CS 886: Multiagent Systems
Introduction to Mechanism Design

Kate Larson

Computer Science
University of Waterloo
Outline

1. Introduction
   - Introduction
   - Fundamentals

2. Mechanisms
   - Mechanism Design Problem
   - Direct Mechanisms
   - Revelation Principle
   - Gibbard-Satterthwaite
   - Quasi-Linear Preferences
   - Groves Mechanisms
Outline

1. Introduction
   - Introduction
   - Fundamentals

2. Mechanisms
   - Mechanism Design Problem
   - Direct Mechanisms
   - Revelation Principle
   - Gibbard-Satterthwaite
   - Quasi-Linear Preferences
   - Groves Mechanisms
Introduction

Game Theory
- Given a game we are able to analyse the strategies agents will follow

Social Choice
- Given a set of agents’ preferences we can choose some outcome
Introduction

Today **Mechanism Design**

- Game Theory + Social Choice

- Goal of Mechanism Design is to
  - Obtain some outcome (function of agents’ preferences)
  - But agents are rational
    - They may lie about their preferences

**Goal**

Define the rules of a game so that in equilibrium the agents do what we want.
Today **Mechanism Design**

- Game Theory + Social Choice

Goal of Mechanism Design is to

- Obtain some outcome (function of agents’ preferences)
- But agents are rational
- They may lie about their preferences

**Goal**

Define the rules of a game so that in equilibrium the agents do what we want.
Introduction

Today **Mechanism Design**

- Game Theory + Social Choice

- Goal of Mechanism Design is to
  - Obtain some outcome (function of agents’ preferences)
  - But agents are rational
    - They may lie about their preferences

**Goal**

Define the rules of a game so that in equilibrium the agents do what we want.
Today **Mechanism Design**
- Game Theory + Social Choice
- Goal of Mechanism Design is to
  - Obtain some outcome (function of agents’ preferences)
  - But agents are rational
    - They may lie about their preferences

**Goal**

Define the rules of a game so that in equilibrium the agents do what we want.
Fundamentals

- Set of possible outcomes $O$
- Set of agents $N$, $|N| = n$
  - Each agent $i$ has type $\theta_i \in \Theta_i$
  - Type captures all private information that is relevant to the agent’s decision making
- Utility $u_i(o, \theta_i)$ over outcome $o \in O$
- Recall: goal is to implement some system wide solution
  - Captured by a social choice function

$$f : \Theta_1 \times \ldots \times \Theta_n \rightarrow O$$

where $f(\theta_1, \ldots, \theta_n) = o$ is a collective choice
Fundamentals

- Set of possible outcomes $O$
- Set of agents $N$, $|N| = n$
  - Each agent $i$ has type $\theta_i \in \Theta_i$
  - Type captures all private information that is relevant to the agent’s decision making
- Utility $u_i(o, \theta_i)$ over outcome $o \in O$
- Recall: goal is to implement some system wide solution
  - Captured by a social choice function

$$f : \Theta_1 \times \ldots \times \Theta_n \rightarrow O$$

where $f(\theta_1, \ldots, \theta_n) = o$ is a collective choice
Fundamentals

- Set of possible outcomes $O$
- Set of agents $N$, $|N| = n$
  - Each agent $i$ has type $\theta_i \in \Theta_i$
  - Type captures all private information that is relevant to the agent’s decision making
- Utility $u_i(o, \theta_i)$ over outcome $o \in O$
- Recall: goal is to implement some system wide solution
  - Captured by a social choice function

$$f : \Theta_1 \times \ldots \times \Theta_n \rightarrow O$$

where $f(\theta_1, \ldots, \theta_n) = o$ is a collective choice
Fundamentals

- Set of possible outcomes $O$
- Set of agents $N$, $|N| = n$
  - Each agent $i$ has type $\theta_i \in \Theta_i$
  - Type captures all private information that is relevant to the agent’s decision making
- Utility $u_i(o, \theta_i)$ over outcome $o \in O$
- Recall: goal is to implement some system wide solution
  - Captured by a social choice function

$$f : \Theta_1 \times \ldots \times \Theta_n \rightarrow O$$

where $f(\theta_1, \ldots, \theta_n) = o$ is a collective choice
Fundamentals

- Set of possible outcomes $O$
- Set of agents $N$, $|N| = n$
  - Each agent $i$ has type $\theta_i \in \Theta_i$
  - Type captures all private information that is relevant to the agent’s decision making
- Utility $u_i(o, \theta_i)$ over outcome $o \in O$
- Recall: goal is to implement some system wide solution
  - Captured by a social choice function

$$f : \Theta_1 \times \ldots \times \Theta_n \rightarrow O$$

where $f(\theta_1, \ldots, \theta_n) = o$ is a collective choice
Examples of Social Choice Functions

- **Voting:**
  - Choose a candidate among a group

- Public project:
  - Decide whether to build a swimming pool whose cost must be funded by the agents themselves

- **Allocation:**
  - Allocate a single, indivisible item to one agent in a group
Examples of Social Choice Functions

- **Voting:**
  - Choose a candidate among a group

- **Public project:**
  - Decide whether to build a swimming pool whose cost must be funded by the agents themselves

- **Allocation:**
  - Allocate a single, indivisible item to one agent in a group
Examples of Social Choice Functions

- **Voting:**
  - Choose a candidate among a group

- **Public project:**
  - Decide whether to build a swimming pool whose cost must be funded by the agents themselves

- **Allocation:**
  - Allocate a single, indivisible item to one agent in a group
Mechanisms
Recall that we want to implement a social choice function
- Need to know agents’ preferences
- They may not reveal them to us truthfully
Example:
Outline

1 Introduction
   - Introduction
   - Fundamentals

2 Mechanisms
   - Mechanism Design Problem
   - Direct Mechanisms
   - Revelation Principle
   - Gibbard-Satterthwaite
   - Quasi-Linear Preferences
   - Groves Mechanisms
Mechanism Design Problem

- By having agents interact through an institution we might be able to solve the problem

Mechanism:

\[ M = (S_1, \ldots, S_n, g(\cdot)) \]

where

- \( S_i \) is the strategy space of agent \( i \)
- \( g : S_1 \times \ldots \times S_n \rightarrow O \) is the outcome function
Mechanism Design Problem

- By having agents interact through an institution we might be able to solve the problem

**Mechanism:**

\[ M = (S_1, \ldots, S_n, g(\cdot)) \]

where

- \( S_i \) is the strategy space of agent \( i \)
- \( g : S_1 \times \ldots \times S_n \rightarrow O \) is the outcome function
Implementation

Definition

A mechanism $M = (S_1, \ldots, S_n, g(\cdot))$ implements social choice function $f(\Theta)$ if there is an equilibrium strategy profile

$$s^* = (s^*_1(\theta_1), \ldots, s^*_n(\theta_n))$$

of the game induced by $M$ such that

$$g(s^*_1(\theta_1), \ldots, s^*_n(\theta_n)) = f(\theta_1, \ldots, \theta_n)$$

for all

$$(\theta_1, \ldots, \theta_n) \in \Theta_1 \times \ldots \times \Theta_n$$
Implementation

We did not specify the type of equilibrium in the definition

- Nash

\[ u_i(g(s_i^*(\theta_i), s_{-i}^*(\theta_{-i})), \theta_i) \geq u_i(g(s_i'(\theta_i), s_{-i}^*(\theta_{-i})), \theta_i) \]

\[ \forall i, \forall \theta_i, \forall s_i' \neq s_i^* \]

- Bayes-Nash

\[ E[u_i(g(s_i^*(\theta_i), s_{-i}^*(\theta_{-i})), \theta_i)] \geq E[u_i(g(s_i'(\theta_i), s_{-i}^*(\theta_{-i})), \theta_i)] \]

\[ \forall i, \forall \theta_i, \forall s_i' \neq s_i^* \]

- Dominant

\[ u_i(g(s_i^*(\theta_i), s_{-i}^*(\theta_{-i})), \theta_i) \geq u_i(g(s_i'(\theta_i), s_{-i}^*(\theta_{-i})), \theta_i) \]

\[ \forall i, \forall \theta_i, \forall s_i' \neq s_i^*, \forall s_{-i} \]
Implementation

We did not specify the type of equilibrium in the definition

- Nash

\[ u_i(g(s_i^*(\theta_i), s_{-i}^*(\theta_{-i})), \theta_i) \geq u_i(g(s_i'(\theta_i), s_{-i}^*(\theta_{-i})), \theta_i) \]

\[ \forall i, \forall \theta_i, \forall s_i' \neq s_i^* \]

- Bayes-Nash

\[ E[u_i(g(s_i^*(\theta_i), s_{-i}^*(\theta_{-i})), \theta_i)] \geq E[u_i(g(s_i'(\theta_i), s_{-i}^*(\theta_{-i})), \theta_i)] \]

\[ \forall i, \forall \theta_i, \forall s_i' \neq s_i^* \]

- Dominant

\[ u_i(g(s_i^*(\theta_i), s_{-i}^*(\theta_{-i})), \theta_i) \geq u_i(g(s_i'(\theta_i), s_{-i}^*(\theta_{-i})), \theta_i) \]

\[ \forall i, \forall \theta_i, \forall s_i' \neq s_i^*, \forall s_{-i} \]
Implementation

We did not specify the type of equilibrium in the definition

- Nash

\[ u_i(g(s_i^*(\theta_i), s_{-i}^*(\theta_{-i})), \theta_i) \geq u_i(g(s'_i(\theta_i), s_{-i}^*(\theta_{-i})), \theta_i) \]

\[ \forall i, \forall \theta_i, \forall s'_i \neq s_i^* \]

- Bayes-Nash

\[ E[u_i(g(s_i^*(\theta_i), s_{-i}^*(\theta_{-i})), \theta_i)] \geq E[u_i(g(s'_i(\theta_i), s_{-i}^*(\theta_{-i})), \theta_i)] \]

\[ \forall i, \forall \theta_i, \forall s'_i \neq s_i^* \]

- Dominant

\[ u_i(g(s_i^*(\theta_i), s_{-i}^*(\theta_{-i})), \theta_i) \geq u_i(g(s'_i(\theta_i), s_{-i}^*(\theta_{-i})), \theta_i) \]

\[ \forall i, \forall \theta_i, \forall s'_i \neq s_i^*, \forall s_{-i} \]
Properties for Mechanisms

- **Efficiency**
  - Select the outcome that maximizes total utility

- **Fairness**
  - Select outcome that minimizes the variance in utility

- **Revenue maximization**
  - Select outcome that maximizes revenue to a seller (or, utility to one of the agents)

- **Budget-balanced**
  - Implement outcomes that have balanced transfers across agents

- **Pareto Optimal**
  - Only implement outcomes \( o^\ast \) for which for all \( o' \neq o^\ast \) either
    \[
    u_i(o', \theta_i) = u_i(o^\ast, \theta_i) \forall i \text{ or } \exists i \in N \text{ with } u_i(o', \theta_i) < u_i(o^\ast, \theta_i)
    \]
Participation Constraints

We can not force agents to participate in the mechanism. Let \( \hat{u}_i(\theta_i) \) denote the (expected) utility to agent \( i \) with type \( \theta_i \) of its outside option.

- **ex ante individual-rationality**: agents choose to participate before they know their own type

\[
E_{\theta \in \Theta}[u_i(f(\theta), \theta_i)] \geq E_{\theta_i \in \Theta_i} \hat{u}_i(\theta_i)
\]

- **interim individual-rationality**: agents can withdraw once they know their own type

\[
E_{\theta_{-i} \in \Theta_{-i}}[u_i(f(\theta_i, \theta_{-i}), \theta_i)] \geq \hat{u}_i(\theta_i)
\]

- **ex-post individual-rationality**: agents can withdraw from the mechanism at the end

\[
u_i(f(\theta_i), \theta_i) \geq \hat{u}_i(\theta_i)
\]
Participation Constraints

We can not force agents to participate in the mechanism. Let $\hat{u}_i(\theta_i)$ denote the (expected) utility to agent $i$ with type $\theta_i$ of its outside option.

- **ex ante individual-rationality**: agents choose to participate before they know their own type

  \[ E_{\theta \in \Theta}[u_i(f(\theta), \theta_i)] \geq E_{\theta_i \in \Theta_i} \hat{u}_i(\theta_i) \]

- **interim individual-rationality**: agents can withdraw once they know their own type

  \[ E_{\theta_{-i} \in \Theta_{-i}}[u_i(f(\theta_i, \theta_{-i}), \theta_i)] \geq \hat{u}_i(\theta_i) \]

- **ex-post individual-rationality**: agents can withdraw from the mechanism at the end

  \[ u_i(f(\theta_i), \theta_i) \geq \hat{u}_i(\theta_i) \]
Participation Constraints

We can not force agents to participate in the mechanism. Let \( \hat{u}_i(\theta_i) \) denote the (expected) utility to agent \( i \) with type \( \theta_i \) of its outside option.

- **ex ante individual-rationality**: agents choose to participate before they know their own type
  \[
  E_{\theta \in \Theta}[u_i(f(\theta), \theta_i)] \geq E_{\theta_i \in \Theta_i}[\hat{u}_i(\theta_i)]
  \]

- **interim individual-rationality**: agents can withdraw once they know their own type
  \[
  E_{\theta_{-i} \in \Theta_{-i}}[u_i(f(\theta_i, \theta_{-i}), \theta_i)] \geq \hat{u}_i(\theta_i)
  \]

- **ex-post individual-rationality**: agents can withdraw from the mechanism at the end
  \[
  u_i(f(\theta), \theta_i) \geq \hat{u}_i(\theta_i)
  \]
Participation Constraints

We can not force agents to participate in the mechanism. Let \( \hat{u}_i(\theta_i) \) denote the (expected) utility to agent \( i \) with type \( \theta_i \) of its outside option.

- **ex ante individual-rationality**: agents choose to participate before they know their own type
  \[
  E_{\theta \in \Theta}[u_i(f(\theta), \theta_i)] \geq E_{\theta_i \in \Theta_i}\hat{u}_i(\theta_i)
  \]

- **interim individual-rationality**: agents can withdraw once they know their own type
  \[
  E_{\theta_{-i} \in \Theta_{-i}}[u_i(f(\theta_i, \theta_{-i}), \theta_i)] \geq \hat{u}_i(\theta_i)
  \]

- **ex-post individual-rationality**: agents can withdraw from the mechanism at the end
  \[
  u_i(f(\theta), \theta_i) \geq \hat{u}_i(\theta_i)
  \]
Outline

1. Introduction
   - Introduction
   - Fundamentals

2. Mechanisms
   - Mechanism Design Problem
   - Direct Mechanisms
     - Revelation Principle
     - Gibbard-Satterthwaite
     - Quasi-Linear Preferences
     - Groves Mechanisms
Direct Mechanisms

**Definition**

A **direct mechanism** is a mechanism where

$$S_i = \Theta_i \text{ for all } i$$

and

$$g(\theta) = f(\theta) \text{ for all } \theta \in \Theta_1 \times \ldots \times \Theta_n$$
Incentive Compatibility

**Definition**

A direct mechanism is **incentive compatible** if it has an equilibrium $s^*$ where

$$s^*_i(\theta_i) = \theta_i$$

for all $\theta_i \in \Theta_i$ and for all $i$. That is, truth-telling by all agents is an equilibrium.

**Definition**

A direct mechanism is **strategy-proof** if it is incentive compatible and the equilibrium is a dominant strategy equilibrium.
Incentive Compatibility

**Definition**

A direct mechanism is **incentive compatible** if it has an equilibrium $s^*$ where

$$s^*_i(\theta_i) = \theta_i$$

for all $\theta_i \in \Theta_i$ and for all $i$. That is, truth-telling by all agents is an equilibrium.

**Definition**

A direct mechanism is **strategy-proof** if it is incentive compatible and the equilibrium is a dominant strategy equilibrium.
Outline

1. Introduction
   - Introduction
   - Fundamentals

2. Mechanisms
   - Mechanism Design Problem
   - Direct Mechanisms
   - Revelation Principle
   - Gibbard-Satterthwaite
   - Quasi-Linear Preferences
   - Groves Mechanisms
Revelation Principle

Theorem

Suppose there exists a mechanism $M = (S_1, \ldots, S_n, g(\cdot))$ that implements social choice function $f$ in dominant strategies. Then there is a direct strategy-proof mechanism $M'$ which also implements $f$.

[Gibbard 73; Green & Laffont 77; Myerson 79]

“The computations that go on within the mind of any bidder in the nondirect mechanism are shifted to become part of the mechanism in the direct mechanism.”

[McAfee & McMillan 87]
Revelation Principle

Theorem

Suppose there exists a mechanism $M = (S_1, \ldots, S_n, g(\cdot))$ that implements social choice function $f$ in dominant strategies. Then there is a direct strategy-proof mechanism $M'$ which also implements $f$.

[Gibbard 73; Green & Laffont 77; Myerson 79]

“The computations that go on within the mind of any bidder in the nondirect mechanism are shifted to become part of the mechanism in the direct mechanism.”

[McAfee & McMillan 87]
Revelation Principle: Proof

1. Construct mechanism $M = (S, g)$ that implements $f(\theta)$ in dominant strategies. Then $g(s^*(\theta)) = f(\theta)$ for all $\theta \in \Theta$ where $s^*$ is a dominant strategy equilibrium.

2. Construct direct mechanism $M' = (\Theta, f(\Theta))$.

3. By contradiction suppose $\exists \theta'_i \neq \theta_i$ s.t. $u_i(f(\theta'_i, \theta_{-i}), \theta_i) > u_i(f(\theta_i, \theta_{-i}), \theta_i)$ for some $\theta'_i \neq \theta_i$, some $\theta_{-i}$.

4. But, because $f(\theta) = g(s^*(\theta))$ this implies that $u_i(g(s^*_i(\theta'_i), s^*_{-i}(\theta_{-i})), \theta_i) > u_i(g(s^*_i(\theta_i), s^*_{-i}(\theta_{-i})), \theta_i)$ which contradicts the strategyproofness of $s^*$ in mechanism $M$. 

Kate Larson  Mechanism Design
Revelation Principle: Proof

1. Construct mechanism $M = (S, g)$ that implements $f(\theta)$ in dominant strategies. Then $g(s^*(\theta)) = f(\theta)$ for all $\theta \in \Theta$ where $s^*$ is a dominant strategy equilibrium.

2. Construct direct mechanism $M' = (\Theta, f(\Theta))$.

3. By contradiction suppose $\exists \theta'_i \neq \theta_i$ s.t. $u_i(f(\theta'_i, \theta_{-i}), \theta_i) > u_i(f(\theta_i, \theta_{-i}), \theta_i)$ for some $\theta'_i \neq \theta_i$, some $\theta_{-i}$.

4. But, because $f(\theta) = g(s^*(\theta))$ this implies that $u_i(g(s^*_i(\theta'_i), s^*_{-i}(\theta_{-i})), \theta_i) > u_i(g(s^*_i(\theta_i), s^*_{-i}(\theta_{-i})), \theta_i)$ which contradicts the strategyproofness of $s^*$ in mechanism $M$. 
Revelation Principle: Proof

1. Construct mechanism \( M = (S, g) \) that implements \( f(\theta) \) in dominant strategies. Then \( g(s^*(\theta)) = f(\theta) \) for all \( \theta \in \Theta \) where \( s^* \) is a dominant strategy equilibrium.

2. Construct direct mechanism \( M' = (\Theta, f(\Theta)) \).

3. By contradiction suppose

\[ \exists \theta_i' \neq \theta_i \text{ s.t. } u_i(f(\theta_i', \theta_{-i}), \theta_i) > u_i(f(\theta_i, \theta_{-i}), \theta_i) \]

for some \( \theta_i' \neq \theta_i \), some \( \theta_{-i} \).

4. But, because \( f(\theta) = g(s^*(\theta)) \) this implies that

\[ u_i(g(s^*_i(\theta_i'), s^*_{-i}(\theta_{-i})), \theta_i) > u_i(g(s^*_i(\theta_i), s^*_{-i}(\theta_{-i})), \theta_i) \]

which contradicts the strategyproofness of \( s^* \) in mechanism \( M \).
Revelation Principle: Proof

1. Construct mechanism $M = (S, g)$ that implements $f(\theta)$ in dominant strategies. Then $g(s^*(\theta)) = f(\theta)$ for all $\theta \in \Theta$ where $s^*$ is a dominant strategy equilibrium.

2. Construct direct mechanism $M' = (\Theta, f(\Theta))$.

3. By contradiction suppose $\exists \theta'_i \neq \theta_i \text{ s.t. } u_i(f(\theta'_i, \theta_{-i}), \theta_i) > u_i(f(\theta_i, \theta_{-i}), \theta_i)$

   for some $\theta'_i \neq \theta_i$, some $\theta_{-i}$.

4. But, because $f(\theta) = g(s^*(\theta))$ this implies that $u_i(g(s^*_i(\theta'_i), s^*_{-i}(\theta_{-i})), \theta_i) > u_i(g(s^*_i(\theta_i), s^*_{-i}(\theta_{-i})), \theta_i)$

   which contradicts the strategyproofness of $s^*$ in mechanism $M$. 
Revelation Principle: Intuition

**Constructed “direct revelation” mechanism**

- Agent 1’s preferences
- Agent |A|’s preferences

**Original “complex” “indirect” mechanism**

- Strategy formulator
- Strategy

**Outcome**

Kate Larson

Mechanism Design
Theoretical Implications

**Literal interpretation:** Need only study direct mechanisms

- A modeler can limit the search for an optimal mechanism to the class of direct IC mechanisms
- If no direct mechanism can implement social choice function $f$ then no mechanism can
- Useful because the space of possible mechanisms is huge
Theoretical Implications

- **Literal interpretation:** Need only study direct mechanisms
  - A modeler can limit the search for an optimal mechanism to the class of direct IC mechanisms
  - If no direct mechanism can implement social choice function $f$ then no mechanism can
  - Useful because the space of possible mechanisms is huge
Theoretical Implications

- **Literal interpretation:** Need only study direct mechanisms
  - A modeler can limit the search for an optimal mechanism to the class of direct IC mechanisms
  - If no direct mechanism can implement social choice function $f$ then no mechanism can
  - Useful because the space of possible mechanisms is huge
**Theoretical Implications**

- **Literal interpretation:** Need only study direct mechanisms
  - A modeler can limit the search for an optimal mechanism to the class of direct IC mechanisms
  - If no direct mechanism can implement social choice function $f$ then no mechanism can
  - Useful because the space of possible mechanisms is huge
Practical Implications

- Incentive-compatibility is “free”
  - Any outcome implemented by mechanism $M$ can be implemented by incentive-compatible mechanism $M'$
- “Fancy” mechanisms are unnecessary
  - Any outcome implemented by a mechanism with complex strategy space $S$ can be implemented by a direct mechanism

**BUT** Lots of mechanisms used in practice are not direct and incentive-compatible!
Practical Implications

- Incentive-compatibility is “free”
  - Any outcome implemented by mechanism $M$ can be implemented by incentive-compatible mechanism $M'$
- “Fancy” mechanisms are unnecessary
  - Any outcome implemented by a mechanism with complex strategy space $S$ can be implemented by a direct mechanism

BUT Lots of mechanisms used in practice are not direct and incentive-compatible!
Practical Implications

- Incentive-compatibility is “free”
  - Any outcome implemented by mechanism $M$ can be implemented by incentive-compatible mechanism $M'$
- “Fancy” mechanisms are unnecessary
  - Any outcome implemented by a mechanism with complex strategy space $S$ can be implemented by a direct mechanism

**BUT** Lots of mechanisms used in practice are not direct and incentive-compatible!
Introduction

Mechanisms

Mechanism Design Problem
Direct Mechanisms
Revelation Principle
Gibbard-Satterthwaite
Quasi-Linear Preferences
Groves Mechanisms

Outline

1. Introduction
   - Introduction
   - Fundamentals

2. Mechanisms
   - Mechanism Design Problem
   - Direct Mechanisms
   - Revelation Principle
   - Gibbard-Satterthwaite
   - Quasi-Linear Preferences
   - Groves Mechanisms
Quick Review

We now know:
- What a mechanism is
- What it means for a SCF to be dominant-strategy implementable
- Revelation Principle

We do not yet know:
- What types of SCF are dominant-strategy implementable
Quick Review

We now know

- What a mechanism is
- What it means for a SCF to be dominant-strategy implementable
- Revelation Principle

We do not yet know

- What types of SCF are dominant-strategy implementable
Gibbard-Satterthwaite Impossibility

Theorem
Assume that

- $O$ is finite and $|O| \geq 3$,
- each $o \in O$ can be achieved by SCF $f$ for some $\theta$, and
- $\Theta$ includes all possible strict orderings over $O$.

Then $f$ is implementable in dominant strategies (strategy-proof) if and only if it is dictatorial.

Definition
SCF $f$ is **dictatorial** if there is an agent $i$ such that for all $\theta$

$$f(\theta) \in \{ o \in O | u_i(o, \theta) \geq u_i(o', \theta) \forall o' \in O \}$$
Theorem

Assume that

- \( O \) is finite and \( |O| \geq 3 \),
- each \( o \in O \) can be achieved by SCF \( f \) for some \( \theta \), and
- \( \Theta \) includes all possible strict orderings over \( O \).

Then \( f \) is implementable in dominant strategies (strategy-proof) if and only if it is dictatorial.

Definition

SCF \( f \) is **dictatorial** if there is an agent \( i \) such that for all \( \theta \)

\[
f(\theta) \in \{ o \in O | u_i(o, \theta_i) \geq u_i(o', \theta_i) \forall o' \in O \}
\]
Circumventing Gibbard-Satterthwaite

- Use a weaker equilibrium concept
- Design mechanisms where computing a beneficial manipulation is hard
- Randomization
- Restrict the structure of agents’ preferences
Outline

1. Introduction
   - Introduction
   - Fundamentals

2. Mechanisms
   - Mechanism Design Problem
   - Direct Mechanisms
   - Revelation Principle
   - Gibbard-Satterthwaite
   - Quasi-Linear Preferences
   - Groves Mechanisms
Quasi-linear preferences

- **Outcome** $o = (x, t_1, \ldots, t_n)$
  - $x$ is a “project choice”
  - $t_i \in \mathbb{R}$ are transfers (money)

- Utility function of agent $i$
  $$u_i(o, \theta_i) = v_i(x, \theta_i) - t_i$$

- Quasi-linear mechanism
  $$M = (S_1, \ldots, S_n, g(\cdot))$$

where
  $$g(\cdot) = (x(\cdot), t_1(\cdot), \ldots, t_n(\cdot))$$
Quasi-linear preferences

- Outcome $o = (x, t_1, \ldots, t_n)$
  - $x$ is a “project choice”
  - $t_i \in \mathbb{R}$ are transfers (money)
- Utility function of agent $i$
  $$u_i(o, \theta_i) = v_i(x, \theta_i) - t_i$$

- Quasi-linear mechanism
  $$M = (S_1, \ldots, S_n, g(\cdot))$$
  where
  $$g(\cdot) = (x(\cdot), t_1(\cdot), \ldots, t_n(\cdot))$$
Quasi-linear preferences

- Outcome \( o = (x, t_1, \ldots, t_n) \)
  - \( x \) is a “project choice”
  - \( t_i \in \mathbb{R} \) are transfers (money)
- Utility function of agent \( i \)
  \[
u_i(o, \theta_i) = v_i(x, \theta_i) - t_i\]
- Quasi-linear mechanism
  \[
M = (S_1, \ldots, S_n, g(\cdot))
  \]
  where
  \[
g(\cdot) = (x(\cdot), t_1(\cdot), \ldots, t_n(\cdot))\]
Social Choice Functions and Quasi-linearity

- SCF is **efficient** if for all $\theta$

$$\sum_{i=1}^{n} v_i(x(\theta), \theta_i) \geq \sum_{i=1}^{n} v_i(x'(\theta), \theta_i) \forall x'(\theta)$$

This is also known as **social welfare maximizing**

- SCF is **budget-balanced** if

$$\sum_{i=1}^{n} t_i(\theta) = 0$$

Weakly budget-balanced if

$$\sum_{i=1}^{n} t_i(\theta) \geq 0$$
Social Choice Functions and Quasi-linearity

- SCF is **efficient** if for all $\theta$
  \[ \sum_{i=1}^{n} v_i(x(\theta), \theta_i) \geq \sum_{i=1}^{n} v_i(x'(\theta), \theta_i) \forall x'(\theta) \]

  This is also known as **social welfare maximizing**

- SCF is **budget-balanced** if
  \[ \sum_{i=1}^{n} t_i(\theta) = 0 \]

  Weakly budget-balanced if
  \[ \sum_{i=1}^{n} t_i(\theta) \geq 0 \]
Outline

1. Introduction
   - Introduction
   - Fundamentals

2. Mechanisms
   - Mechanism Design Problem
   - Direct Mechanisms
   - Revelation Principle
   - Gibbard-Satterthwaite
   - Quasi-Linear Preferences
   - Groves Mechanisms
A Groves mechanism \( M = (S_1, \ldots, S_n, (x, t_1, \ldots, t_n)) \) is defined by

- **Choice rule**

\[
x^*(\theta) = \arg \max_x \sum_i v_i(x, \theta_i)
\]

- **Transfer rules**

\[
t_i(\theta) = h_i(\theta_{-i}) - \sum_{j \neq i} v_j(x^*(\theta), \theta_j)
\]

where \( h_i(\cdot) \) is an (arbitrary) function that does not depend on the reported type \( \theta_i' \) of agent \( i \).
Groves Mechanisms [Groves 73]

A Groves mechanism $M = (S_1, \ldots, S_n, (x, t_1, \ldots, t_n))$ is defined by

1. **Choice rule**

   $$x^*(\theta) = \arg \max_x \sum_i v_i(x, \theta_i)$$

2. **Transfer rules**

   $$t_i(\theta) = h_i(\theta_{-i}) - \sum_{j \neq i} v_j(x^*(\theta), \theta_j)$$

   where $h_i(\cdot)$ is an (arbitrary) function that does not depend on the reported type $\theta'_i$ of agent $i$. 
Groves Mechanisms [Groves 73]

A Groves mechanism \( M = (S_1, \ldots, S_n, (x, t_1, \ldots, t_n)) \) is defined by

- **Choice rule**
  \[
  x^*(\theta) = \arg \max_x \sum_i v_i(x, \theta_i)
  \]

- **Transfer rules**
  \[
  t_i(\theta) = h_i(\theta_{-i}) - \sum_{j \neq i} v_j(x^*(\theta), \theta_j)
  \]

where \( h_i(\cdot) \) is an (arbitrary) function that does not depend on the reported type \( \theta_i' \) of agent \( i \).
Groves Mechanisms

Theorem

*Groves mechanisms are strategy-proof and efficient.*

We have gotten around Gibbard-Satterthwaite.
**Proof**

Agent $i$’s utility for strategy $\hat{\theta}_i$, given $\hat{\theta}_{-i}$ from agents $j \neq i$ is

\[
\begin{align*}
  u_i(\hat{\theta}_i) &= v_i(x^*(\hat{\theta}, \theta_i) - t_i(\hat{\theta}) \\
  &= v_i(x^*(\hat{\theta}, \theta_i) + \sum_{j \neq i} v_j(x^*(\hat{\theta}, \hat{\theta}_j) - h_i(\hat{\theta}_{-i})
\end{align*}
\]

Ignore $h_i(\hat{\theta}_{-i})$ and notice $x^*(\hat{\theta}) = \arg \max_x \sum_i v_i(x, \hat{\theta}_i)$ i.e it maximizes the sum of reported values. Therefore, agent $i$ should announce $\hat{\theta}_i = \theta_i$ to maximize its own payoff.

**Thm:** Groves mechanisms are unique (up to $h_i(\theta_{-i})$).
Vickrey-Clarke-Groves Mechanism
aka Clarke mechanisms, aka Pivotal mechanism

- Implement efficient outcome

$$x^* = \arg\max_x \sum_i v_i(x, \theta_i)$$

- Compute transfers

$$t_i(\theta) = \sum_{j \neq i} v_j(x^{-i}, \theta_j) - \sum_{j \neq i} v_j(x^*, \theta_j)$$

where $$x^{-i} = \arg\max_x \sum_{j \neq i} v_j(x, \theta_j)$$

VCG are efficient and strategy-proof.
Vickrey-Clarke-Groves Mechanism
aka Clarke mechanism, aka Pivotal mechanism

- Implement efficient outcome

\[ x^* = \arg \max_x \sum_i v_i(x, \theta_i) \]

- Compute transfers

\[ t_i(\theta) = \sum_{j \neq i} v_j(x^{-i}, \theta_j) - \sum_{j \neq i} v_j(x^*, \theta_j) \]

where \( x^{-i} = \arg \max_x \sum_{j \neq i} v_j(x, \theta_j) \)

VCG are efficient and strategy-proof.
VCG Mechanism

Agent’s equilibrium utility is

\[ u_i((x^*, t), \theta_i) = v_i(x^*, \theta_i) - \left[ \sum_{j \neq i} v_j(x^{-i}, \theta_j) - \sum_{j \neq i} v_j(x^*, \theta_j) \right] \]

\[= \sum_{j=1}^{n} v_j(x^*, \theta_j) - \sum_{j \neq i} v_j(x^{-i}, \theta_j) \]

\[= \text{marginal contribution to the welfare of the system} \]
Example: Building a Pool

- Cost of building the pool is $300
- If together all agents value the pool more than $300 then it will be built

Clarke Mechanism

- Each agent announces $v_i$ and if $\sum_i v_i \geq 300$ then it is built
- Payments $t_i = \sum_{j \neq i} v_j(x^{-i}, v_j) - \sum_{j \neq i} v_j(x^*, v_j)$

Assume $v_1 = 50$, $v_2 = 50$, $v_3 = 250$. Clearly, the pool should be built.

Transfers: $t_1 = (250 + 50) - (250 + 50) = 0 = t_2$ and $t_3 = (0) - (100) = -100$. Note that it is not budget-balanced.
Example: Building a Pool

- Cost of building the pool is $300
- If together all agents value the pool more than $300 then it will be built
- Clarke Mechanism
  - Each agent announces $v_i$ and if $\sum_j v_j \geq 300$ then it is built
  - Payments $t_i = \sum_{j \neq i} v_j(x^{-i}, v_j) - \sum_{j \neq i} v_j(x^*, v_j)$

Assume $v_1 = 50$, $v_2 = 50$, $v_3 = 250$. Clearly, the pool should be built.

Transfers: $t_1 = (250 + 50) - (250 + 50) = 0 = t_2$ and $t_3 = (0) - (100) = -100$. Note that it is not budget-balanced.
Example: Building a Pool

- Cost of building the pool is $300
- If together all agents value the pool more than $300 then it will be built

Clarke Mechanism

- Each agent announces $v_i$ and if $\sum_i v_i \geq 300$ then it is built
- Payments $t_i = \sum_{j \neq i} v_j(x^{-i}, v_j) - \sum_{j \neq i} v_j(x^*, v_j)$

Assume $v_1 = 50$, $v_2 = 50$, $v_3 = 250$. Clearly, the pool should be built.

Transfers: $t_1 = (250 + 50) - (250 + 50) = 0 = t_2$ and $t_3 = (0) - (100) = -100$. Note that it is not budget-balanced.
Vickrey Auction

- Highest bidder gets the item and pays an amount equal to the second highest bid
- This is also a VCG mechanism
  - Allocation rule: get item if $b_i = \max_j [b_j]$
  - Every agent pays

$$t_i(v) = \sum_{j \neq i} v_j(x^{-i}, v_j) - \sum_{j \neq i} v_j(x^*, v_j)$$

Note that $\sum_{j \neq i} v_j(x^{-i}, v_j) = \max_{j \neq i} b_j$ and

$$\sum_{j \neq i} v_j(x^*, v_j) = \begin{cases} \max_{j \neq i} [b_j] & \text{if } i \text{ is not the highest bidder} \\ 0 & \text{if it is.} \end{cases}$$
Vickrey Auction

- Highest bidder gets the item and pays an amount equal to the second highest bid
- This is also a VCG mechanism
  - Allocation rule: get item if \( b_i = \max_j [b_j] \)
  - Every agent pays

\[
t_i(v) = \sum_{j \neq i} v_j(x^{-i}, v_j) - \sum_{j \neq i} v_j(x^*, v_j)
\]

Note that \( \sum_{j \neq i} v_j(x^{-i}, v_j) = \max_{j \neq i} b_j \) and

\[
\sum_{j \neq i} v_j(x^*, v_j) = \begin{cases} 
\max_{j \neq i} [b_j] & \text{if } i \text{ is not the highest bidder} \\
0 & \text{if it is}.
\end{cases}
\]
London Bus System\(^1\)

- 5 million passengers daily
- 7500 buses
- 700 routes
- The system has been privatized since 1997 by using competitive tendering
- *Idea*: Run an auction to allocate routes to companies

\(^1\)As of April 2004
Auction Protocol

- Let $G$ be set of all routes, $I$ be the set of bidders
- Agent $i$ submits bid $v_i(S)$ for all bundles $S \subseteq G$
- Compute allocation $S^*$ to maximize sum of reported bids
  \[
  V^*(I) = \max_{(S_1, \ldots, S_n)} \sum_i v_i(S_i)
  \]
- Compute best allocation without each agent
  \[
  V^*(I \setminus i) = \max_{(S_1, \ldots, S_n)} \sum_{j \neq i} v_j^*(S_j)
  \]
- Allocate each agent $S_i^*$, each agent pays
  \[
  P(i) = v_i^*(S_i^*) - [V^*(I) - V^*(I \setminus i)]
  \]
Auction Protocol

- Let $G$ be the set of all routes, $I$ be the set of bidders
- Agent $i$ submits bid $v_i(S)$ for all bundles $S \subseteq G$
- Compute allocation $S^*$ to maximize sum of reported bids
  \[ V^*(I) = \max_{(S_1, \ldots, S_n)} \sum_i v_i(S_i) \]
- Compute best allocation without each agent
  \[ V^*(I \setminus i) = \max_{(S_1, \ldots, S_n)} \sum_{j \neq i} v_j^*(S_j) \]
- Allocate each agent $S_i^*$, each agent pays
  \[ P(i) = v_i^*(S_i^*) - [V^*(I) - V^*(I \setminus i)] \]
Auction Protocol

- Let $G$ be the set of all routes, $I$ the set of bidders
- Agent $i$ submits bid $v_i(S)$ for all bundles $S \subseteq G$
- Compute allocation $S^*$ to maximize sum of reported bids
  \[
  V^*(I) = \max_{(S_1,\ldots,S_n)} \sum_i v_i(S_i)
  \]

- Compute best allocation without each agent
  \[
  V^*(I \setminus i) = \max_{(S_1,\ldots,S_n)} \sum_{j \neq i} v^*_j(S_j)
  \]

- Allocate each agent $S^*_i$, each agent pays
  \[
  P(i) = v^*_i(S^*_i) - [V^*(I) - V^*(I \setminus i)]
  \]
Auction Protocol

- Let $G$ be set of all routes, $I$ be the set of bidders
- Agent $i$ submits bid $v_i(S)$ for all bundles $S \subseteq G$
- Compute allocation $S^*$ to maximize sum of reported bids

$$V^*(I) = \max_{(S_1, \ldots, S_n)} \sum_i v_i(S_i)$$

- Compute best allocation without each agent

$$V^*(I \setminus i) = \max_{(S_1, \ldots, S_n)} \sum_{j \neq i} v_j^*(S_j)$$

- Allocate each agent $S_i^*$, each agent pays

$$P(i) = v_i^*(S_i^*) - [V^*(I) - V^*(I \setminus i)]$$
For Further Reading I
