Computing Correlated Equilibrium and Succinct Representation of Games

Branislav Bošanský

Artificial Intelligence Center,
Department of Computer Science,
Faculty of Electrical Engineering,
Czech Technical University in Prague

branislav.bosansky@agents.fel.cvut.cz

March 25, 2019

Correlated Equilibrium

Correlated Equilibrium

Correlated Equilibrium – a probability distribution over pure strategy profiles $p = \Delta(\mathcal{S})$ that recommends each player i to play the best response; $\forall s_i, s_i' \in \mathcal{S}_i$:

$$\sum_{s_{-i} \in \mathcal{S}_{-i}} p(s_i, s_{-i}) u_i(s_i, s_{-i}) \geq \sum_{s_{-i} \in \mathcal{S}_{-i}} p(s_i, s_{-i}) u_i(s_i', s_{-i})$$

Correlated Equilibrium

Correlated Equilibrium – a probability distribution over pure strategy profiles $p = \Delta(\mathcal{S})$ that recommends each player i to play the best response; $\forall s_i, s_i' \in \mathcal{S}_i$:

$$\sum_{s_{-i} \in \mathcal{S}_{-i}} p(s_i, s_{-i}) u_i(s_i, s_{-i}) \geq \sum_{s_{-i} \in \mathcal{S}_{-i}} p(s_i, s_{-i}) u_i(s_i', s_{-i})$$

Coarse Correlated Equilibrium – a probability distribution over pure strategy profiles $p = \Delta(\mathcal{S})$ that **in expectation** recommends each player i to play the best response; $\forall s_i \in \mathcal{S}_i$:

$$\sum_{s' \in \mathcal{S}'} p(s') u_i(s') \geq \sum_{s' \in \mathcal{S}'} p(s') u_i(s_i, s_{-i}')$$

Correlated Equilibrium

Correlated Equilibrium

The solution concept describes situations with a correlation device present in the environment.

Correlated Equilibrium

The solution concept describes situations with a correlation device present in the environment.

Correlated equilibrium is closely related to learning in competitive scenarios.

Correlated Equilibrium

The solution concept describes situations with a correlation device present in the environment.

Correlated equilibrium is closely related to learning in competitive scenarios.

(Coarse) Correlated equilibrium is often a result of a no-regret learning strategy in a game.

Correlated Equilibrium

Correlated Equilibrium

Computing a CE in normal-form games:

Correlated Equilibrium

Computing a CE in normal-form games:

$$\sum_{s_{-i}\in\mathcal{S}_{-i}} p(s_i, s_{-i}) u_i(s_i, s_{-i}) \geq \sum_{s_{-i}\in\mathcal{S}_{-i}} p(s_i, s_{-i}) u_i(s_i', s_{-i}) \quad \forall s_i, s_i' \in \mathcal{S}_i$$

Correlated Equilibrium

Computing a CE in normal-form games:

$$\sum_{s_{-i} \in \mathcal{S}_{-i}} p(s_i, s_{-i}) u_i(s_i, s_{-i}) \geq \sum_{s_{-i} \in \mathcal{S}_{-i}} p(s_i, s_{-i}) u_i(s'_i, s_{-i}) \quad \forall s_i, s'_i \in \mathcal{S}_i$$

Computation in succinct games:

Correlated Equilibrium

Computing a CE in normal-form games:

$$\sum_{s_{-i} \in \mathcal{S}_{-i}} p(s_i, s_{-i}) u_i(s_i, s_{-i}) \geq \sum_{s_{-i} \in \mathcal{S}_{-i}} p(s_i, s_{-i}) u_i(s_i', s_{-i}) \quad \forall s_i, s_i' \in \mathcal{S}_i$$

Computation in succinct games:

Correlated Equilibrium

Computing a CE in normal-form games:

$$\sum_{s_{-i} \in \mathcal{S}_{-i}} p(s_i, s_{-i}) u_i(s_i, s_{-i}) \geq \sum_{s_{-i} \in \mathcal{S}_{-i}} p(s_i, s_{-i}) u_i(s_i', s_{-i}) \quad \forall s_i, s_i' \in \mathcal{S}_i$$

Computation in succinct games:

- polymatrix games

Correlated Equilibrium

Computing a CE in normal-form games:

$$\sum_{s_{-i} \in \mathcal{S}_{-i}} p(s_i, s_{-i}) u_i(s_i, s_{-i}) \geq \sum_{s_{-i} \in \mathcal{S}_{-i}} p(s_i, s_{-i}) u_i(s_i', s_{-i}) \quad \forall s_i, s_i' \in \mathcal{S}_i$$

Computation in succinct games:

- polymatrix games
- congestion games

Correlated Equilibrium

Computing a CE in normal-form games:

$$\sum_{s_{-i} \in \mathcal{S}_{-i}} p(s_i, s_{-i}) u_i(s_i, s_{-i}) \geq \sum_{s_{-i} \in \mathcal{S}_{-i}} p(s_i, s_{-i}) u_i(s_i', s_{-i}) \quad \forall s_i, s_i' \in \mathcal{S}_i$$

Computation in succinct games:

- polymatrix games
- congestion games
- anonymous games

Correlated Equilibrium

Computing a CE in normal-form games:

$$\sum_{s_{-i} \in \mathcal{S}_{-i}} p(s_i, s_{-i}) u_i(s_i, s_{-i}) \geq \sum_{s_{-i} \in \mathcal{S}_{-i}} p(s_i, s_{-i}) u_i(s'_i, s_{-i}) \quad \forall s_i, s'_i \in \mathcal{S}_i$$

Computation in succinct games:

- polymatrix games
- congestion games
- anonymous games
- symmetric games

Correlated Equilibrium

Computing a CE in normal-form games:

$$\sum_{s_{-i} \in \mathcal{S}_{-i}} p(s_i, s_{-i}) u_i(s_i, s_{-i}) \geq \sum_{s_{-i} \in \mathcal{S}_{-i}} p(s_i, s_{-i}) u_i(s'_i, s_{-i}) \quad \forall s_i, s'_i \in \mathcal{S}_i$$

Computation in succinct games:

- polymatrix games
- congestion games
- anonymous games
- symmetric games
- graphical games with a bounded tree-width

Succinct Representations

Succinct Representations

compact representation of the game with $n = |\mathcal{N}|$ players

Succinct Representations

compact representation of the game with $n = |\mathcal{N}|$ players

we want to reduce the input from $|\mathcal{S}|^{|\mathcal{N}|}$ to $|\mathcal{S}|^d$, where $d \ll |\mathcal{N}|$

Succinct Representations

compact representation of the game with $n = |\mathcal{N}|$ players

we want to reduce the input from $|\mathcal{S}|^{|\mathcal{N}|}$ to $|\mathcal{S}|^d$, where $d \ll |\mathcal{N}|$

which succinct representations are we going to talk about:

Succinct Representations

compact representation of the game with $n = |\mathcal{N}|$ players

we want to reduce the input from $|\mathcal{S}|^{|\mathcal{N}|}$ to $|\mathcal{S}|^{d}$, where $d \ll |\mathcal{N}|$

which succinct representations are we going to talk about:

- congestion games (network congestion games, ...)

Succinct Representations

compact representation of the game with $n = |\mathcal{N}|$ players

we want to reduce the input from $|\mathcal{S}|^{|\mathcal{N}|}$ to $|\mathcal{S}|^d$, where $d \ll |\mathcal{N}|$

which succinct representations are we going to talk about:

- congestion games (network congestion games, ...)
- polymatrix games (zero-sum polymatrix games)

Succinct Representations

compact representation of the game with $n = |\mathcal{N}|$ players

we want to reduce the input from $|\mathcal{S}|^{|\mathcal{N}|}$ to $|\mathcal{S}|^{d}$, where $d \ll |\mathcal{N}|$

which succinct representations are we going to talk about:

- congestion games (network congestion games, ...)
- polymatrix games (zero-sum polymatrix games)
- graphical games (action graph games)

Succinct Representations

Definition (Papadimitriou and Roughgarden, 2008)

A *succinct game* $G = (I, T, U)$ is defined, like all computational problems, in terms of a set of efficiently recognizable inputs I, and two polynomial algorithms T and U. For each $z \in I$, $T(z)$ returns a type, that is, an integer $n \geq 2$ (the number of players) and an n-tuple of integers $(t_1, \ldots, t_n)$, each at least 2 (the cardinalities of the strategy sets). If n and the t_p's are polynomially bounded in $|z|$, the game is said to be of *polynomial type*. Given any n-tuple of positive integers $s = (s_1, \ldots, s_n)$, with $s_p \leq t_p$ for all $p \leq n$, $U(z, p, s)$ returns an integer standing for the utility $u_p(s)$. The resulting game is denoted $G(z)$.

Computing Correlated Equilibria in Succinct Games [1]

Computing Correlated Equilibria in Succinct Games [1]

For almost all succinct representations it holds that the problem of finding any correlated equilibrium can be solved in polynomial time.

Computing Correlated Equilibria in Succinct Games [1]

For almost all succinct representations it holds that the problem of finding any correlated equilibrium can be solved in polynomial time.

Consider a general n-player game.

Computing Correlated Equilibria in Succinct Games [1]

For almost all succinct representations it holds that the problem of finding any correlated equilibrium can be solved in polynomial time.

Consider a general n-player game. Let σ_s be the product of distributions over pure strategies for all players for strategy profile s; $\sigma_s = \Pi_i \sigma_i(s_i)$.

Computing Correlated Equilibria in Succinct Games [1]

For almost all succinct representations it holds that the problem of finding any correlated equilibrium can be solved in polynomial time.

Consider a general n-player game. Let σ_s be the product of distributions over pure strategies for all players for strategy profile s; $\sigma_s = \Pi_i \sigma_i(s_i)$.
For a *correlated equilibrium* σ it must hold:

$$\sum_{s_{-i} \in \mathcal{S}_{-i}} \sigma(s_i, s_{-i}) \left(u_i(s_i, s_{-i}) - u_i(s'_i, s_{-i}) \right) \geq 0 \quad \forall i \in \mathcal{N}, \forall s_i, s'_i \in \mathcal{S}_i$$

Computing Correlated Equilibria in Succinct Games [1]

For almost all succinct representations it holds that the problem of finding any correlated equilibrium can be solved in polynomial time.

Consider a general n-player game. Let σ_s be the product of distributions over pure strategies for all players for strategy profile s; $\sigma_s = \Pi_i \sigma_i(s_i)$.
For a *correlated equilibrium* σ it must hold:

$$\sum_{s_{-i} \in \mathcal{S}_{-i}} \sigma(s_i, s_{-i}) \left(u_i(s_i, s_{-i}) - u_i(s'_i, s_{-i}) \right) \geq 0 \quad \forall i \in \mathcal{N}, \forall s_i, s'_i \in \mathcal{S}_i$$

Consider the linear program:

Computing Correlated Equilibria in Succinct Games [1]

For almost all succinct representations it holds that the problem of finding any correlated equilibrium can be solved in polynomial time.

Consider a general n-player game. Let σ_s be the product of distributions over pure strategies for all players for strategy profile s; $\sigma_s = \Pi_i \sigma_i(s_i)$.
For a *correlated equilibrium* σ it must hold:

$$\sum_{s_{-i} \in \mathcal{S}_{-i}} \sigma(s_i, s_{-i}) \left(u_i(s_i, s_{-i}) - u_i(s'_i, s_{-i}) \right) \geq 0 \quad \forall i \in \mathcal{N}, \forall s_i, s'_i \in \mathcal{S}_i$$

Consider the linear program:

$$\max \sum_{s \in \mathcal{S}} \sigma_s$$
$$U\sigma \geq 0$$
$$\sigma \geq 0$$

Computing Correlated Equilibria in Succinct Games [1]

Consider the linear program:

$$\max \sum_{s \in \mathcal{S}} \sigma_s$$
$$U\sigma \geq 0$$
$$\sigma \geq 0$$

where $U\sigma$ are the constraints for correlated equilibrium.

Computing Correlated Equilibria in Succinct Games [1]

Consider the linear program:

$$\max \sum_{s \in \mathcal{S}} \sigma_s$$
$$U\sigma \geq 0$$
$$\sigma \geq 0$$

where $U\sigma$ are the constraints for correlated equilibrium. If there exists a correlated equilibrium, then this LP is unbounded.
Consider the dual:

Computing Correlated Equilibria in Succinct Games [1]

Consider the linear program:

$$\max \sum_{s \in \mathcal{S}} \sigma_s$$
$$U\sigma \geq 0$$
$$\sigma \geq 0$$

where $U\sigma$ are the constraints for correlated equilibrium. If there exists a correlated equilibrium, then this LP is unbounded. Consider the dual:

$$U^T y \leq -1$$
$$y \geq 0$$

Computing Correlated Equilibria in Succinct Games [1]

Consider the linear program:

$$\max \sum_{s \in \mathcal{S}} \sigma_s$$
$$U\sigma \geq 0$$
$$\sigma \geq 0$$

where $U\sigma$ are the constraints for correlated equilibrium. If there exists a correlated equilibrium, then this LP is unbounded.
Consider the dual:

$$U^T y \leq -1$$
$$y \geq 0$$

Lemma:
For every $y \geq 0$, there is a product distribution σ such that $\sigma U^T y = 0$.

Computing Correlated Equilibria in Succinct Games [1]

Computing Correlated Equilibria in Succinct Games [1]

Therefore, the dual program is infeasible.

Computing Correlated Equilibria in Succinct Games [1]

Therefore, the dual program is infeasible. Thanks to the duality we know that the original LP has exponentially many variables (σ) and the dual has exponentially many constraints.

Computing Correlated Equilibria in Succinct Games [1]

Therefore, the dual program is infeasible. Thanks to the duality we know that the original LP has exponentially many variables (σ) and the dual has exponentially many constraints.

We can make use of the ellipsoid method for the dual (*ellipsoid against hope*) – we iteratively add constraints $\sigma_\ell U^T y \leq -1$ to the dual for some product distributions σ_ℓ.

Computing Correlated Equilibria in Succinct Games [1]

Therefore, the dual program is infeasible. Thanks to the duality we know that the original LP has exponentially many variables (σ) and the dual has exponentially many constraints.

We can make use of the ellipsoid method for the dual (*ellipsoid against hope*) – we iteratively add constraints $\sigma_\ell U^T y \leq -1$ to the dual for some product distributions σ_ℓ.

Say, after L iterations the dual becomes infeasible – we have added L constraints and we have L added product distributions σ_ℓ.

Computing Correlated Equilibria in Succinct Games [1]

Therefore, the dual program is infeasible. Thanks to the duality we know that the original LP has exponentially many variables (σ) and the dual has exponentially many constraints.

We can make use of the ellipsoid method for the dual (*ellipsoid against hope*) – we iteratively add constraints $\sigma_\ell U^T y \leq -1$ to the dual for some product distributions σ_ℓ.

Say, after L iterations the dual becomes infeasible – we have added L constraints and we have L added product distributions σ_ℓ. We can translate them to the original LP, where

$$[U\sigma_\ell^T]\alpha \geq 0 \qquad \alpha \geq 0$$

and α is a correlated equilibrium (a convex combination of product distributions over $\mathcal{S}$ that satisfies CE constraints).

Computing Correlated Equilibria in Succinct Games [1]

Computing Correlated Equilibria in Succinct Games [1]

Some details were omitted:

Computing Correlated Equilibria in Succinct Games [1]

Some details were omitted:

- L is guaranteed to be polynomial, however, there is a problem with precision (in practice; addressed by the follow-up work [2])

Computing Correlated Equilibria in Succinct Games [1]

Some details were omitted:

- L is guaranteed to be polynomial, however, there is a problem with precision (in practice; addressed by the follow-up work [2])
- we need a polynomial algorithm for computing an expected utility (the product $U\sigma_\ell^T$) – i.e., the *polynomial expectation property*

Computing Correlated Equilibria in Succinct Games [1]

Some details were omitted:

- L is guaranteed to be polynomial, however, there is a problem with precision (in practice; addressed by the follow-up work [2])
- we need a polynomial algorithm for computing an expected utility (the product $U\sigma_\ell^T$) – i.e., the *polynomial expectation property*
- this algorithm is specific for each type of a succinct game:

Computing Correlated Equilibria in Succinct Games [1]

Some details were omitted:

- L is guaranteed to be polynomial, however, there is a problem with precision (in practice; addressed by the follow-up work [2])
- we need a polynomial algorithm for computing an expected utility (the product $U\sigma_\ell^T$) – i.e., the *polynomial expectation property*
- this algorithm is specific for each type of a succinct game:
  - polymatrix games

Computing Correlated Equilibria in Succinct Games [1]

Some details were omitted:

- L is guaranteed to be polynomial, however, there is a problem with precision (in practice; addressed by the follow-up work [2])
- we need a polynomial algorithm for computing an expected utility (the product $U\sigma_\ell^T$) – i.e., the *polynomial expectation property*
- this algorithm is specific for each type of a succinct game:
  - polymatrix games
  - congestion games

Computing Correlated Equilibria in Succinct Games [1]

Some details were omitted:

- L is guaranteed to be polynomial, however, there is a problem with precision (in practice; addressed by the follow-up work [2])
- we need a polynomial algorithm for computing an expected utility (the product $U\sigma_\ell^T$) – i.e., the *polynomial expectation property*
- this algorithm is specific for each type of a succinct game:
  - polymatrix games
  - congestion games

Computing Correlated Equilibria in Succinct Games [1]

This approach does not generalize to finding some optimum correlated equilibrium. For example, maximizing the expected utility of players $(\max \sum_s u_s \sigma_s)$ and constraining σ to be a probability distribution $(\sum_s \sigma_s = 1)$ would lead to dual constraints

$$(U_s)^T y \leq -u_s + z,$$

for which it is often not possible to find a polynomial-time separating oracle necessary for the ellipsoid algorithm.

Computing Correlated Equilibria in Succinct Games [1]

This approach does not generalize to finding some optimum correlated equilibrium. For example, maximizing the expected utility of players ($\max \sum_s u_s \sigma_s$) and constraining σ to be a probability distribution ($\sum_s \sigma_s = 1$) would lead to dual constraints

$$(U_s)^T y \leq -u_s + z,$$

for which it is often not possible to find a polynomial-time separating oracle necessary for the ellipsoid algorithm.

For some games it is possible to find optimal correlated equilibrium in polynomial time:

Computing Correlated Equilibria in Succinct Games [1]

This approach does not generalize to finding some optimum correlated equilibrium. For example, maximizing the expected utility of players $(\max \sum_s u_s \sigma_s)$ and constraining σ to be a probability distribution $(\sum_s \sigma_s = 1)$ would lead to dual constraints

$$(U_s)^T y \leq -u_s + z,$$

for which it is often not possible to find a polynomial-time separating oracle necessary for the ellipsoid algorithm.

For some games it is possible to find optimal correlated equilibrium in polynomial time:

1. anonymous games

Computing Correlated Equilibria in Succinct Games [1]

This approach does not generalize to finding some optimum correlated equilibrium. For example, maximizing the expected utility of players $(\max \sum_s u_s \sigma_s)$ and constraining σ to be a probability distribution $(\sum_s \sigma_s = 1)$ would lead to dual constraints

$$(U_s)^T y \leq -u_s + z,$$

for which it is often not possible to find a polynomial-time separating oracle necessary for the ellipsoid algorithm.

For some games it is possible to find optimal correlated equilibrium in polynomial time:

1. anonymous games
2. symmetric games

Computing Correlated Equilibria in Succinct Games [1]

This approach does not generalize to finding some optimum correlated equilibrium. For example, maximizing the expected utility of players ($\max \sum_s u_s \sigma_s$) and constraining σ to be a probability distribution ($\sum_s \sigma_s = 1$) would lead to dual constraints

$$(U_s)^T y \leq -u_s + z,$$

for which it is often not possible to find a polynomial-time separating oracle necessary for the ellipsoid algorithm.

For some games it is possible to find optimal correlated equilibrium in polynomial time:

1. anonymous games
2. symmetric games
3. graphical games with a bounded tree-width

Exact Polynomial Algorithm for Correlated Equilibrium

Exact Polynomial Algorithm for Correlated Equilibrium

Ellipsoid Against Hope has been simplified by [2].

Exact Polynomial Algorithm for Correlated Equilibrium

Ellipsoid Against Hope has been simplified by [2].

Instead of adding a randomized vector $x^{(k)}$, Jiang and Leyton-Brown proved that it is sufficient to use a "purified separation oracle" that adds cuts according to pure strategies.

Exact Polynomial Algorithm for Correlated Equilibrium

Ellipsoid Against Hope has been simplified by [2].

Instead of adding a randomized vector $x^{(k)}$, Jiang and Leyton-Brown proved that it is sufficient to use a "purified separation oracle" that adds cuts according to pure strategies.

As a consequence, their algorithm computes an exact and rational CE with support at most

Exact Polynomial Algorithm for Correlated Equilibrium

Ellipsoid Against Hope has been simplified by [2].

Instead of adding a randomized vector $x^{(k)}$, Jiang and Leyton-Brown proved that it is sufficient to use a "purified separation oracle" that adds cuts according to pure strategies.

As a consequence, their algorithm computes an exact and rational CE with support at most

$$1 + \sum_{i \in \mathcal{N}} |\mathcal{S}_i| \left(|\mathcal{S}_i| - 1 \right)$$

in polynomial time.

Exact Polynomial Algorithm for Correlated Equilibrium

Exact Polynomial Algorithm for Correlated Equilibrium

1. Apply the ellipsoid method using the Purified Separation Oracle, a starting ball with radius of $R = u_{max}^{5\mathcal{N}^3}$ centered at 0, and stopping when the volume of the ellipsoid is below $v = \alpha_{\mathcal{N}} u_{max}^{-7N^5}$, where $\alpha_{\mathcal{N}}$ is the volume of the $\mathcal{N}$-dimensional unit ball.

Exact Polynomial Algorithm for Correlated Equilibrium

1. Apply the ellipsoid method using the Purified Separation Oracle, a starting ball with radius of $R = u_{max}^{5\mathcal{N}^3}$ centered at 0, and stopping when the volume of the ellipsoid is below $v = \alpha_{\mathcal{N}} u_{max}^{-7N^5}$, where $\alpha_{\mathcal{N}}$ is the volume of the $\mathcal{N}$-dimensional unit ball.
2. Form the matrix U' whose columns are $U_{s^{(1)},\ldots,s^{(L)}}$ generated by the separation oracle during the run of the ellipsoid method.

Exact Polynomial Algorithm for Correlated Equilibrium

1. Apply the ellipsoid method using the Purified Separation Oracle, a starting ball with radius of $R = u_{max}^{5\mathcal{N}^3}$ centered at 0, and stopping when the volume of the ellipsoid is below $v = \alpha_{\mathcal{N}} u_{max}^{-7N^5}$, where $\alpha_{\mathcal{N}}$ is the volume of the $\mathcal{N}$-dimensional unit ball.
2. Form the matrix U' whose columns are $U_{s^{(1)},\ldots,s^{(L)}}$ generated by the separation oracle during the run of the ellipsoid method.
3. Find a feasible solution x' of the linear feasibility program

$$U'x' \geq 0,\ x' \geq 0,\ \mathbf{1}^\top x' = 1.$$

Correlated Equilibrium in Dynamic Games

Correlated equilibrium in sequential games.

Correlated Equilibrium in Dynamic Games

Correlated equilibrium in sequential games.

The signals can arrive in two different settings:

Correlated Equilibrium in Dynamic Games

Correlated equilibrium in sequential games.

The signals can arrive in two different settings:

- a player receives a signal (a recommendation) that is a strategy in the whole game (standard correlated equilibrium)

Correlated Equilibrium in Dynamic Games

Correlated equilibrium in sequential games.

The signals can arrive in two different settings:

- a player receives a signal (a recommendation) that is a strategy in the whole game (standard correlated equilibrium)
- a player receives a signal (a recommendation) that is an action to play when a certain decision point in the game is reached

Correlated Equilibrium in Dynamic Games

Correlated equilibrium in sequential games.

The signals can arrive in two different settings:

- a player receives a signal (a recommendation) that is a strategy in the whole game (standard correlated equilibrium)
- a player receives a signal (a recommendation) that is an action to play when a certain decision point in the game is reached
  - formally defined as *Extensive-Form Correlated Equilibrium* (EFCE)

Correlated Equilibrium in Dynamic Games

Correlated equilibrium in sequential games.

The signals can arrive in two different settings:

- a player receives a signal (a recommendation) that is a strategy in the whole game (standard correlated equilibrium)
- a player receives a signal (a recommendation) that is an action to play when a certain decision point in the game is reached
    - formally defined as *Extensive-Form Correlated Equilibrium* (EFCE)
    - computing one EFCE is computable in polynomial time

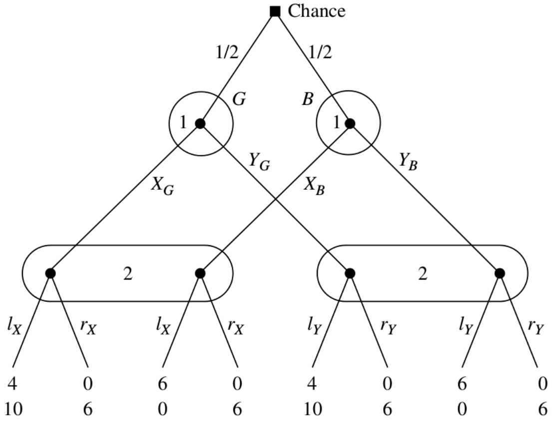
Correlated Equilibrium in Dynamic Games

Correlated equilibrium in sequential games.

The signals can arrive in two different settings:

- a player receives a signal (a recommendation) that is a strategy in the whole game (standard correlated equilibrium)
- a player receives a signal (a recommendation) that is an action to play when a certain decision point in the game is reached
  - formally defined as *Extensive-Form Correlated Equilibrium* (EFCE)
  - computing one EFCE is computable in polynomial time
  - computing an optimal EFCE is NP-hard for almost all cases (two-player games with no chance is the exception)

Extensive-Form Correlated Equilibrium



Extensive-Form Correlated Equilibrium

Representation of strategies in the two-player case: probability distribution over pairs of *relevant sequences*.

Extensive-Form Correlated Equilibrium

Representation of strategies in the two-player case: probability distribution over pairs of *relevant sequences*.

$$p(\emptyset, \emptyset) = 1; \quad 0 \leq p(\sigma_1, \sigma_2) \leq 1 \tag{1}$$

$$p(\sigma_i, \sigma_{-i}) = \sum_{a \in A(I)} p(\sigma_i a, \sigma_{-i}) \qquad \forall I \in \mathcal{I}_i, \sigma_i = \mathsf{seq}_i(I), \forall \sigma_{-i} \in rel(\sigma_i) \tag{2}$$

$$v(\sigma_{-i}) = \sum_{\sigma_i \in rel(\sigma_{-i})} p(\sigma_i, \sigma_{-i}) g_{-i}(\sigma_i, \sigma_{-i}) + \sum_{a \in A_{-i}(I)} v(\sigma_{-i} a) \quad \forall \sigma_{-i} \in \Sigma_{-i} \tag{3}$$

$$v(I, \sigma_{-i}) \geq \sum_{\sigma_i \in rel(\sigma_{-i})} p(\sigma_i, \sigma_{-i}) g_{-i}(\sigma_i, \mathsf{seq}_{-i}(I)a) + \sum_{I' \in \mathcal{I}_{-i};\ \mathsf{seq}_{-i}(I') = \sigma_{-i}(I)a} v(I', \sigma_{-i}) \tag{4}$$

$$v(\mathsf{seq}_{-i}(I)a) = v(I, \mathsf{seq}_{-i}(I)a) \qquad \forall I \in \mathcal{I}_{-i}, \forall a \in A(I) \tag{5}$$

Computing Correlated Equilibrium in Stochastic Games

Computing Correlated Equilibrium in Stochastic Games

EFCE can be generalized also to infinite (turn-based/concurrent-move) stochastic games.

Computing Correlated Equilibrium in Stochastic Games

EFCE can be generalized also to infinite (turn-based/concurrent-move) stochastic games.

We can seek for a probability distribution over a space of joint actions applicable in states of a stochastic games.

Computing Correlated Equilibrium in Stochastic Games

EFCE can be generalized also to infinite (turn-based/concurrent-move) stochastic games.

We can seek for a probability distribution over a space of joint actions applicable in states of a stochastic games.

$$V_i^\pi(h) = \sum_a \pi(h,a) Q_i^\pi(h,a,a)$$

$$Q_i^\pi(h,a,a') = R(s(h),a') + \gamma \sum_{s'} P(s'|s(h),a') V_i^\pi(\langle h,a,a',s'\rangle)$$

Computing Correlated Equilibrium in Stochastic Games

EFCE can be generalized also to infinite (turn-based/concurrent-move) stochastic games.

We can seek for a probability distribution over a space of joint actions applicable in states of a stochastic games.

$$V_i^\pi(h) = \sum_a \pi(h,a) Q_i^\pi(h,a,a)$$

$$Q_i^\pi(h,a,a') = R(s(h),a') + \gamma \sum_{s'} P(s'|s(h),a') V_i^\pi(\langle h,a,a',s'\rangle)$$

Each recommended action must be a best action to play in given state and given possible future policies:

Computing Correlated Equilibrium in Stochastic Games

EFCE can be generalized also to infinite (turn-based/concurrent-move) stochastic games.

We can seek for a probability distribution over a space of joint actions applicable in states of a stochastic games.

$$V_i^{\pi}(h) = \sum_a \pi(h,a) Q_i^{\pi}(h,a,a)$$

$$Q_i^{\pi}(h,a,a') = R(s(h),a') + \gamma \sum_{s'} P(s'|s(h),a') V_i^{\pi}(\langle h,a,a',s' \rangle)$$

Each recommended action must be a best action to play in given state and given possible future policies:

$$\forall (h,i,a_i,a_i') \qquad Q_i^{\pi}(h,a_i,a_i) \geq Q_i^{\pi}(h,a_i,a_i')$$

Computing Correlated Equilibrium in Stochastic Games

Computing Correlated Equilibrium in Stochastic Games

The achievable set of values $V(s)$ in a correlated equilibrium is a polytope in $\mathbb{R}^{|\mathcal{N}|}$.

Computing Correlated Equilibrium in Stochastic Games

The achievable set of values $V(s)$ in a correlated equilibrium is a polytope in $\mathbb{R}^{|\mathcal{N}|}$.

This is constrained due to incentives constraints of players; hence, there can be many of such constraints (undbounded number due to [3]).

Computing Correlated Equilibrium in Stochastic Games

The achievable set of values $V(s)$ in a correlated equilibrium is a polytope in $\mathbb{R}^{|\mathcal{N}|}$.

This is constrained due to incentives constraints of players; hence, there can be many of such constraints (undbounded number due to [3]).

We can approximate the polytope using a predefined set of half-spaces $H = [H_1, \ldots, H_m]$.

Computing Correlated Equilibrium in Stochastic Games

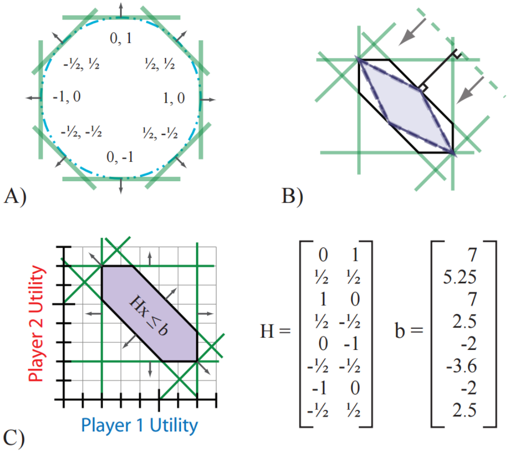
The achievable set of values $V(s)$ in a correlated equilibrium is a polytope in $\mathbb{R}^{|\mathcal{N}|}$.

This is constrained due to incentives constraints of players; hence, there can be many of such constraints (undbounded number due to [3]).

We can approximate the polytope using a predefined set of half-spaces $H = [H_1, \ldots, H_m]$.

This gives us a compact approximate representation (it is sufficient to remember the offset) that further simplifies value backup functions – this generally leads to Minkowski sum of convex sets.

Computing Correlated Equilibrium in Stochastic Games



$$H = \begin{bmatrix} 0 & 1 \\ \frac{1}{2} & \frac{1}{2} \\ 1 & 0 \\ \frac{1}{2} & -\frac{1}{2} \\ 0 & -1 \\ -\frac{1}{2} & -\frac{1}{2} \\ -1 & 0 \\ -\frac{1}{2} & \frac{1}{2} \end{bmatrix} \quad b = \begin{bmatrix} 7 \\ 5.25 \\ 7 \\ 2.5 \\ -2 \\ -3.6 \\ -2 \\ 2.5 \end{bmatrix}$$ [1]

[1] Figure from [3]

Computing Correlated Equilibrium in Stochastic Games

Computing Correlated Equilibrium in Stochastic Games

The general outline of QPACE algorithm [3] per iteration, is:

Computing Correlated Equilibrium in Stochastic Games

The general outline of QPACE algorithm [3] per iteration, is:

1. Calculate the action achievable sets $Q(s, a, a')$.

Computing Correlated Equilibrium in Stochastic Games

The general outline of QPACE algorithm [3] per iteration, is:

1. Calculate the action achievable sets $Q(s, a, a')$.
2. Construct a set of inequalities that defines the set of correlated equilibria.

Computing Correlated Equilibrium in Stochastic Games

The general outline of QPACE algorithm [3] per iteration, is:

1. Calculate the action achievable sets $Q(s, a, a')$.
2. Construct a set of inequalities that defines the set of correlated equilibria.
3. Approximately project the feasible set into value-vector space by solving a linear program for each hyperplane of $V(s)$.

Computing Correlated Equilibrium in Stochastic Games

The general outline of QPACE algorithm [3] per iteration, is:

1. Calculate the action achievable sets $Q(s, a, a')$.
2. Construct a set of inequalities that defines the set of correlated equilibria.
3. Approximately project the feasible set into value-vector space by solving a linear program for each hyperplane of $V(s)$.

The policy after a deviation can be pre-computed – so called *grim trigger strategy*, where all the other players try to punish the deviating player [3].

Computing Correlated Equilibrium in Stochastic Games

The general outline of QPACE algorithm [3] per iteration, is:

1. Calculate the action achievable sets $Q(s, a, a')$.
2. Construct a set of inequalities that defines the set of correlated equilibria.
3. Approximately project the feasible set into value-vector space by solving a linear program for each hyperplane of $V(s)$.

The policy after a deviation can be pre-computed – so called *grim trigger strategy*, where all the other players try to punish the deviating player [3].

Alternatively, we may require subgame perfection – i.e., even after a deviation the players play rationally [4].

Atomic Congestion Games

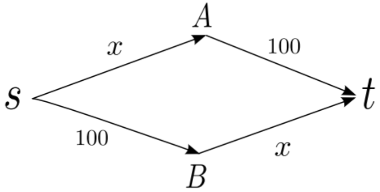
Atomic Congestion Games

We have n players, set of edges E, strategies for each player are *paths* in the network ($\mathcal{S}$), and there is a congestion function $c_e : \{0, 1, \ldots, n\} \to \mathbb{Z}^+$. When all players choose their strategy path $s_i \in \mathcal{S}_i$ we have the load of edge e, $\ell_s(e) = |\{s_i : e \in s_i\}|$ and $u_i = -\sum_{e \in s_i} c_e(\ell_s(e))$.

Atomic Congestion Games

We have n players, set of edges E, strategies for each player are *paths* in the network ($\mathcal{S}$), and there is a congestion function $c_e : \{0, 1, \ldots, n\} \to \mathbb{Z}^+$. When all players choose their strategy path $s_i \in \mathcal{S}_i$ we have the load of edge e, $\ell_s(e) = |\{s_i : e \in s_i\}|$ and $u_i = -\sum_{e \in s_i} c_e(\ell_s(e))$.

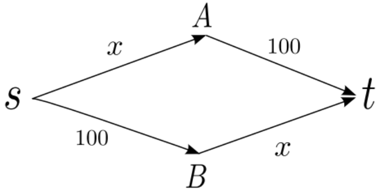
Braess' paradox



Atomic Congestion Games

We have n players, set of edges E, strategies for each player are *paths* in the network ($\mathcal{S}$), and there is a congestion function $c_e : \{0, 1, \ldots, n\} \to \mathbb{Z}^+$. When all players choose their strategy path $s_i \in \mathcal{S}_i$ we have the load of edge e, $\ell_s(e) = |\{s_i : e \in s_i\}|$ and $u_i = -\sum_{e \in s_i} c_e(\ell_s(e))$.

Braess' paradox

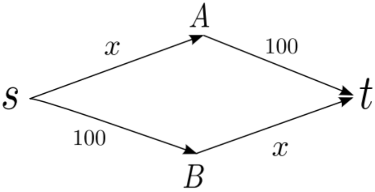


100 drivers that want to go from s to t.

Atomic Congestion Games

We have n players, set of edges E, strategies for each player are *paths* in the network ($\mathcal{S}$), and there is a congestion function $c_e : \{0, 1, \ldots, n\} \to \mathbb{Z}^+$. When all players choose their strategy path $s_i \in \mathcal{S}_i$ we have the load of edge e, $\ell_s(e) = |\{s_i : e \in s_i\}|$ and $u_i = -\sum_{e \in s_i} c_e(\ell_s(e))$.

Braess' paradox

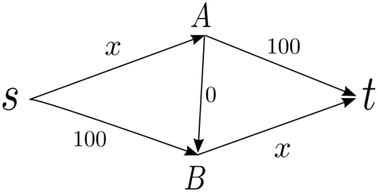


100 drivers that want to go from s to t.
What is Nash equilibrium?

Atomic Congestion Games

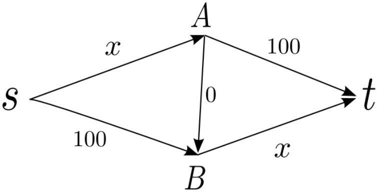
Atomic Congestion Games

Now consider that we introduce a new edge between A and B, such that $c_{(A,B)}(x) = 0,\ \forall x \in \ell_{(A,B)}$.



Atomic Congestion Games

Now consider that we introduce a new edge between A and B, such that $c_{(A,B)}(x) = 0, \ \forall x \in \ell_{(A,B)}$.



What is Nash equilibrium?

Atomic Congestion Games

Theorem

Every atomic congestion game has a pure Nash equilibrium.

Proof Sketch:

Atomic Congestion Games

Theorem

Every atomic congestion game has a pure Nash equilibrium.

Proof Sketch:

We define a potential function $\phi(s) = \sum_e \sum_{j=1}^{\ell_s(e)} c_e(j)$.

Atomic Congestion Games

Theorem

Every atomic congestion game has a pure Nash equilibrium.

Proof Sketch:

We define a potential function $\phi(s) = \sum_e \sum_{j=1}^{\ell_s(e)} c_e(j)$.

Define $\ell_s^{\leq i}(e) = |\{s_j : e \in s_j \wedge j = 1, \ldots, i\}|$.

Atomic Congestion Games

Theorem

Every atomic congestion game has a pure Nash equilibrium.

Proof Sketch:

We define a potential function $\phi(s) = \sum_e \sum_{j=1}^{\ell_s(e)} c_e(j)$.
Define $\ell_s^{\leq i}(e) = |\{s_j : e \in s_j \wedge j = 1, \ldots, i\}|$. Now,

$$\phi(s) = \sum_{i=1}^{n} \sum_{e \in s_i} c_e(\ell_s^{\leq i}(e))$$

Atomic Congestion Games

Theorem

Every atomic congestion game has a pure Nash equilibrium.

Proof Sketch:
We define a potential function $\phi(s) = \sum_e \sum_{j=1}^{\ell_s(e)} c_e(j)$.
Define $\ell_s^{\leq i}(e) = |\{s_j : e \in s_j \wedge j = 1, \ldots, i\}|$. Now,

$$\phi(s) = \sum_{i=1}^{n} \sum_{e \in s_i} c_e(\ell_s^{\leq i}(e))$$

Consider player n switching from s_n to s_n'

Atomic Congestion Games

Atomic Congestion Games

Proof Continued:

$$\phi(s) = \sum_{i=1}^{n} \sum_{e \in s_i} c_e(\ell_s^{\leq i}(e))$$

Atomic Congestion Games

Proof Continued:

$$\phi(s) = \sum_{i=1}^{n} \sum_{e \in s_i} c_e(\ell_s^{\leq i}(e))$$

Consider a player (WLOG n) switching from s_i to s'_n:

Atomic Congestion Games

Proof Continued:

$$\phi(s) = \sum_{i=1}^{n} \sum_{e \in s_i} c_e(\ell_s^{\leq i}(e))$$

Consider a player (WLOG n) switching from s_i to s'_n:

$$\phi(s) - \phi(s') = \sum_{e \in s_n} c_e(\ell_s^{\leq n}(e)) - \sum_{e \in s'_n} c_e(\ell_{s'}^{\leq n}(e)) \quad (6)$$

$$= \sum_{e \in s_n} c_e(\ell_s(e)) - \sum_{e \in s'_n} c_e(\ell_{s'}(e)) \quad (7)$$

$$= c_n(s) - c_n(s') = u_n(s') - u_n(s) \quad (8)$$

Atomic Congestion Games

Proof Continued:

$$\phi(s) = \sum_{i=1}^{n} \sum_{e \in s_i} c_e(\ell_s^{\leq i}(e))$$

Consider a player (WLOG n) switching from s_i to s'_n:

$$\begin{aligned}
\phi(s) - \phi(s') &= \sum_{e \in s_n} c_e(\ell_s^{\leq n}(e)) - \sum_{e \in s'_n} c_e(\ell_{s'}^{\leq n}(e)) && (6) \\
&= \sum_{e \in s_n} c_e(\ell_s(e)) - \sum_{e \in s'_n} c_e(\ell_{s'}(e)) && (7) \\
&= c_n(s) - c_n(s') = u_n(s') - u_n(s) && (8)
\end{aligned}$$

Function ϕ attains a minimum (that must exist) at a Nash equilibrium. $\square$

Congestion Games

Congestion Games

Finding a pure Nash equilibrium is PLS-complete for congestion games.

Congestion Games

Finding a pure Nash equilibrium is PLS-complete for congestion games.

This holds for generalizations:

Congestion Games

Finding a pure Nash equilibrium is PLS-complete for congestion games.

This holds for generalizations:

- weighted congestion games

Congestion Games

Finding a pure Nash equilibrium is PLS-complete for congestion games.

This holds for generalizations:

- weighted congestion games
- offers a strongly polynomial approximate algorithm for non-atomic congestion games

Congestion Games

Finding a pure Nash equilibrium is PLS-complete for congestion games.

This holds for generalizations:

- weighted congestion games
- offers a strongly polynomial approximate algorithm for non-atomic congestion games

For some subclasses, it is polynomial to find a pure NE (e.g., for symmetric network congestion games due to min-cost flow).

Congestion Games

Finding a pure Nash equilibrium is PLS-complete for congestion games.

This holds for generalizations:

- weighted congestion games
- offers a strongly polynomial approximate algorithm for non-atomic congestion games

For some subclasses, it is polynomial to find a pure NE (e.g., for symmetric network congestion games due to min-cost flow).

Many works study *Price of Anarchy* (or other) concepts in such games.

Generalization to Potential Games

[2] In potential games, a maximum of the potential function is sought which is different to the congestion games case.

Generalization to Potential Games

This result generalizes to a wider class of *potential games* [6].

[2] In potential games, a maximum of the potential function is sought which is different to the congestion games case.

Generalization to Potential Games

This result generalizes to a wider class of *potential games* [6]. Informally, a potential game is such that has a potential function same as in the proof for the congestion games[2]:

$$\phi(s') - \phi(s) = u_i(s') - u_i(s),$$

where i is the deviating player.

[2] In potential games, a maximum of the potential function is sought which is different to the congestion games case.

Generalization to Potential Games

This result generalizes to a wider class of *potential games* [6]. Informally, a potential game is such that has a potential function same as in the proof for the congestion games[2]:

$$\phi(s') - \phi(s) = u_i(s') - u_i(s),$$

where i is the deviating player.

Theorem ([5])

Any exact potential game is isomorphic to a congestion game.

[2]In potential games, a maximum of the potential function is sought which is different to the congestion games case.

Generalization to Potential Games

This result generalizes to a wider class of *potential games* [6]. Informally, a potential game is such that has a potential function same as in the proof for the congestion games[2]:

$$\phi(s') - \phi(s) = u_i(s') - u_i(s),$$

where i is the deviating player.

Theorem ([5])

Any exact potential game is isomorphic to a congestion game.

Theorem (shortened [5])

Any PLS problem can be reduced in polynomial time to a general potential game.

[2] In potential games, a maximum of the potential function is sought which is different to the congestion games case.

Example of Potential Games

Example of Potential Games

Prisoners' Dilemma:

Example of Potential Games

Prisoners' Dilemma:

a

	C	D
C	$-1, -1$	$-6, \ 0$
D	$0, -6$	$-4, -4$

b

	C	D
C	1	2
D	2	4

Polymatrix Games

Polymatrix Games

A *polymatrix game* $\mathcal{G}$ consists of the following:

Polymatrix Games

A *polymatrix game* $\mathcal{G}$ consists of the following:

- a finite set of players $\mathcal{N} = \{1, \ldots, n\}$, where each player corresponds to a node in a graph, and a set of edges $\mathcal{E}$ that are unordered pairs of players (i, j) such that $i \neq j$

Polymatrix Games

A *polymatrix game* $\mathcal{G}$ consists of the following:

- a finite set of players $\mathcal{N} = \{1, \ldots, n\}$, where each player corresponds to a node in a graph, and a set of edges $\mathcal{E}$ that are unordered pairs of players (i, j) such that $i \neq j$
- a finite set of strategies for each player $\mathcal{S}_i$

Polymatrix Games

A *polymatrix game* $\mathcal{G}$ consists of the following:

- a finite set of players $\mathcal{N} = \{1, \ldots, n\}$, where each player corresponds to a node in a graph, and a set of edges $\mathcal{E}$ that are unordered pairs of players (i, j) such that $i \neq j$
- a finite set of strategies for each player $\mathcal{S}_i$
- for each edge $e \in \mathcal{E}$, there is a two-player game (u^{ij}, u^{ji}) where the players are i, j, strategy sets $\mathcal{S}_i, \mathcal{S}_j$ respectively, and utility function $u^{ij} : \mathcal{S}_i \times \mathcal{S}_j \to \mathbb{R}$ (similarly for u^{ji})

Polymatrix Games

A *polymatrix game* $\mathcal{G}$ consists of the following:

- a finite set of players $\mathcal{N} = \{1, \ldots, n\}$, where each player corresponds to a node in a graph, and a set of edges $\mathcal{E}$ that are unordered pairs of players (i, j) such that $i \neq j$
- a finite set of strategies for each player $\mathcal{S}_i$
- for each edge $e \in \mathcal{E}$, there is a two-player game (u^{ij}, u^{ji}) where the players are i, j, strategy sets $\mathcal{S}_i, \mathcal{S}_j$ respectively, and utility function $u^{ij} : \mathcal{S}_i \times \mathcal{S}_j \to \mathbb{R}$ (similarly for u^{ji})
- for each player $i \in \mathcal{N}$ and strategy profile $s = (s_1, \ldots, s_n)$, the utility of player i is

$$u_i(s) = \sum_{\forall j \in \mathcal{N} : (i,j) \in \mathcal{E}} u^{ij}(s_i, s_j)$$

Polymatrix Games

Polymatrix Games

For some subclasses that admit pure Nash equilibria, it is PLS-hard to compute one (e.g., in case we have symmetric two-player games over the edges – also known as "team polymatrix games").

Polymatrix Games

For some subclasses that admit pure Nash equilibria, it is PLS-hard to compute one (e.g., in case we have symmetric two-player games over the edges – also known as “team polymatrix games”).

Examples: coordination game among agents, games among agents in a network

Zero-Sum Polymatrix Games [7]

Zero-Sum Polymatrix Games [7]

We talk about zero-sum polymatrix games if for all strategy profiles $s \in \mathcal{S}$ it holds that $\sum_{i \in \mathcal{N}} u_i(s) = 0$.

Zero-Sum Polymatrix Games [7]

We talk about zero-sum polymatrix games if for all strategy profiles $s \in \mathcal{S}$ it holds that $\sum_{i \in \mathcal{N}} u_i(s) = 0$.
Example: security game between multiple defenders and multiple attackers

Zero-Sum Polymatrix Games [7]

We talk about zero-sum polymatrix games if for all strategy profiles $s \in \mathcal{S}$ it holds that $\sum_{i \in \mathcal{N}} u_i(s) = 0$.
Example: security game between multiple defenders and multiple attackers

Theorem

A Nash equilibrium of a zero-sum polymatrix game can be found in polynomial time by solving a single linear program.

Zero-Sum Polymatrix Games [7]

We talk about zero-sum polymatrix games if for all strategy profiles $s \in \mathcal{S}$ it holds that $\sum_{i \in \mathcal{N}} u_i(s) = 0$.
Example: security game between multiple defenders and multiple attackers

Theorem

A Nash equilibrium of a zero-sum polymatrix game can be found in polynomial time by solving a single linear program.

Proof Sketch:

Zero-Sum Polymatrix Games [7]

We talk about zero-sum polymatrix games if for all strategy profiles $s \in \mathcal{S}$ it holds that $\sum_{i \in \mathcal{N}} u_i(s) = 0$.
Example: security game between multiple defenders and multiple attackers

Theorem

A Nash equilibrium of a zero-sum polymatrix game can be found in polynomial time by solving a single linear program.

Proof Sketch:

$$
\begin{aligned}
&\min_{x,w} \sum_{i \in \mathcal{N}} w_i \\
&\text{s.t.} \qquad w_i \geq u_i(s_i, x_{-i}) \qquad \forall i \in \mathcal{N},\ \forall s_i \in \mathcal{S}_i \\
&x_i \in \Delta(\mathcal{S}_i)
\end{aligned}
$$

Zero-Sum Polymatrix Games [7]

Zero-Sum Polymatrix Games [7]

Proof Sketch:

Zero-Sum Polymatrix Games [7]

Proof Sketch:

$$\begin{aligned}
&\min_{x,w} \sum_{i\in\mathcal{N}} w_i \\
&\text{s.t.} \qquad w_i \geq u_i(s_i, x_{-i}) \qquad \forall i \in \mathcal{N},\ \forall s_i \in \mathcal{S}_i \\
&x_i \in \Delta(\mathcal{S}_i)
\end{aligned}$$

It holds

Zero-Sum Polymatrix Games [7]

Proof Sketch:

$$
\begin{aligned}
& \min_{x,w} \sum_{i \in \mathcal{N}} w_i \\
& \text{s.t.} \qquad w_i \geq u_i(s_i, x_{-i}) \qquad \forall i \in \mathcal{N},\ \forall s_i \in \mathcal{S}_i \\
& x_i \in \Delta(\mathcal{S}_i)
\end{aligned}
$$

It holds

$$
\sum_{i \in \mathcal{N}} w_i \geq \sum_{i \in \mathcal{N}} \max_{s \in \mathcal{S}_i} u_i(s, x_{-i}) = \max_{x_i \in \Delta(\mathcal{S}_i)} \sum_{i \in \mathcal{N}} u_i(s, x_{-i}) \geq 0
$$

Setting $w_i = \max_{s \in \mathcal{S}_i} u_i(s, x^*_{-i})$, where x^* is a NE is a feasible solution (and vice versa).

Zero-Sum Polymatrix Games

Zero-Sum Polymatrix Games

Generalization of the min-max theorem and two-player zero-sum games.

Zero-Sum Polymatrix Games

Generalization of the min-max theorem and two-player zero-sum games.

Many "nice properties" of two-player zero-sum games **do not hold**:

Zero-Sum Polymatrix Games

Generalization of the min-max theorem and two-player zero-sum games.

Many "nice properties" of two-player zero-sum games **do not hold**:

- players do not have unique payoff value (or *value of the game*)

Zero-Sum Polymatrix Games

Generalization of the min-max theorem and two-player zero-sum games.

Many "nice properties" of two-player zero-sum games **do not hold**:

- players do not have unique payoff value (or *value of the game*)
- equilibrium strategies are not max-min strategies

Zero-Sum Polymatrix Games

Generalization of the min-max theorem and two-player zero-sum games.

Many "nice properties" of two-player zero-sum games **do not hold**:

- players do not have unique payoff value (or *value of the game*)
- equilibrium strategies are not max-min strategies
- equilibrium strategies are not exchangeable

References I

(besides the books)

[1] C. H. Papadimitriou and T. Roughgarden.
Computing Correlated Equilbria in Multi-Player Games.
Journal of ACM, 2008.

[2] A. X. Jiang and K. Leyton-Brown, "Polynomial-time computation of exact correlated equilibrium in compact games," in *Proceedings of the 12th ACM Conference on Electronic Commerce*, EC '11, (New York, NY, USA), pp. 119–126, ACM, 2011.

[3] L. MacDermed, K. S. Narayan, C. L. Isbell, and L. Weiss, "Quick polytope approximation of all correlated equilibria in stochastic games," in *Proceedings of the Twenty-Fifth AAAI Conference on Artificial Intelligence*, AAAI'11, pp. 707–712, AAAI Press, 2011.

[4] C. Murray and G. Gordon, "Finding correlated equilibria in general sum stochastic games," tech. rep., CMU-ML-07-113, Carnegie Mellon University, 2007.

References II

[5] A. Fabrikant, C. H. Papadimitriou, and K. Talwar.
The Complexity of Pure Nash Equilibria.
In *STOC*, 2004.

[6] D. Monderer and L. S. Shapley.
Potential games.
Games and Economic Behavior, 14:124–143, 1996.

[7] Y. Cai, O. Candogan, C. Daskalakis, and C. Papadimitriou.
Zero-sum polymatrix games: A generalization of minmax.
Mathematics of Operations Research.