Maskin Meets Abreu and Matsushima

Yi-Chun Chen (NUS) Takashi Kunimoto (SMU) Yifei Sun (UIBE)
Siyang Xiong (Bristol)

July 10, 2018
IMS, NUS

Nash Implementation

- A social planner has an objective summarized by a social choice function (SCF) $f : \Theta \longrightarrow X$.

Nash Implementation

- A social planner has an objective summarized by a social choice function (SCF) $f : \Theta \longrightarrow X$.

- An SCF is Nash implementable if there exists a mechanism/game-form that satisfies two requirements:

Nash Implementation

- A social planner has an objective summarized by a social choice function (SCF) $f : \Theta \longrightarrow X$.

- An SCF is Nash implementable if there exists a mechanism/game-form that satisfies two requirements:
  - Existence: there always exists a good NE whose outcome is socially desirable;

Nash Implementation

- A social planner has an objective summarized by a social choice function (SCF) $f : \Theta \longrightarrow X$.
- An SCF is Nash implementable if there exists a mechanism/game-form that satisfies two requirements:
  - Existence: there always exists a good NE whose outcome is socially desirable;
  - Uniqueness: every NE results in the socially desirable outcome.

Maskin (1977, 1999)

- Maskin proposes a monotonicity condition and shows that it is a necessary and almost sufficient condition for Nash implementation.

Maskin (1977, 1999)

- Maskin proposes a monotonicity condition and shows that it is a necessary and almost sufficient condition for Nash implementation.
- The sufficiency result has been criticized/improved on different aspects:

Maskin (1977, 1999)

- Maskin proposes a monotonicity condition and shows that it is a necessary and almost sufficient condition for Nash implementation.
- The sufficiency result has been criticized/improved on different aspects:
    - Integer game;

Maskin (1977, 1999)

- Maskin proposes a monotonicity condition and shows that it is a necessary and almost sufficient condition for Nash implementation.
- The sufficiency result has been criticized/improved on different aspects:
  - Integer game;
  - Mixed NE;

Maskin (1977, 1999)

- Maskin proposes a monotonicity condition and shows that it is a necessary and almost sufficient condition for Nash implementation.

- The sufficiency result has been criticized/improved on different aspects:
    - Integer game;
    - Mixed NE;
    - Two agents.

Maskin (1977, 1999)

- Maskin proposes a monotonicity condition and shows that it is a necessary and almost sufficient condition for Nash implementation.
- The sufficiency result has been criticized/improved on different aspects:
  - Integer game;
  - Mixed NE;
  - Two agents.
- Revelation principle does not hold.

Abreu and Matsushima (1992, 1994)

- AM (1992, 94) dispense with Maskin monotonicity and resolve all the issues.

Abreu and Matsushima (1992, 1994)

- AM (1992, 94) dispense with Maskin monotonicity and resolve all the issues.

- AM consider an economic environment with lotteries and transfers; moreover,

Abreu and Matsushima (1992, 1994)

- AM (1992, 94) dispense with Maskin monotonicity and resolve all the issues.

- AM consider an economic environment with lotteries and transfers; moreover,
    - AM (1992) appeal to virtual implementation and use rationalizability;

Abreu and Matsushima (1992, 1994)

- AM (1992, 94) dispense with Maskin monotonicity and resolve all the issues.
- AM consider an economic environment with lotteries and transfers; moreover,
  - AM (1992) appeal to virtual implementation and use rationalizability;
  - AM (1994) appeal to iterated weak dominance rather than NE.

Robustness Necessitates Monotonicity

- Complete information is an idealization

Robustness Necessitates Monotonicity

- Complete information is an idealization
  - Agents receive signals (=states) that may be wrong with small yet positive probability.

Robustness Necessitates Monotonicity

- Complete information is an idealization
  - Agents receive signals (=states) that may be wrong with small yet positive probability.
  - Want to implement f robustly: i.e., the closure of equilibrium outcomes under nearly complete information.

Robustness Necessitates Monotonicity

- Complete information is an idealization
  - Agents receive signals (=states) that may be wrong with small yet positive probability.
  - Want to implement f robustly: i.e., the closure of equilibrium outcomes under nearly complete information.
- Monotonicity is necessary for robustness:

Robustness Necessitates Monotonicity

- Complete information is an idealization
  - Agents receive signals (=states) that may be wrong with small yet positive probability.
  - Want to implement f robustly: i.e., the closure of equilibrium outcomes under nearly complete information.

- Monotonicity is necessary for robustness:
  - UNE: Chung and Ely (2003);

Robustness Necessitates Monotonicity

- Complete information is an idealization
  - Agents receive signals (=states) that may be wrong with small yet positive probability.
  - Want to implement f robustly: i.e., the closure of equilibrium outcomes under nearly complete information.
- Monotonicity is necessary for robustness:
  - UNE: Chung and Ely (2003);
  - SPE: Aghion et al. (2012).

Robustness Necessitates Monotonicity

- Complete information is an idealization
  - Agents receive signals (=states) that may be wrong with small yet positive probability.
  - Want to implement f robustly: i.e., the closure of equilibrium outcomes under nearly complete information.
- Monotonicity is necessary for robustness:
  - UNE: Chung and Ely (2003);
  - SPE: Aghion et al. (2012).
- Harsanyi's purification argument: for robustness we can't ignore mixed NE.

Maskin Meets Abreu and Matsushima

- In a (finite) economic environment with lotteries and transfers, we unify the two approaches to achieve NE implementation.

Maskin Meets Abreu and Matsushima

- In a (finite) economic environment with lotteries and transfers, we unify the two approaches to achieve NE implementation.

- Main results:

Maskin Meets Abreu and Matsushima

- In a (finite) economic environment with lotteries and transfers, we unify the two approaches to achieve NE implementation.

- Main results:
  - When $I \geq 3$, an SCF is pure Nash implementable by a direct mechanism if and only if it satisfies Maskin monotonicity.

Maskin Meets Abreu and Matsushima

- In a (finite) economic environment with lotteries and transfers, we unify the two approaches to achieve NE implementation.

- Main results:
  - When $I \geq 3$, an SCF is pure Nash implementable by a direct mechanism if and only if it satisfies Maskin monotonicity.
  - When $I \geq 2$, an SCF is mixed Nash implementable by a finite mechanism if and only if it satisfies Maskin monotonicity.

Maskin Meets Abreu and Matsushima

- In a (finite) economic environment with lotteries and transfers, we unify the two approaches to achieve NE implementation.

- Main results:
  - When $I \geq 3$, an SCF is pure Nash implementable by a direct mechanism if and only if it satisfies Maskin monotonicity.
  - When $I \geq 2$, an SCF is mixed Nash implementable by a finite mechanism if and only if it satisfies Maskin monotonicity.

- Both implementation results are exact and robust.

Model

- $\mathcal{I} = \{1, 2, ..., I\}$: finite set of players

Model

- $\mathcal{I} = \{1, 2, ..., I\}$: finite set of players
- $X \equiv \Delta(A) \times \mathbb{R}^I$: set of allocations

Model

- $\mathcal{I} = \{1, 2, ..., I\}$: finite set of players
- $X \equiv \Delta(A) \times \mathbb{R}^I$: set of allocations
- Θ_i: finite set of types

Model

- $\mathcal{I} = \{1, 2, ..., I\}$: finite set of players
- $X \equiv \Delta(A) \times \mathbb{R}^I$: set of allocations
- Θ_i: finite set of types
- Each θ_i induces a quasilinear EU $u_i(\cdot, \theta_i) : X \to \mathbb{R}$

Model

- $\mathcal{I} = \{1, 2, ..., I\}$: finite set of players
- $X \equiv \Delta(A) \times \mathbb{R}^I$: set of allocations
- Θ_i: finite set of types
- Each θ_i induces a quasilinear EU $u_i(\cdot, \theta_i) : X \to \mathbb{R}$
- $u_i(\cdot, \mathbf{0}, \theta_i)$ is bounded

Model Cont.

- $\theta \in \Theta \subseteq \times_{i \in \mathcal{I}} \Theta_i$ is called a state

Model Cont.

- $\theta \in \Theta \subseteq \times_{i \in \mathcal{I}} \Theta_i$ is called a state

- $f : \Theta \to X$: a social choice function

Model Cont.

- $\theta \in \Theta \subseteq \times_{i \in \mathcal{I}} \Theta_i$ is called a state
- $f : \Theta \to X$: a social choice function
- θ is common knowledge among the agents but unknown to the designer

Model Cont.

- $\theta \in \Theta \subseteq \times_{i \in \mathcal{I}} \Theta_i$ is called a state
- $f : \Theta \to X$: a social choice function
- θ is common knowledge among the agents but unknown to the designer
- **Assumption**: Any two types θ_i and θ_i' induce distinct preference orderings over $\Delta(A)$.

Model Cont.

- $\theta \in \Theta \subseteq \times_{i \in \mathcal{I}} \Theta_i$ is called a state

- $f : \Theta \to X$: a social choice function

- θ is common knowledge among the agents but unknown to the designer

- **Assumption**: Any two types θ_i and θ_i' induce distinct preference orderings over $\Delta(A)$.
  - There is a menu of dictator lotteries $l_k^* : \Theta_k \to \Delta(A)$ such that

$$u_k\left(l_k^*\left(\theta_k\right), \theta_k\right) > u_k\left(l_k^*\left(\theta_k'\right), \theta_k\right) \text{ whenever } \theta_k' \neq \theta_k.$$

(Maskin) Monotonicity

- Define the strict lower-counter set of allocation x for type θ_i as

$$\mathcal{SL}_i(x, \theta_i) = \left\{x' \in X : u_i(x, \theta_i) > u_i\left(x', \theta_i\right)\right\}.$$

(Maskin) Monotonicity

- Define the strict lower-counter set of allocation x for type θ_i as

$$\mathcal{SL}_i(x, \theta_i) = \left\{x' \in X : u_i(x, \theta_i) > u_i\left(x', \theta_i\right)\right\}.$$

- Define the strict upper-contour set of allocation x for type θ_i as,

$$\mathcal{SU}_i(x, \theta_i) = \left\{x' \in X : u_i\left(x', \theta_i\right) > u_i(x, \theta_i)\right\}.$$

(Maskin) Monotonicity

- Define the strict lower-counter set of allocation x for type θ_i as

$$\mathcal{SL}_i(x, \theta_i) = \left\{x' \in X : u_i(x, \theta_i) > u_i\left(x', \theta_i\right)\right\}.$$

- Define the strict upper-contour set of allocation x for type θ_i as,

$$\mathcal{SU}_i(x, \theta_i) = \left\{x' \in X : u_i\left(x', \theta_i\right) > u_i(x, \theta_i)\right\}.$$

- Say an SCF f satisfies (Maskin-)monotonicity if

$$f(\theta) \neq f\left(\theta'\right) \Rightarrow \exists \text{ agent } i \text{ s.t. } \mathcal{SL}_i(f(\theta), \theta_i) \cap \mathcal{SU}_i\left(f(\theta), \theta_i'\right) \neq \varnothing.$$

Best Challenge Scheme

- Whenever $\mathcal{SL}_i(f(\tilde{\theta}), \tilde{\theta}_i) \cap \mathcal{SU}_i(f(\tilde{\theta}), \theta_i) \neq \varnothing$, select a test allocation

$$x(\tilde{\theta}, \theta_i) \in \mathcal{SL}_i(f(\tilde{\theta}), \tilde{\theta}_i) \cap \mathcal{SU}_i(f(\tilde{\theta}), \theta_i).$$

Best Challenge Scheme

- Whenever $\mathcal{SL}_i(f(\tilde{\theta}), \tilde{\theta}_i) \cap \mathcal{SU}_i(f(\tilde{\theta}), \theta_i) \neq \varnothing$, select a test allocation

$$x(\tilde{\theta}, \theta_i) \in \mathcal{SL}_i(f(\tilde{\theta}), \tilde{\theta}_i) \cap \mathcal{SU}_i(f(\tilde{\theta}), \theta_i).$$

- The best challenge scheme for type θ_i against state $\tilde{\theta} \in \Theta$ is defined as

$$B_{\theta_i}(\tilde{\theta}) = \begin{cases} f(\tilde{\theta}), & \text{if } \mathcal{SL}_i(f(\tilde{\theta}), \tilde{\theta}_i) \cap \mathcal{SU}_i(f(\tilde{\theta}), \theta_i) = \varnothing; \\ x(\tilde{\theta}, \theta_i), & \text{if } \mathcal{SL}_i(f(\tilde{\theta}), \tilde{\theta}_i) \cap \mathcal{SU}_i(f(\tilde{\theta}), \theta_i) \neq \varnothing. \end{cases}$$

Mechanism

- A mechanism is a triplet $((M_i), g, (\tau_i))_{i \in I}$ where M_i is the message space; $g : M \longrightarrow X$ is an outcome function; and $\tau_i : M \longrightarrow \mathbb{R}$ is a transfer rule.

Mechanism

- A mechanism is a triplet $((M_i), g, (\tau_i))_{i \in I}$ where M_i is the message space; $g : M \longrightarrow X$ is an outcome function; and $\tau_i : M \longrightarrow \mathbb{R}$ is a transfer rule.

- Goal: find a mechanism $\left((M_i, \tau_i)_{i \in \mathcal{I}}, g\right)$ such that at each θ,

Mechanism

- A mechanism is a triplet $((M_i), g, (\tau_i))_{i \in I}$ where M_i is the message space; $g : M \longrightarrow X$ is an outcome function; and $\tau_i : M \longrightarrow \mathbb{R}$ is a transfer rule.

- Goal: find a mechanism $\left((M_i, \tau_i)_{i \in \mathcal{I}}, g\right)$ such that at each θ,
  - A pure NE exists;

Mechanism

- A mechanism is a triplet $((M_i), g, (\tau_i))_{i\in I}$ where M_i is the message space; $g : M \longrightarrow X$ is an outcome function; and $\tau_i : M \longrightarrow \mathbb{R}$ is a transfer rule.

- Goal: find a mechanism $\left((M_i, \tau_i)_{i\in\mathcal{I}}, g\right)$ such that at each θ,
  - A pure NE exists;
  - For any mixed NE $\sigma \in \times_{i\in\mathcal{I}}\Delta(M_i)$, we have

$$\sigma(m) > 0 \Rightarrow g(m) = f(\theta) \text{ and } \tau_i(m) = 0 \text{ for every } i.$$

Pure NE Implementation in DRM

- **Rule 1.** If $I(\geq 3)$ agents all report $\tilde{\theta}$, then implement $f(\tilde{\theta})$;

Pure NE Implementation in DRM

- **Rule 1.** If $I(\geq 3)$ agents all report $\tilde{\theta}$, then implement $f(\tilde{\theta})$;

- **Rule 2.** If $I-1$ agents report $\tilde{\theta}$ and agent i reports $\theta \neq \tilde{\theta}$, then implement $B_{\theta_i}(\tilde{\theta})$. Moreover, agent $i+1$ has to pay a large penalty of $2D$.

Pure NE Implementation in DRM

- **Rule 1.** If $I (\geq 3)$ agents all report $\tilde{\theta}$, then implement $f(\tilde{\theta})$;
- **Rule 2.** If $I - 1$ agents report $\tilde{\theta}$ and agent i reports $\theta \neq \tilde{\theta}$, then implement $B_{\theta_i}(\tilde{\theta})$. Moreover, agent $i + 1$ has to pay a large penalty of $2D$.
- **Rule 3.** Otherwise, implement $f(m_1)$. Moreover, any agent i who does not report a state in the unique majority is asked to pay D.

Mixed NE Implementation

- Consider $I = 2$. Each agent reports

$$m_i = \left(m_i^1, \left(m_{i,i}^2, m_{i,j}^2\right), m_i^3\right) \in \Theta_i \times \Theta \times \Theta_i.$$

Mixed NE Implementation

- Consider $I = 2$. Each agent reports

$$m_i = \left(m_i^1, \left(m_{i,i}^2, m_{i,j}^2\right), m_i^3\right) \in \Theta_i \times \Theta \times \Theta_i.$$

- The outcome is either [1 checks 2] or [2 checks 1] with equal probability.

Mixed NE Implementation

- Consider $I = 2$. Each agent reports

$$m_i = \left(m_i^1, \left(m_{i,i}^2, m_{i,j}^2\right), m_i^3\right) \in \Theta_i \times \Theta \times \Theta_i.$$

- The outcome is either [1 checks 2] or [2 checks 1] with equal probability.

- Two key notions:

Mixed NE Implementation

- Consider $I = 2$. Each agent reports

$$m_i = \left(m_i^1, \left(m_{i,i}^2, m_{i,j}^2\right), m_i^3\right) \in \Theta_i \times \Theta \times \Theta_i.$$

- The outcome is either [1 checks 2] or [2 checks 1] with equal probability.

- Two key notions:
    - consistency: $m_i^2 = m_j^2$.

Mixed NE Implementation

- Consider $I = 2$. Each agent reports

$$m_i = \left(m_i^1, \left(m_{i,i}^2, m_{i,j}^2\right), m_i^3\right) \in \Theta_i \times \Theta \times \Theta_i.$$

- The outcome is either [1 checks 2] or [2 checks 1] with equal probability.

- Two key notions:
  - consistency: $m_i^2 = m_j^2$.
  - no challenge: $B_{m_i^3}\left(m_j^2\right) = f\left(m_j^2\right)$.

i Checks j: Outcome Function

- If it is consistent and no challenge, then implement $f(m_j^2)$.

i Checks j: Outcome Function

- If it is consistent and no challenge, then implement $f\left(m_j^2\right)$.

- If there is either inconsistency or challenge, then implement

$\frac{1}{2}\left(l_i^*\left(m_i^1\right)+l_j^*\left(m_j^1\right)\right)$	with probability ε
$B_{m_i^3}\left(m_j^2\right)$	with probability $1-\varepsilon$

where l_k^* is the dictator lotteries constructed earlier.

i Checks j: Outcome Function

- If it is consistent and no challenge, then implement $f\left(m_j^2\right)$.
- If there is either inconsistency or challenge, then implement

$\frac{1}{2}\left(l_i^*\left(m_i^1\right)+l_j^*\left(m_j^1\right)\right)$	with probability ε
$B_{m_i^3}\left(m_j^2\right)$	with probability $1-\varepsilon$

where l_k^* is the dictator lotteries constructed earlier.

- Choose ε small so that all test allocations remain valid.

i Checks j: Transfer Rule

Choose D large so that transfers dominate:

Transfer to agents	$m^2_{i,j} = m^2_{j,j}$	$m^2_{i,j} \neq m^2_{j,j}$	
		$m^2_{i,j} = m^1_j$	$m^2_{i,j} \neq m^1_j$
$(\tau_i(m), \tau_j(m))$	$(0,0)$	$(D,-D)$	$(-D,-D)$

- j's 1st report is truthful$\Rightarrow$ i's 2nd report is truthful:

Transfer to i	$\left(m^1_j, m^2_{j,j}\right) = (\theta_j, \theta_j)$	$\left(m^1_j, m^2_{j,j}\right) = (\theta_j, \theta'_j)$
$m^2_{i,j} = \theta_j$	0	D
$m^2_{i,j} \neq \theta_j$	$-D$	0 or $-D$

Contagion of Truth

	1st report	2nd report
Agent 1	θ_1	β_1, β_2
Agent 2	α_2	γ_1, γ_2

Contagion of Truth

	1st report	2nd report
Agent 1	θ_1	β_1, β_2
Agent 2	α_2	γ_1, γ_2

	1st report	2nd report
Agent 1	θ_1	β_1, β_2
Agent 2	α_2	θ_1, γ_2

Contagion of Truth

	1st report	2nd report
Agent 1	θ_1	β_1, β_2
Agent 2	α_2	γ_1, γ_2

	1st report	2nd report
Agent 1	θ_1	β_1, β_2
Agent 2	α_2	θ_1, γ_2

	1st report	2nd report
Agent 1	θ_1	θ_1, β_2
Agent 2	α_2	θ_1, γ_2

Contagion of Truth

	1st report	2nd report
Agent 1	θ_1	β_1, β_2
Agent 2	α_2	γ_1, γ_2

	1st report	2nd report
Agent 1	θ_1	β_1, β_2
Agent 2	α_2	θ_1, γ_2

	1st report	2nd report
Agent 1	θ_1	θ_1, β_2
Agent 2	α_2	θ_1, γ_2

- Truth-telling all the way constitutes a pure-strategy NE.

Summary

- To sum up,

Summary

- To sum up,
  - m_i^1 controls (only) dictator lotteries l_i^*;

Summary

- To sum up,
  - m_i^1 controls (only) dictator lotteries l_i^*;
  - m_i^2 controls consistency and transfers;

Summary

- To sum up,
  - m_i^1 controls (only) dictator lotteries l_i^*;
  - m_i^2 controls consistency and transfers;
  - m_i^3 controls whether to challenge m_j^2.

Summary

- To sum up,
  - m_i^1 controls (only) dictator lotteries l_i^*;
  - m_i^2 controls consistency and transfers;
  - m_i^3 controls whether to challenge m_j^2.
- We show that in any equilibrium, we have

Summary

- To sum up,
  - m_i^1 controls (only) dictator lotteries l_i^*;
  - m_i^2 controls consistency and transfers;
  - m_i^3 controls whether to challenge m_j^2.
- We show that in any equilibrium, we have
  - consistency ($\Rightarrow \tau_i(m) = 0$);

Summary

- To sum up,
  - m_i^1 controls (only) dictator lotteries l_i^*;
  - m_i^2 controls consistency and transfers;
  - m_i^3 controls whether to challenge m_j^2.

- We show that in any equilibrium, we have
  - consistency ($\Rightarrow \tau_i(m) = 0$);
  - no challenge ($\Rightarrow g(m) = f(\theta)$).

Consistency

- For instance, it cannot be both agents randomize their 2nd reports.

Consistency

- For instance, it cannot be both agents randomize their 2nd reports.
  - If so, inconsistency (and dictator lotteries) occurs with σ_j-positive probability for every m_i. Hence, for some $(\beta_1, \beta_2) \neq (\gamma_1, \gamma_2)$,

	1st report	2nd report
Agent 1	θ_1	β_1, β_2
Agent 2	θ_2	γ_1, γ_2

Consistency

- For instance, it cannot be both agents randomize their 2nd reports.
  - If so, inconsistency (and dictator lotteries) occurs with σ_j-positive probability for every m_i. Hence, for some $(\beta_1, \beta_2) \neq (\gamma_1, \gamma_2)$,

	1st report	2nd report
Agent 1	θ_1	β_1, β_2
Agent 2	θ_2	γ_1, γ_2

  - This contradicts contagion of truth.

Consistency

- For instance, it cannot be both agents randomize their 2nd reports.
  - If so, inconsistency (and dictator lotteries) occurs with σ_j-positive probability for every m_i. Hence, for some $(\beta_1, \beta_2) \neq (\gamma_1, \gamma_2)$,

	1st report	2nd report
Agent 1	θ_1	β_1, β_2
Agent 2	θ_2	γ_1, γ_2

  - This contradicts contagion of truth.

- Similarly, it cannot be both agents choose a deterministic yet inconsistent 2nd report.

Consistency

- For instance, it cannot be both agents randomize their 2nd reports.
  - If so, inconsistency (and dictator lotteries) occurs with σ_j-positive probability for every m_i. Hence, for some $(\beta_1, \beta_2) \neq (\gamma_1, \gamma_2)$,

	1st report	2nd report
Agent 1	θ_1	β_1, β_2
Agent 2	θ_2	γ_1, γ_2

  - This contradicts contagion of truth.

- Similarly, it cannot be both agents choose a deterministic yet inconsistent 2nd report.
- Similarly, it cannot be only one agent who randomizes his 2nd report.

No Challenge

- By consistency, there can only be a unanimous 2nd report

	2st report	3nd report
Agent 1	$\tilde{\theta}_1, \tilde{\theta}_2$	δ_1
Agent 2	$\tilde{\theta}_1, \tilde{\theta}_2$	δ_2

No Challenge

- By consistency, there can only be a unanimous 2nd report

	2st report	3nd report
Agent 1	$\tilde{\theta}_1, \tilde{\theta}_2$	δ_1
Agent 2	$\tilde{\theta}_1, \tilde{\theta}_2$	δ_2

- We argue that

$$\mathcal{SL}_i(f(\tilde{\theta}), \tilde{\theta}_i) \cap \mathcal{SU}_i(f(\tilde{\theta}), \theta_i) = \varnothing \text{ for every } i$$

which implies

$$\mathcal{SL}_i(f(\tilde{\theta}), \tilde{\theta}_i) \cap \mathcal{U}_i(f(\tilde{\theta}), \theta_i) = \varnothing \text{ for every } i$$

which further implies no challenge.

No Challenge

- Suppose to the contrary that

$$\mathcal{SL}_1(f(\tilde{\theta}), \tilde{\theta}_1) \cap \mathcal{SU}_1(f(\tilde{\theta}), \theta_1) \neq \varnothing$$

No Challenge

- Suppose to the contrary that

$$\mathcal{SL}_1(f(\tilde{\theta}), \tilde{\theta}_1) \cap \mathcal{SU}_1(f(\tilde{\theta}), \theta_1) \neq \varnothing$$

- When 1 checks 2, $\tilde{\theta}$ will also be challenged by every 3rd report δ_1,

$$\mathcal{SL}_1(f(\tilde{\theta}), \tilde{\theta}_1) \cap \mathcal{SU}_1(f(\tilde{\theta}), \delta_1) \neq \varnothing$$

No Challenge

- Suppose to the contrary that

$$\mathcal{SL}_1(f(\tilde{\theta}), \tilde{\theta}_1) \cap \mathcal{SU}_1(f(\tilde{\theta}), \theta_1) \neq \varnothing$$

- When 1 checks 2, $\tilde{\theta}$ will also be challenged by every 3rd report δ_1,

$$\mathcal{SL}_1(f(\tilde{\theta}), \tilde{\theta}_1) \cap \mathcal{SU}_1(f(\tilde{\theta}), \delta_1) \neq \varnothing$$

- Dictator lottery happens with probability 1. Then, contagion of truth implies

$$\tilde{\theta} = \theta.$$

Extension

- Counterexample which shows we can't implement in DRM even in pure NE with $I = 2$.

Extension

- Counterexample which shows we can't implement in DRM even in pure NE with $I = 2$.
- Rationalizable implementation of any monotonic* SCF (Bergemann, Morris, Tercieux (2011));

Extension

- Counterexample which shows we can't implement in DRM even in pure NE with $I = 2$.
- Rationalizable implementation of any monotonic* SCF (Bergemann, Morris, Tercieux (2011));
- Social choice correspondences;

Extension

- Counterexample which shows we can't implement in DRM even in pure NE with $I = 2$.
- Rationalizable implementation of any monotonic* SCF (Bergemann, Morris, Tercieux (2011));
- Social choice correspondences;
- Small transfers (Abreu and Matsushima (1994));

Extension

- Counterexample which shows we can't implement in DRM even in pure NE with $I = 2$.
- Rationalizable implementation of any monotonic* SCF (Bergemann, Morris, Tercieux (2011));
- Social choice correspondences;
- Small transfers (Abreu and Matsushima (1994));
- Infinite Θ;

Extension

- Counterexample which shows we can't implement in DRM even in pure NE with $I = 2$.
- Rationalizable implementation of any monotonic* SCF (Bergemann, Morris, Tercieux (2011));
- Social choice correspondences;
- Small transfers (Abreu and Matsushima (1994));
- Infinite Θ;
- Implementation for every cardinalization (Mezzetti and Renou (2012)).