Strong Nash equilibria: complexity and algorithms

Nicola Gatti

November 5 2014

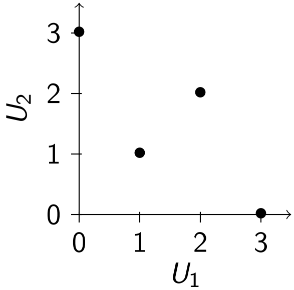
Non–cooperative game

agent 2

		a_3	a_4
agent 1	a_1	$2,2$	$0,3$
	a_2	$3,0$	$1,1$

Non–cooperative game

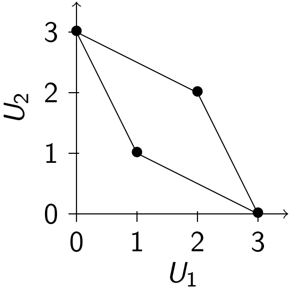
		agent 2	
		a_3	a_4
agent 1	a_1	$2, 2$	$0, 3$
	a_2	$3, 0$	$1, 1$



Non–cooperative game

agent 2

		a_3	a_4
agent 1	a_1	$2, 2$	$0, 3$
	a_2	$3, 0$	$1, 1$

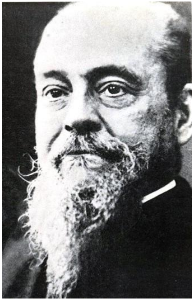


Solution concepts

- Game theory provides different solution concepts
- Each solution concept captures a different game situation
- Examples:
    - Can players communicate before playing and forming coalitions?
    - Can some player commit to a given strategy?
    - Can players correlate their strategies?

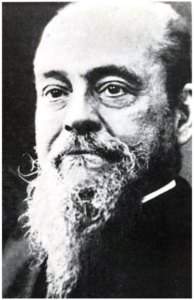
Solution concepts discussed in the talk

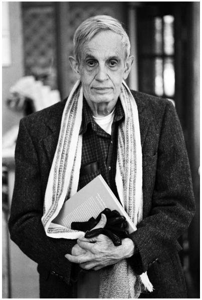
- **Pareto efficiency**: it captures the situation in which players cooperate



Solution concepts discussed in the talk

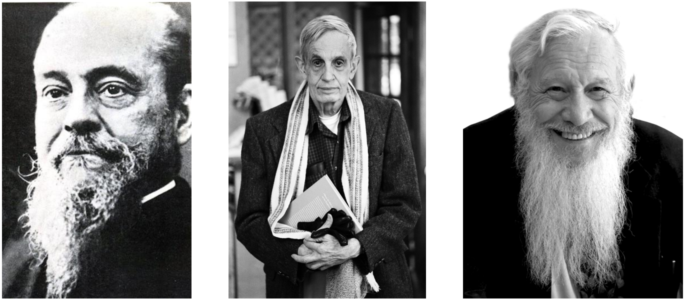
- **Pareto efficiency**: it captures the situation in which players cooperate
- **Nash equilibrium**: it captures the situation in which each player is selfish, doing its best given the strategy of the others





Solution concepts discussed in the talk

- **Pareto efficiency**: it captures the situation in which players cooperate
- **Nash equilibrium**: it captures the situation in which each player is selfish, doing its best given the strategy of the others
- **Strong Nash equilibrium**: it merges Nash equilibrium and Pareto efficiency for each possible coalition



Social efficiency: Pareto

Pareto Efficiency (PE)

A strategy profile is Pareto efficient if there is no other strategy profile in which:

- *all the players gain at least the same utility and*
- *at least one player gains strictly more.*

agent 2

		a_3	a_4
agent 1	a_1	$2, 2$	$0, 3$
	a_2	$3, 0$	$1, 1$

Social efficiency: Pareto

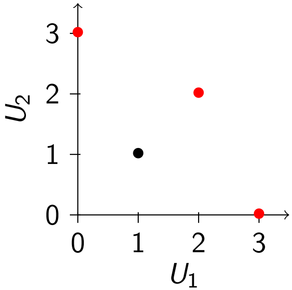
Pareto Efficiency (PE)

A strategy profile is Pareto efficient if there is no other strategy profile in which:

- *all the players gain at least the same utility and*
- *at least one player gains strictly more.*

agent 2

agent 1	a_3	a_4
a_1	$2,2$	$0,3$
a_2	$3,0$	$1,1$



Social efficiency: Pareto

Pareto Efficiency (PE)

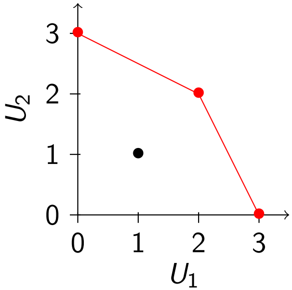
A strategy profile is Pareto efficient if there is no other strategy profile in which:

- *all the players gain at least the same utility and*
- *at least one player gains strictly more.*

agent 2

agent 1

	a_3	a_4
a_1	$2, 2$	$0, 3$
a_2	$3, 0$	$1, 1$



Pareto efficiency: properties

- Every game admits at least one Pareto efficient solution
- Finding a Pareto efficient solution and/or verifying whether a given solution is Pareto efficient could be not easy problems
- The behavior of rational players could be not socially efficient

Individual efficiency: Nash

Nash equilibrium (NE)

A strategy profile is a Nash equilibrium if no player can gain strictly more by unilateral deviations.

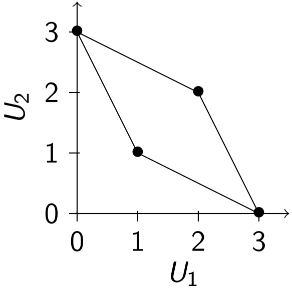
		agent 2	
		a_3	a_4
agent 1	a_1	$2, 2$	$0, 3$
	a_2	$3, 0$	$1, 1$

Individual efficiency: Nash

Nash equilibrium (NE)

A strategy profile is a Nash equilibrium if no player can gain strictly more by unilateral deviations.

		agent 2	
		a_3	a_4
agent 1	a_1	$2, 2$	$0, 3$
	a_2	$3, 0$	$1, 1$

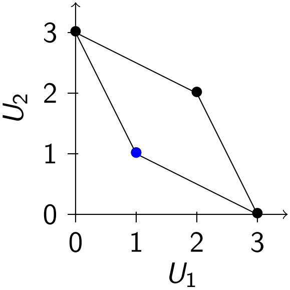


Individual efficiency: Nash

Nash equilibrium (NE)

A strategy profile is a Nash equilibrium if no player can gain strictly more by unilateral deviations.

		agent 2	
		a_3	a_4
agent 1	a_1	$2, 2$	$0, 3$
	a_2	$3, 0$	$1, 1$



Nash equilibrium: properties

- Every finite game admits at least one Nash equilibrium in mixed strategies
- Finding a Nash equilibrium could be not an easy problem
- Nash equilibria may be not Pareto efficient

Selfishness with communication: Strong Nash

Strong Nash equilibrium (SNE)

A strategy profile is a Strong Nash equilibrium if:

- *it is a Nash equilibrium*
- *it is Pareto efficient for each possible sub coalition.*

		agent 2	
		a_3	a_4
agent 1	a_1	$2, 2$	$1, 3$
	a_2	$3, 1$	$0, 0$

Selfishness with communication: Strong Nash

Strong Nash equilibrium (SNE)

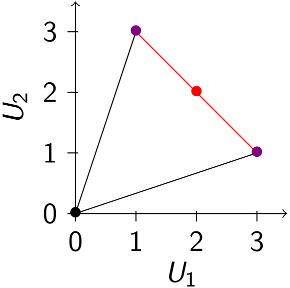
A strategy profile is a Strong Nash equilibrium if:

- *it is a Nash equilibrium*
- *it is Pareto efficient for each possible sub coalition.*

agent 2

agent 1

	a_3	a_4
a_1	$2, 2$	$1, 3$
a_2	$3, 1$	$0, 0$



Strong Nash equilibrium: properties

- A Strong Nash equilibrium may not exist
- The verification problem involves the verification of two separate properties:
    - Pareto efficiency and
    - Nash equilibrium

  and therefore it could be not easy
- The finding problem could not be easy

Strong Nash equilibrium: in practice

- Strong Nash would require that players can communicate and form coalitions

Strong Nash equilibrium: in practice

- Strong Nash would require that players can communicate and form coalitions
- In many scenarios, communication can be conducted by software agents operating over a network

Strong Nash equilibrium: in practice

- Strong Nash would require that players can communicate and form coalitions
- In many scenarios, communication can be conducted by software agents operating over a network
- Internet is the main example

Strong Nash equilibrium: in practice

- Strong Nash would require that players can communicate and form coalitions
- In many scenarios, communication can be conducted by software agents operating over a network
- Internet is the main example
- In many scenarios, only some sub coalitions are possible (e.g. size–bounded sub coalitions)

Strong Nash equilibrium: in practice

- Strong Nash would require that players can communicate and form coalitions
- In many scenarios, communication can be conducted by software agents operating over a network
- Internet is the main example
- In many scenarios, only some sub coalitions are possible (e.g. size–bounded sub coalitions)
- Graph–action games: games in which each agent is a node of a graph and its utility depends only on the neighbors' actions (*future research*)

Computation: motivation

- In practice, we want to find solutions and therefore we need to answer to the question “What is the complexity of finding a solution?”

Computation: motivation

- In practice, we want to find solutions and therefore we need to answer to the question "What is the complexity of finding a solution?"
- If finding a solution is not achievable in reasonable time, then we need to relax a solution concept (e.g., ϵ–approximate NE)

Computation: motivation

- In practice, we want to find solutions and therefore we need to answer to the question "What is the complexity of finding a solution?"
- If finding a solution is not achievable in reasonable time, then we need to relax a solution concept (e.g., ϵ–approximate NE)
- We resort to computational complexity tools

Computation: introduction

- Each computational problem is characterized by:
  - **An input**
  - **A question**

Computation: introduction

- Each computational problem is characterized by:
  - **An input**
  - **A question**

- We distinguish two main classes for the question:
  - **Decision**: the answer can be YES/NO
  - **Functional/Search**: the answer is a solution

Computation: introduction

- Each computational problem is characterized by:
  - **An input**
  - **A question**

- We distinguish two main classes for the question:
  - **Decision**: the answer can be YES/NO
  - **Functional/Search**: the answer is a solution

- We would like to characterize problems in two main classes:
  - **Polynomial time**: problems admitting a polynomial–time (in the size of the input) algorithm
  - **Exponential time**: problems not admitting any polynomial–time (in the size of the input) algorithm

Computation: complexity classes (decision)

Polynomial ($\mathcal{P}$) There exists a polynomial–time algorithm

Computation: complexity classes (decision)

Polynomial ($\mathcal{P}$) There exists a polynomial–time algorithm

Non–deterministic Polynomial ($\mathcal{NP}$) There exists a polynomial–time algorithm certifying that a solution given in input is a YES solution of the problem

Computation: complexity classes (decision)

Polynomial ($\mathcal{P}$)	There exists a polynomial–time algorithm
Non–deterministic Polynomial ($\mathcal{NP}$)	There exists a polynomial–time algorithm certifying that a solution given in input is a YES solution of the problem
$\mathcal{NP}$–hard	There exists a polynomial–time reduction from every $\mathcal{NP}$ problems

Computation: complexity classes (decision)

Polynomial ($\mathcal{P}$)	There exists a polynomial–time algorithm
Non–deterministic Polynomial ($\mathcal{NP}$)	There exists a polynomial–time algorithm certifying that a solution given in input is a YES solution of the problem
$\mathcal{NP}$–hard	There exists a polynomial–time reduction from every $\mathcal{NP}$ problems
$\mathcal{NP}$–complete	The problem is in $\mathcal{NP}$ and it is $\mathcal{NP}$–hard

Computation: complexity classes (decision)

Polynomial ($\mathcal{P}$) There exists a polynomial–time algorithm

Non–deterministic Polynomial ($\mathcal{NP}$) There exists a polynomial–time algorithm certifying that a solution given in input is a YES solution of the problem

$\mathcal{NP}$–hard There exists a polynomial–time reduction from every $\mathcal{NP}$ problems

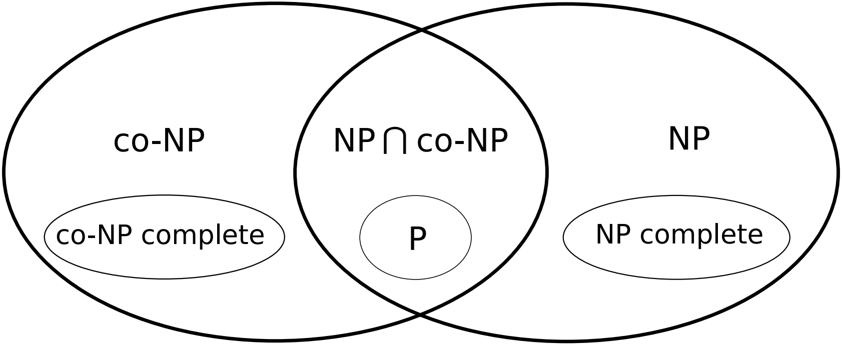
$\mathcal{NP}$–complete The problem is in $\mathcal{NP}$ and it is $\mathcal{NP}$–hard

Complement Non–deterministic Polynomial (co–$\mathcal{NP}$) There exists a polynomial–time algorithm certifying that a solution given in input is a NO solution of the problem

co–$\mathcal{NP}$–hard Analogous to $\mathcal{NP}$–hard

co–$\mathcal{NP}$–complete Analogous to co–$\mathcal{NP}$–hard

Computation: complexity classes (decision)



Decision problems and equilibria

PE existence	$\mathcal{P}$ class, given that there always exists a Pareto efficient solution

Decision problems and equilibria

PE existence $\mathcal{P}$ class, given that there always exists a Pareto efficient solution

NE existence $\mathcal{P}$ class, given that there always exists a Nash equilibrium

Decision problems and equilibria

PE existence	$\mathcal{P}$ class, given that there always exists a Pareto efficient solution
NE existence	$\mathcal{P}$ class, given that there always exists a Nash equilibrium
SNE existence	An SNE may not exist, addressing this task is not trivial; the problem has been proved $\mathcal{NP}$–complete

Decision problems and equilibria

PE existence $\mathcal{P}$ class, given that there always exists a Pareto efficient solution

NE existence $\mathcal{P}$ class, given that there always exists a Nash equilibrium

SNE existence An SNE may not exist, addressing this task is not trivial; the problem has been proved $\mathcal{NP}$–complete

- $\mathcal{NP}$-hardness Proved by Conitzer and Sandholm (GEB, 2008) with 2–player symmetric games (from SAT)
- $\mathcal{NP}$ membership Proved by Gatti, Rocco, Sandholm (AAMAS, 2013)

NP–membership

What do we need? We need an algorithm that, given a strategy profile, certifies in polynomial time that it is an SNE

NP–membership

What do we need? We need an algorithm that, given a strategy profile, certifies in polynomial time that it is an SNE

NE verification Can we certify in polynomial time that a given solution is an NE?

NP–membership

What do we need? We need an algorithm that, given a strategy profile, certifies in polynomial time that it is an SNE

NE verification Can we certify in polynomial time that a given solution is an NE?

Yes, it is trivial

NP–membership

What do we need? We need an algorithm that, given a strategy profile, certifies in polynomial time that it is an SNE

NE verification Can we certify in polynomial time that a given solution is an NE?

Yes, it is trivial

PE verification Can we certify in polynomial time that a given solution is an NE?

NP–membership

What do we need? We need an algorithm that, given a strategy profile, certifies in polynomial time that it is an SNE

NE verification Can we certify in polynomial time that a given solution is an NE?

Yes, it is trivial

PE verification Can we certify in polynomial time that a given solution is an NE?

Yes, but the proof is not trivial

Nash equilibrium verification

		agent 2				
		a_6	a_7	a_8	a_9	a_10
agent 1	a_1	$2, 2$	$1, 3$	$3, 1$	$5, 2$	$3, 3$
	a_2	$3, 1$	$3, 0$	$0, 0$	$4, 0$	$4, 0$
	a_3	$1, 2$	$1, 1$	$2, 2$	$0, 1$	$0, 3$
	a_4	$3, 3$	$0, 2$	$0, 3$	$0, 4$	$1, 5$
	a_5	$4, 1$	$5, 0$	$1, 1$	$4, 0$	$3, 0$

Nash equilibrium verification

agent 2

		a_6	a_7	a_8	a_9	a_10
agent 1	a_1	2, 2	1, 3	3, 1	5, 2	3, 3
	a_2	3, 1	3, 0	0, 0	4, 0	4, 0
	a_3	1, 2	1, 1	2, 2	0, 1	0, 3
	a_4	3, 3	0, 2	0, 3	0, 4	1, 5
	a_5	4, 1	5, 0	1, 1	4, 0	3, 0

Nash equilibrium verification

agent 2

agent 1

	a_6	a_7	a_8	a_9	a_10
a_1	$2, 2$	$1, 3$	$3, 1$	$5, 2$	$3, 3$
a_2	$3, 1$	$3, 0$	$0, 0$	$4, 0$	$4, 0$
a_3	$1, 2$	$1, 1$	$2, 2$	$0, 1$	$0, 3$
a_4	$3, 3$	$0, 2$	$0, 3$	$0, 4$	$1, 5$
a_5	$4, 1$	$5, 0$	$1, 1$	$4, 0$	$3, 0$

Nash equilibrium verification

agent 2

agent 1

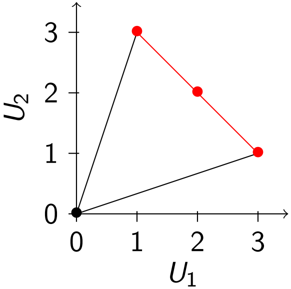
	a_6	a_7	a_8	a_9	$a_1 0$
a_1	$2, 2$	$1, 3$	$3, 1$	$5, 2$	$3, 3$
a_2	$3, 1$	$3, 0$	$0, 0$	$4, 0$	$4, 0$
a_3	$1, 2$	$1, 1$	$2, 2$	$0, 1$	$0, 3$
a_4	$3, 3$	$0, 2$	$0, 3$	$0, 4$	$1, 5$
a_5	$4, 1$	$5, 0$	$1, 1$	$4, 0$	$3, 0$

Pareto efficiency verification

agent 2

agent 1

	a_3	a_4
a_1	$2, 2$	$1, 3$
a_2	$3, 1$	$0, 0$

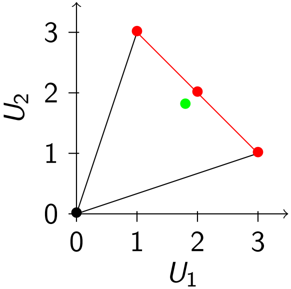


Pareto efficiency verification

agent 2

agent 1

	a_3	a_4
a_1	$2, 2$	$1, 3$
a_2	$3, 1$	$0, 0$



Pareto efficiency verification

Given in input $(\overline{x}_1, \overline{x}_2)$, we need to solve a feasibility problem

$$\mathbf{x}_1^t U_1 \mathbf{x}_2 - \overline{\mathbf{x}}_1^t U_1 \overline{\mathbf{x}}_2 > 0 \quad (1)$$

$$\mathbf{x}_1^t U_2 \mathbf{x}_2 - \overline{\mathbf{x}}_1^t U_2 \overline{\mathbf{x}}_2 > 0 \quad (2)$$

$$\mathbf{1}^t \cdot \mathbf{x}_1 = 1 \quad (3)$$

$$\mathbf{1}^t \cdot \mathbf{x}_2 = 1 \quad (4)$$

$$\mathbf{x}_1 > \mathbf{0} \quad (5)$$

$$\mathbf{x}_2 > \mathbf{0} \quad (6)$$

Pareto efficiency verification

Given in input $(\overline{x}_1, \overline{x}_2)$, we need to solve a feasibility problem

$$\mathbf{x}_1^t U_1 \mathbf{x}_2 - \overline{\mathbf{x}}_1^t U_1 \overline{\mathbf{x}}_2 > 0 \tag{1}$$

$$\mathbf{x}_1^t U_2 \mathbf{x}_2 - \overline{\mathbf{x}}_1^t U_2 \overline{\mathbf{x}}_2 > 0 \tag{2}$$

$$\mathbf{1}^t \cdot \mathbf{x}_1 = 1 \tag{3}$$

$$\mathbf{1}^t \cdot \mathbf{x}_2 = 1 \tag{4}$$

$$\mathbf{x}_1 > \mathbf{0} \tag{5}$$

$$\mathbf{x}_2 > \mathbf{0} \tag{6}$$

The problem (1)–(6) is:

- Non–linear
- Non–convex

Result

Theorem (PE verification is easy)

There is a polynomial–time algorithm verifying whether a given strategy profile is Pareto efficient when the number of players is fixed.

Result

Theorem (PE verification is easy)

There is a polynomial–time algorithm verifying whether a given strategy profile is Pareto efficient when the number of players is fixed.

Corollary

The problem of deciding whether a game admits an SNE is in $\mathcal{NP}$ *when the number of players is fixed.*

Result

Theorem (PE verification is easy)

There is a polynomial–time algorithm verifying whether a given strategy profile is Pareto efficient when the number of players is fixed.

Corollary

The problem of deciding whether a game admits an SNE is in $\mathcal{NP}$ *when the number of players is fixed.*

Corollary (PE finding is easy)

There is a polynomial–time algorithm returning a (generic) Pareto efficient solution when the number of players is fixed.

Proof sketch [PE verification is easy]

Step 1 We can formulate the problem as a minmax problem with

  - n minimizing players with m actions against

Proof sketch [PE verification is easy]

Step 1 We can formulate the problem as a minmax problem with

- n minimizing players with m actions against
- 1 maximizing player with n actions

Step 2 There exists at least one solution in which each minimizing player randomizes over at most n actions (*by simple linear programming considerations*)

Proof sketch [PE verification is easy]

Step 1 We can formulate the problem as a minmax problem with

- n minimizing players with m actions against
- 1 maximizing player with n actions

Step 2 There exists at least one solution in which each minimizing player randomizes over at most n actions (*by simple linear programming considerations*)

Step 3 The minmax problem in which the minimizing players use n actions can be solved in time $L^{O(1)} \cdot (n+1)^{O(n^2)}$ (*by means of real algebraic geometry algorithms*)

Proof sketch [PE verification is easy]

Step 1 We can formulate the problem as a minmax problem with

- n minimizing players with m actions against
- 1 maximizing player with n actions

Step 2 There exists at least one solution in which each minimizing player randomizes over at most n actions (*by simple linear programming considerations*)

Step 3 The minmax problem in which the minimizing players use n actions can be solved in time $L^{O(1)} \cdot (n+1)^{O(n^2)}$ (*by means of real algebraic geometry algorithms*)

Step 4 The optimal solution can be found by solving the minmax problem for $O(m^{n^2})$ supports

Summary The total complexity is $L^{O(1)} \cdot (n+1)^{O(n^2)} \cdot O(m^{n^2})$

Example

agent 2

		a_6	a_7	a_8	a_9	a_10
agent 1	a_1	2, 2	1, 3	3, 1	5, 2	3, 3
	a_2	3, 1	3, 0	0, 0	4, 0	4, 0
	a_3	1, 2	1, 1	2, 2	0, 1	0, 3
	a_4	3, 3	0, 2	0, 3	0, 4	1, 5
	a_5	4, 1	5, 0	1, 1	4, 0	3, 0

Example

		agent 2				
		a_6	a_7	a_8	a_9	$a_1$0
agent 1	a_1	$2, 2$	$1, 3$	$3, 1$	$5, 2$	$3, 3$
	a_2	$3, 1$	$3, 0$	$0, 0$	$4, 0$	$4, 0$
	a_3	$1, 2$	$1, 1$	$2, 2$	$0, 1$	$0, 3$
	a_4	$3, 3$	$0, 2$	$0, 3$	$0, 4$	$1, 5$
	a_5	$4, 1$	$5, 0$	$1, 1$	$4, 0$	$3, 0$

Example

agent 2

		a_6	a_7	a_8	a_9	a_10
agent 1	a_1	2, 2	1, 3	3, 1	5, 2	3, 3
	a_2	3, 1	3, 0	0, 0	4, 0	4, 0
	a_3	1, 2	1, 1	2, 2	0, 1	0, 3
	a_4	3, 3	0, 2	0, 3	0, 4	1, 5
	a_5	4, 1	5, 0	1, 1	4, 0	3, 0

Theory vs. practice (1)

Real algebraic geometry algorithms

- Mathematica
- Matlab
- Maple (SyNRAC)
- QEPCAD

Theory vs. practice (1)

Real algebraic geometry algorithms

- Mathematica
- Matlab
- Maple (SyNRAC)
- QEPCAD

More than 3 hours for solving a single $3 \times 3 \times 3$ problem!

Theory vs. practice (2)

Global optimization solvers (tolerance = 10^{-14})

- SCIP
- BARON

Size (polymatrix games)	Average time	
	SCIP	BARON
2 players, 2 actions per player	< 1 s	< 1 s
3 players, 3 actions per player	~ 20 s	< 1 s
4 players, 4 actions per player	> 1 h	< 1 s
5 players, 5 actions per player	> 1 h	< 1 s
6 players, 6 actions per player	> 1 h	< 1 s
7 players, 7 actions per player	> 1 h	~ 2 s
8 players, 8 actions per player	> 1 h	~ 3 s
9 players, 9 actions per player	> 1 h	~ 4 s
10 players, 10 actions per player	> 1 h	~ 6 s

Theory vs. practice (3)

BARON — whole problem — (average time)					
Players	10 actions	20 actions	30 actions	40 actions	50 actions
2	0.04 s	0.15 s	0.45 s	1.13 s	1.31 s
3	0.09 s	0.76 s	1.73 s	2.02 s	2.74 s
4	0.34 s	2.22 s	2.90 s	3.85 s	5.72 s
5	1.02 s	3.17 s	3.88 s	6.35 s	10.40 s
6	1.85 s	3.86 s	6.03 s	10.13 s	18.41 s
7	3.62 s	5.10 s	8.34 s	16.03 s	30.74 s
8	4.34 s	6.13 s	11.19 s	24.27 s	48.33 s
9	4.96 s	8.27 s	15.84 s	35.26 s	72.43 s
10	5.65 s	9.27 s	20.99 s	49.19 s	106.62 s

SNE verification

Step 1 We verify whether $\bar{\mathbf{x}}$ is a NE

Step 2 For each sub coalition

Step 3 We check whether the strategy of the sub coalition is PE given the strategies of the other players

SNE verification

Step 1 We verify whether $\bar{\mathbf{x}}$ is a NE

Step 2 For each sub coalition

Step 3 We check whether the strategy of the sub coalition is PE given the strategies of the other players

The number of sub coalitions are $2^n - 1$

SNE verification

Step 1 We verify whether $\bar{\mathbf{x}}$ is a NE

Step 2 For each sub coalition

Step 3 We check whether the strategy of the sub coalition is PE given the strategies of the other players

The number of sub coalitions are $2^n - 1$

SNE verification can be solved in polynomial time in the number of actions available to the players

SNE finding

Corollary

There exists a polynomial–time algorithm finding (if exists) a pure–strategy SNE.

Step 1 Scan all the pure strategy profile

Step 2 Check whether it is an SNE

Considerations

Question Why is the decision problem $\mathcal{NP}$–complete?

Considerations

Question	Why is the decision problem $\mathcal{NP}$–complete?
Answer	Because there can exist mixed–strategy SNEs with arbitrarily large supports.

Considerations

agent 2

		a_4	a_5	a_5
agent 1	a_1	$2, 2$	$0, 3$	$0, 3$
	a_2	$3, 0$	$1, 1$	$0, 3$
	a_3	$3, 0$	$1, 1$	$0, 3$

SNEs and mixed strategies

Question How many games admit mixed–strategy SNEs?

SNEs and mixed strategies

Question	How many games admit mixed–strategy SNEs?
Answer	The set is negligible.

SNEs and mixed strategies

Question How many games admit mixed–strategy SNEs?

Answer The set is negligible.

Question What is the probability that a game randomly drawn with uniform probability from the set of all the games admits a mixed–strategy SNE?

SNEs and mixed strategies

Question How many games admit mixed–strategy SNEs?

Answer The set is negligible.

Question What is the probability that a game randomly drawn with uniform probability from the set of all the games admits a mixed–strategy SNE?

Answer Zero.

SNEs and mixed strategies

Question	How many games admit mixed–strategy SNEs?
Answer	The set is negligible.
Question	What is the probability that a game randomly drawn with uniform probability from the set of all the games admits a mixed–strategy SNE?
Answer	Zero.

Therefore, with probability 1, a game either admits a pure–strategy SNE or does not admit any SNE

SNEs and perturbations

agent 2

		a_3	a_4
agent 1	a_1	$2, 0$	$0, 2$
	a_2	$0, 2$	$2, 0$

SNEs and perturbations

		agent 2	
		a_3	a_4
agent 1	a_1	$2, 0$	$0, 2$
	a_2	$0, 2$	$2, 0$

		agent 2	
		a_3	a_4
agent 1	a_1	$2+\epsilon_1, 0+\epsilon_2$	$0+\epsilon_5, 2+\epsilon_6$
	a_2	$0+\epsilon_3, 2+\epsilon_4$	$2+\epsilon_7, 0+\epsilon_8$

SNEs and perturbations

		agent 2	
		a_3	a_4
agent 1	a_1	$2, 0$	$0, 2$
	a_2	$0, 2$	$2, 0$

		agent 2	
		a_3	a_4
agent 1	a_1	$2+\epsilon_1, 0+\epsilon_2$	$0+\epsilon_5, 2+\epsilon_6$
	a_2	$0+\epsilon_3, 2+\epsilon_4$	$2+\epsilon_7, 0+\epsilon_8$

If ϵ_i are i.i.d. random variables in $[0, \overline{\epsilon}]$ even with small $\overline{\epsilon}$, the perturbed game admits a mixed–strategy SNE with zero probability.

Smoothed complexity

Definition (Smoothed $\mathcal{P}$)

A problem is in Smoothed $\mathcal{P}$ if there is an algorithm whose expected compute time is polynomial once the problem has been subject to perturbations.

Smoothed complexity

Definition (Smoothed $\mathcal{P}$)

A problem is in Smoothed $\mathcal{P}$ if there is an algorithm whose expected compute time is polynomial once the problem has been subject to perturbations.

Theorem (SNE is in Smoothed $\mathcal{P}$)

The problem of deciding whether a game admits an SNE is in Smoothed $\mathcal{P}$ *when the number of players is fixed.*

Proof sketch

Algorithm

- Step 1 Scan all the pure strategy profiles and check whether they are SNEs
- Step 2 Check a zero–probability condition necessary for mixed–strategy SNEs
- Step 3 If yes, enumerate all the support profiles and check whether they admit SNEs

Considerations

- Engineering problems are characterized by errors in measurements and noisy
- Errors in measurements and noisy can be seen as perturbations
- In practice, engineering problems admit mixed–strategy SNEs with zero probability
- Only pure–strategy SNEs can be present

Conclusions

Pareto efficiency verification	$\mathcal{P}$
Pareto efficiency finding	$\mathcal{FP}$
Strong Nash equilibrium verification	$\mathcal{P}$
Strong Nash equilibrium decision	$\mathcal{NP}$–complete
Strong Nash equilibrium decision	Smoothed $\mathcal{P}$

Topics non–discussed in the talk

- Algorithms for finding SNEs
- Algorithms for enumerating the PE solutions

Future works

- Computational study of Strong Nash in special cases
    - Graph–action games
    - Speeding up pure–strategy SNE finding algorithms

Future works

- Computational study of Strong Nash in special cases
    - Graph–action games
    - Speeding up pure–strategy SNE finding algorithms
- Computational study of other solution concepts
    - Strong Correlated equilibrium
    - Coalition–proof equilibrium