
  - pick the outcome which is preferred by the most people
- Cumulative voting
  - distribute e.g., 5 votes each
  - possible to vote for the same outcome multiple times
- Approval voting
  - accept as many outcomes as you "like"

Ranking Voting Schemes

- Plurality with elimination ("instant runoff")
  - everyone selects their favorite outcome
  - the outcome with the fewest votes is eliminated
  - repeat until one outcome remains
- Borda
  - assign each outcome a number.
  - The most preferred outcome gets a score of $n-1$, the next most preferred gets $n-2$, down to the n^{th} outcome which gets 0.
  - Then sum the numbers for each outcome, and choose the one that has the highest score
- Pairwise elimination
  - in advance, decide a schedule for the order in which pairs will be compared.
  - given two outcomes, have everyone determine the one that they prefer
  - eliminate the outcome that was not preferred, and continue with the schedule

Lecture Overview

Condorcet Condition

- If there is a candidate who is preferred to every other candidate in pairwise runoffs, that candidate should be the winner
- While the Condorcet condition is considered an important property for a voting system to satisfy, there is not always a Condorcet winner
- sometimes, there's a cycle where A defeats B, B defeats C, and C defeats A in their pairwise runoffs

Condorcet example

$$
\begin{aligned}
\text{499 agents:} \quad & A \succ B \succ C \\
\text{3 agents:} \quad & B \succ C \succ A \\
\text{498 agents:} \quad & C \succ B \succ A
\end{aligned}
$$

- What is the Condorcet winner?

Condorcet example

$$
\begin{array}{rl}
\text{499 agents:} & A \succ B \succ C \\
\text{3 agents:} & B \succ C \succ A \\
\text{498 agents:} & C \succ B \succ A
\end{array}
$$

- What is the Condorcet winner? B

Condorcet example

499 agents:	$A \succ B \succ C$
3 agents:	$B \succ C \succ A$
498 agents:	$C \succ B \succ A$

- What is the Condorcet winner? B
- What would win under plurality voting?

Condorcet example

$$\begin{aligned} 499 \text{ agents:} &\quad A \succ B \succ C \\ 3 \text{ agents:} &\quad B \succ C \succ A \\ 498 \text{ agents:} &\quad C \succ B \succ A \end{aligned}$$

- What is the Condorcet winner? B
- What would win under plurality voting? A

Condorcet example

$$\begin{array}{rl} \text{499 agents:} & A \succ B \succ C \\ \text{3 agents:} & B \succ C \succ A \\ \text{498 agents:} & C \succ B \succ A \end{array}$$

- What is the Condorcet winner? B
- What would win under plurality voting? A
- What would win under plurality with elimination?

Condorcet example

$$\begin{aligned} 499 \text{ agents:} \quad & A \succ B \succ C \\ 3 \text{ agents:} \quad & B \succ C \succ A \\ 498 \text{ agents:} \quad & C \succ B \succ A \end{aligned}$$

- What is the Condorcet winner? B
- What would win under plurality voting? A
- What would win under plurality with elimination? C

Sensitivity to Losing Candidate

35 agents: $A \succ C \succ B$
33 agents: $B \succ A \succ C$
32 agents: $C \succ B \succ A$

- What candidate wins under plurality voting?

Sensitivity to Losing Candidate

35 agents: $A \succ C \succ B$
33 agents: $B \succ A \succ C$
32 agents: $C \succ B \succ A$

- What candidate wins under plurality voting? A

Sensitivity to Losing Candidate

35 agents: $A \succ C \succ B$
33 agents: $B \succ A \succ C$
32 agents: $C \succ B \succ A$

- What candidate wins under plurality voting? A
- What candidate wins under Borda voting?

Sensitivity to Losing Candidate

35 agents: $A \succ C \succ B$
33 agents: $B \succ A \succ C$
32 agents: $C \succ B \succ A$

- What candidate wins under plurality voting? A
- What candidate wins under Borda voting? A

Sensitivity to Losing Candidate

35 agents: $A \succ C \succ B$
33 agents: $B \succ A \succ C$
32 agents: $C \succ B \succ A$

- What candidate wins under plurality voting? A
- What candidate wins under Borda voting? A
- Now consider dropping C. Now what happens under both Borda and plurality?

Sensitivity to Losing Candidate

35 agents: $A \succ C \succ B$
33 agents: $B \succ A \succ C$
32 agents: $C \succ B \succ A$

- What candidate wins under plurality voting? A
- What candidate wins under Borda voting? A
- Now consider dropping C. Now what happens under both Borda and plurality? B wins.

Lecture Overview

Notation

- N is the set of agents
- O is a finite set of outcomes with $|O| \geq 3$
- L is the set of all possible strict preference orderings over O.
  - for ease of exposition we switch to strict orderings
  - we will end up showing that desirable SWFs cannot be found *even if* preferences are restricted to strict orderings
- $[\succ]$ is an element of the set L^n (a preference ordering for every agent; the input to our social welfare function)
- $\succ_W$ is the preference ordering selected by the social welfare function W.
  - When the input to W is ambiguous we write it in the subscript; thus, the social order selected by W given the input $[\succ']$ is denoted as $\succ_{W([\succ'])}$.

Pareto Efficiency

Definition (Pareto Efficiency (PE))

W is Pareto efficient if for any $o_1, o_2 \in O$, $\forall i\, o_1 \succ_i o_2$ implies that $o_1 \succ_W o_2$.

- when all agents agree on the ordering of two outcomes, the social welfare function must select that ordering.

Independence of Irrelevant Alternatives

Definition (Independence of Irrelevant Alternatives (IIA))

W is independent of irrelevant alternatives if, for any $o_1, o_2 \in O$ and any two preference profiles $[\succ'], [\succ''] \in L^n$, $\forall i\,(o_1 \succ'_i o_2$ if and only if $o_1 \succ''_i o_2)$ implies that $(o_1 \succ_{W([\succ'])} o_2$ if and only if $o_1 \succ_{W([\succ''])} o_2)$.

- the selected ordering between two outcomes should depend only on the relative orderings they are given by the agents.

Nondictatorship

Definition (Non-dictatorship)

W does not have a dictator if $\neg\exists i\,\forall o_1, o_2(o_1 \succ_i o_2 \Rightarrow o_1 \succ_W o_2)$.

- there does not exist a single agent whose preferences always determine the social ordering.
- We say that W is dictatorial if it fails to satisfy this property.

Arrow's Theorem

Theorem (Arrow, 1951)

Let $|O| \geq 3$. *No social welfare function* W *over* O *can at the same time be*

- *Pareto efficient;*
- *independent of irrelevant alternatives;*
- *non-dictatorial.*

So, general aggregation of preferences into a single preference ordering is impossible unless one violates some very natural properties.

An exercise: sophisticated voting

$$\text{agent 1: } A \succ B \succ C$$
$$\text{agent 2: } C \succ A \succ B$$
$$\text{agent 3: } B \succ C \succ A$$

The agents will use plurality voting, with the twist that agent 1 is the chair, and can break ties: if everyone proposes a different name, agent 1's candidate passes

Assume everybody knows everybody else's preferences

How will the agents vote?

Mechanism Design

There is still hope: we usually care for social choice functions (= outcomes), rather than social welfare functions (= rankings)

However, in social choice theory, it is assumed that agents do not try to manipulate the selection mechanism

Challenge: extend the social choice setting to a new setting where agents can't be relied upon to disclose their preferences honestly (mechanism design)

Properties of social choice functions

Definition (Dictatorship)

A social choice function $C : L^n \to O$ is a dictatorship if there is an agent i such that

$$(o \succ_i o' \ \forall o' \neq o) \Rightarrow C(\succ_1, \ldots, \succ_n) = o$$

for all $[\succ] \in L^n$.

Definition (Truthful choice function)

A social choice function $C : L^n \to O$ can be manipulated by agent i if for some $\succ_1, \ldots, \succ_n \in L$ and some $\succ_i \in L$ we have $o' \succ_i o$, where

$$o = C(\succ_1, \ldots, \succ_i, \ldots, \succ_n)$$
$$o' = C(\succ_1, \ldots, \succ_i', \ldots, \succ_n)$$

C is truthful if it cannot be manipulated by any agent.

Impossibility of general mechanism design

Theorem (Gibbard-Satterthwaite)

Let C be a social choice function surjective onto O, where $|O| \geq 3$. Then either C is a dictatorship, or C is not truthful.

The proof uses Arrow's Theorem

So, is mechanism design impossible after all?

Lecture overview

- Mechanisms with money

Games with strict incomplete information

Definition (Strict incomplete information game)

A game with strict incomplete information for n agents is given by:

- For every agent i, a set of actions A_i
- For every agent i, a set of types Θ_i. A value $\theta_i \in \Theta_i$ is the private information of i.
- For every agent i, a utility function $u_i : \Theta_i \times A_1 \times \cdots \times A_n \to \mathbb{R}$.

The payoff of i is $u_i(\theta_i, a)$ when action profile a is selected

Strategies for strict incomplete information games

Definition (Strategy for a strict inc. information game)

- A strategy of agent i is a function $s_i : \Theta_i \to A_i$
- s_i is a (weakly) dominant strategy if for every θ_i, the action $s_i(\theta_i)$ is a weakly dominant strategy in the full information game defined by θ_i:

$$u_i(\theta_i, (s_i(\theta_i), s_{-i}(\theta_{-i}))) \geq u_i(\theta_i, (a_i', s_{-i}(\theta_{-i}))) \quad \forall i \in N, \forall \theta \in \Theta, \forall a_i' \in A_i$$

That is, the action $s_i(\theta_i)$ is dominant for agent i (given his type), even without knowing the other agents' actions or types

Mechanisms with money

Definition (Quasilinear mechanism)

A quasilinear mechanism for n agents is given by

- agents' type spaces $\Theta_1, \ldots, \Theta_n$
- agents' action spaces $A_1, \ldots, A_n$
- a set of outcomes X
- valuation functions $v_i : \Theta_i \times X \to \mathbb{R}$
- a choice rule $\chi : A_1 \times \cdots \times A_n \to X$
- payment rules $p_i : A_1 \times \cdots \times A_n \to \mathbb{R}$

The utility of the agent i in the induced game is

$$u_i(\theta_i, a) := v_i(\theta_i, \chi(a)) - p_i(a)$$

Implementation of social choice functions

Definition (Implementation of social choice functions)

The mechanism implements a social choice function $C : \Theta \to X$ if for some dominant strategy equilibrium $s_1, \ldots, s_n$ in the mechanism's induced game,

$$\chi(s_1(\theta_1), \ldots, s_n(\theta_n)) = C(\theta_1, \ldots, \theta_n) \qquad \forall \theta \in \Theta$$

Implementation Comments

We can require that the desired outcome arises

- in the only equilibrium
- in every equilibrium
- in at least one equilibrium

Forms of implementation:

- Direct Implementation: agents each simultaneously send a single message to the center
- Indirect Implementation: agents may send a sequence of messages; in between, information may be (partially) revealed about the messages that were sent previously like extensive form

Direct mechanisms

Definition (Direct mechanism)

A direct revelation mechanism is such that for each agent i,

$$A_i = \{\hat{v}_i \mid \hat{v}_i \in \mathbb{R}^X\}$$

I.e., each agent just declares a numerical valuation for each outcome in X

Truthfulness

Definition (Truthfulness)

A quasilinear mechanism is truthful if it is direct and $\forall i \forall v_i$, agent i's equilibrium strategy is to adopt the strategy $\hat{v}_i = v_i$.

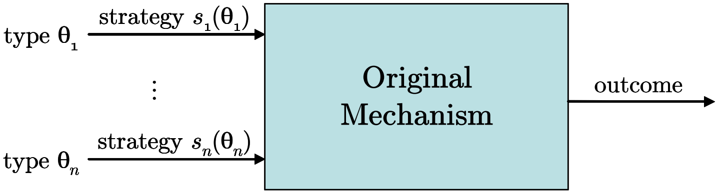
- Our definition before, adapted for the quasilinear setting

Lecture Overview

Revelation Principle

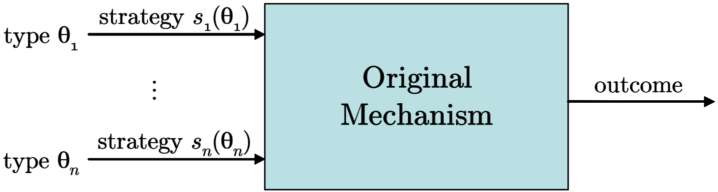
- It turns out that any social choice function that can be implemented by any mechanism can be implemented by a truthful, direct mechanism!
- Consider an arbitrary, non-truthful mechanism (e.g., may be indirect)

Revelation Principle



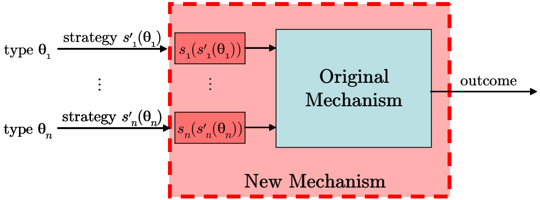
- It turns out that any social choice function that can be implemented by any mechanism can be implemented by a truthful, direct mechanism!
- Consider an arbitrary, non-truthful mechanism (e.g., may be indirect)

Revelation Principle



- It turns out that any social choice function that can be implemented by any mechanism can be implemented by a truthful, direct mechanism!
- Consider an arbitrary, non-truthful mechanism (e.g., may be indirect)
- Recall that a mechanism defines a game, and consider an equilibrium $s = (s_1, \ldots, s_n)$

Revelation Principle



- We can construct a new direct mechanism, as shown above
- This mechanism is truthful by exactly the same argument that s was an equilibrium in the original mechanism
- "The agents don't have to lie, because the mechanism already lies for them."

Computational Criticism of the Revelation Principle

- computation is pushed onto the center
  - often, agents' strategies will be computationally expensive
    - e.g., in the shortest path problem, agents may need to compute shortest paths, cutsets in the graph, etc.
  - since the center plays equilibrium strategies for the agents, the center now incurs this cost
- if computation is intractable, so that it cannot be performed by agents, then in a sense the revelation principle doesn't hold
  - agents can't play the equilibrium strategy in the original mechanism
  - however, in this case it's unclear what the agents will do

Lecture Overview

A positive result

- Recall that in the quasilinear utility setting, a mechanism can be defined as a choice rule and a payment rule.
- The Groves mechanism is a mechanism that satisfies:
  - dominant strategy (truthfulness)
  - efficiency
- In general it's not:
  - budget balanced
  - individual-rational

…though we'll see later that there's some hope for recovering these properties.

The Groves Mechanism

Definition (Groves mechanism)

The Groves mechanism is a direct quasilinear mechanism (χ, p), where

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

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

The Groves Mechanism

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

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

- The choice rule should not come as a surprise (why not?)

The Groves Mechanism

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

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

- The choice rule should not come as a surprise (why not?) because the mechanism is both truthful and efficient: these properties entail the given choice rule.

The Groves Mechanism

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

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

- The choice rule should not come as a surprise (why not?) because the mechanism is both truthful and efficient: these properties entail the given choice rule.
- So what's going on with the payment rule?
  - the agent i must pay some amount $h_i(\hat{v}_{-i})$ that doesn't depend on his own declared valuation
  - the agent i is paid $\sum_{j \neq i} \hat{v}_j(\chi(\hat{v}))$, the sum of the others' valuations for the chosen outcome

Groves Truthfulness

Theorem

Truth telling is a dominant strategy under the Groves mechanism.

Consider a situation where every agent j other than i follows some arbitrary strategy $\hat{v}_j$. Consider agent i's problem of choosing the best strategy $\hat{v}_i$. As a shorthand, we will write $\hat{v} = (\hat{v}_{-i}, \hat{v}_i)$. The best strategy for i is one that solves

$$\max_{\hat{v}_i} \left(v_i(\chi(\hat{v})) - p(\hat{v}) \right)$$

Substituting in the payment function from the Groves mechanism, we have

$$\max_{\hat{v}_i} \left(v_i(\chi(\hat{v})) - h_i(\hat{v}_{-i}) + \sum_{j \neq i} \hat{v}_j(\chi(\hat{v})) \right)$$

Since $h_i(\hat{v}_{-i})$ does not depend on $\hat{v}_i$, it is sufficient to solve

$$\max_{\hat{v}_i} \left(v_i(\chi(\hat{v})) + \sum_{j \neq i} \hat{v}_j(\chi(\hat{v})) \right).$$

Groves Truthfulness

$$\max_{\hat{v}_i}\left(v_i(\chi(\hat{v}))+\sum_{j\neq i}\hat{v}_j(\chi(\hat{v}))\right).$$

The only way the declaration $\hat{v}_i$ influences this maximization is through the choice of x. If possible, i would like to pick a declaration $\hat{v}_i$ that will lead the mechanism to pick an $x \in X$ which solves

$$\max_{x}\left(v_i(x)+\sum_{j\neq i}\hat{v}_j(x)\right). \tag{1}$$

Under the Groves mechanism,

$$\chi(\hat{v})=\arg\max_x\left(\sum_i \hat{v}_i(x)\right)=\arg\max_x\left(\hat{v}_i(x)+\sum_{j\neq i}\hat{v}_j(x)\right).$$

The Groves mechanism will choose x in a way that solves the maximization problem in Equation (1) when i declares $\hat{v}_i = v_i$. Because this argument does not depend in any way on the declarations of the other agents, truth-telling is a dominant strategy for agent i.

Proof intuition

- externalities are internalized
  - agents may be able to change the outcome to another one that they prefer, by changing their declaration
  - however, their utility doesn't just depend on the outcome—it also depends on their payment
  - since they get paid the (reported) utility of all the other agents under the chosen allocation, they now have an interest in maximizing everyone's utility rather than just their own
- in general, DS truthful mechanisms have the property that an agent's payment doesn't depend on the amount of his declaration, but only on the other agents' declarations
  - the agent's declaration is used only to choose the outcome, and to set other agents' payments

Lecture Overview

VCG

Definition (Clarke tax)

The Clarke tax sets the h_i term in a Groves mechanism as

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

Definition (Vickrey-Clarke-Groves (VCG) mechanism)

The Vickrey-Clarke-Groves mechanism is a direct quasilinear mechanism (χ, p), where

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

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

VCG discussion

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

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

- 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 discussion

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

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

Questions:

- who pays 0?

VCG discussion

$$\chi(\hat{v}) = \arg\max_{x} \sum_{i} \hat{v}_i(x)$$

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

Questions:

- who pays 0?
  - agents who don't affect the outcome

VCG discussion

$$\chi(\hat{v}) = \arg\max_{x} \sum_{i} \hat{v}_i(x)$$

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

Questions:

- who pays 0?
    - agents who don't affect the outcome
- who pays more than 0?

VCG discussion

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

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

Questions:

- who pays 0?
    - agents who don't affect the outcome
- who pays more than 0?
    - (pivotal) agents who make things worse for others by existing

VCG discussion

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

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

Questions:

- who pays 0?
  - agents who don't affect the outcome
- who pays more than 0?
  - (pivotal) agents who make things worse for others by existing
- who gets paid?

VCG discussion

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

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

Questions:

- who pays 0?
  - agents who don't affect the outcome
- who pays more than 0?
  - (pivotal) agents who make things worse for others by existing
- who gets paid?
  - (pivotal) agents who make things better for others by existing

VCG properties

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

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

- Because only pivotal agents have to pay, VCG is also called the pivot mechanism
- It's dominant-strategy truthful, because it's a Groves mechanism

Lecture Overview

Multiunit Auctions

- now let's consider a setting in which
  - there are k identical goods for sale in a single auction
  - every bidder only wants one unit
- what is VCG in this setting?

Multiunit Auctions

- now let's consider a setting in which
  - there are k identical goods for sale in a single auction
  - every bidder only wants one unit
- what is VCG in this setting?
  - every unit is sold for the amount of the $k+1$st highest bid

Multiunit Auctions

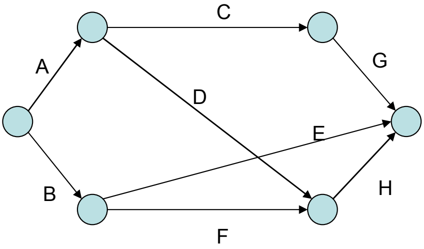
- now let's consider a setting in which
  - there are k identical goods for sale in a single auction
  - every bidder only wants one unit
- what is VCG in this setting?
  - every unit is sold for the amount of the $k+1$st highest bid
- how else can we sell the goods?

Multiunit Auctions

- now let's consider a setting in which
  - there are k identical goods for sale in a single auction
  - every bidder only wants one unit
- what is VCG in this setting?
  - every unit is sold for the amount of the $k+1$st highest bid
- how else can we sell the goods?
  - pay-your-bid: "discriminatory" pricing, because bidders will pay different amounts for the same thing
  - lowest winning bid: very similar to VCG, but ensures that bidders don't pay zero if there are fewer bids than units for sale
  - sequential single-good auctions

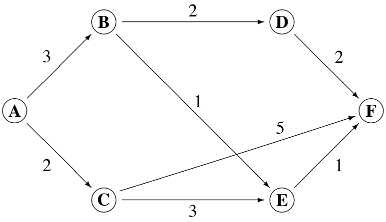
Lecture Overview

Selfish Routing



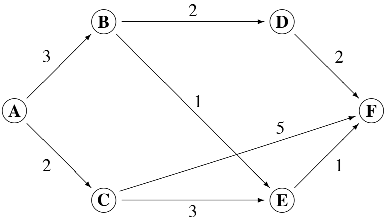
- 8 people play as agents A – H; the others act as mediators.
- Agents' utility functions: $u_i =$ payment - cost if your edge is chosen; 0 otherwise.
- Mediators: find me a path from source to sink, at the lowest cost you can.
- Agents: agree to be paid whatever you like; claim whatever you like; however, you can't show your paper to anyone.

Selfish routing example



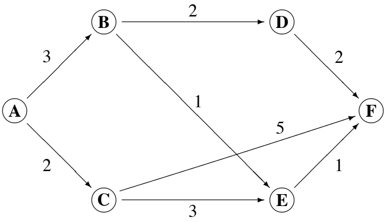
- What outcome will be selected by χ?

Selfish routing example



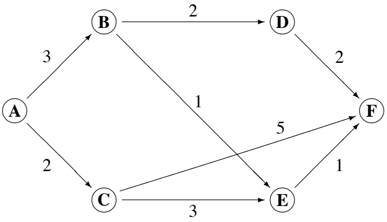
- What outcome will be selected by χ? path $ABEF$.

Selfish routing example



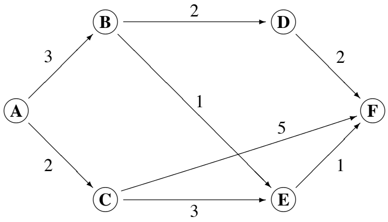
- What outcome will be selected by χ? path $ABEF$.
- How much will AC have to pay?

Selfish routing example



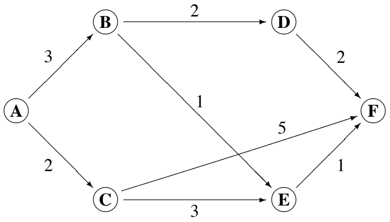
- What outcome will be selected by χ? path $ABEF$.
- How much will AC have to pay?
  - The shortest path taking his declaration into account has a length of 5, and imposes a cost of -5 on agents other than him (because it does not involve him). Likewise, the shortest path without AC's declaration also has a length of 5. Thus, his payment $p_{AC} = (-5) - (-5) = 0$.
  - This is what we expect, since AC is not pivotal.
  - Likewise, BD, CE, CF and DF will all pay zero.

Selfish routing example



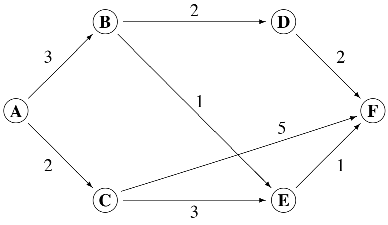
- How much will AB pay?

Selfish routing example



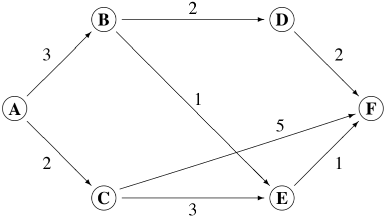
- How much will AB pay?
  - The shortest path taking AB's declaration into account has a length of 5, and imposes a cost of 2 on other agents.
  - The shortest path without AB is $ACEF$, which has a cost of 6.
  - Thus $p_{AB} = (-6) - (-2) = -4$.

Selfish routing example



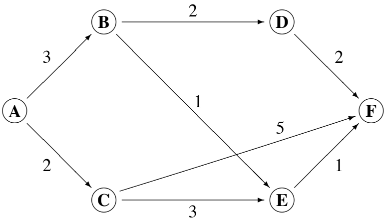
- How much will BE pay?

Selfish routing example



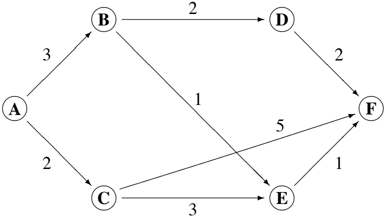
- How much will BE pay? $p_{BE} = (-6) - (-4) = -2$.

Selfish routing example



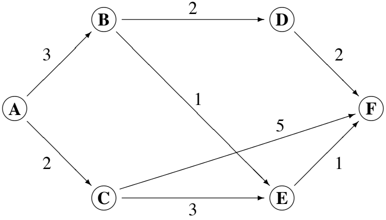
- How much will BE pay? $p_{BE} = (-6) - (-4) = -2$.
- How much will EF pay?

Selfish routing example



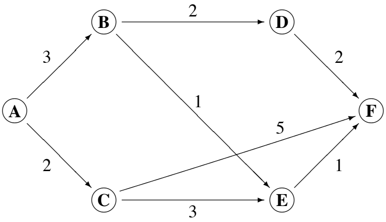
- How much will BE pay? $p_{BE} = (-6) - (-4) = -2.$
- How much will EF pay? $p_{EF} = (-7) - (-4) = -3.$

Selfish routing example



- How much will BE pay? $p_{BE} = (-6) - (-4) = -2.$
- How much will EF pay? $p_{EF} = (-7) - (-4) = -3.$
  - EF and BE have the same costs but are paid different amounts. Why?

Selfish routing example



- How much will BE pay? $p_{BE} = (-6) - (-4) = -2$.
- How much will EF pay? $p_{EF} = (-7) - (-4) = -3$.
  - EF and BE have the same costs but are paid different amounts. Why?
  - EF has more *market power*: for the other agents, the situation without EF is worse than the situation without BE.

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