Proceedings of the Twenty-Sixth AAAI Conference on Artificial Intelligence

Computing Equilibria in Two–Player Zero–Sum Continuous Stochastic Games with Switching Controller

Guido Bonomi and **Nicola Gatti** and **Fabio Panozzo** and **Marcello Restelli**
Politecnico di Milano
Piazza Leonardo da Vinci 32
Milano, Italy
{bonomi, ngatti, panozzo, restelli}@elet.polimi.it

Abstract

Equilibrium computation with continuous games is currently a challenging open task in artificial intelligence. In this paper, we design an iterative algorithm that finds an ϵ–approximate Markov perfect equilibrium with two–player zero–sum continuous stochastic games with switching controller. When the game is polynomial (i.e., utility and state transitions are polynomial functions), our algorithm converges to $\epsilon = 0$ by exploiting semidefinite programming. When the game is not polynomial, the algorithm exploits polynomial approximations and converges to an ϵ value whose upper bound is a function of the maximum approximation error with infinity norm. To our knowledge, this is the first algorithm for equilibrium approximation with arbitrary utility and transition functions providing theoretical guarantees. The algorithm is also empirically evaluated.

Introduction

The computation of game–theoretic solutions is a central task in artificial intelligence (Shoham and Leyton-Brown 2010). Game theory provides the most elegant *game models* and *solution concepts*, but it leaves the problem to compute a solution open (Fudenberg and Tirole 1991). A game is a pair: a *mechanism* (specifying the rules) and the *strategies* (specifying the agents' behavior). The central solution concept is the Nash equilibrium (NE). Every finite game is guaranteed to have at least one NE in mixed strategies, but its computation is PPAD–complete even with just two agents (Chen, Deng, and Teng 2009). Instead, with two agents and zero–sum utilities the problem is in P.

A challenging game class is composed by continuous games (Karlin 1951), in which the actions of the agents are real values. These games are very common in practical scenarios (e.g., auctions, bargaining, dynamic games). Differently from finite games, continuous games may not admit NEs, e.g., due to discontinuity of the utility functions (Rosen 1987). With compact and convex action spaces and continuous utility functions, a slight variation of the Kakutani's fixed point theorem assures the existence of at least one NE in mixed strategies (Glicksberg 1952). Adding the hypothesis of concave utility functions, continuous games always admit equilibria in pure strategies, utility functions being equivalent to the convexification of a finite set of actions (Rosen 1987). However, in general settings, NEs are in mixed strategies and their study is hard. A very special class is that of *separable games* where each agent's payoff is a finite sum of products of functions in each player's strategy separately (e.g. as polynomials). It is known that every separable game admits an NE finitely supported. This has been shown with zero–sum games in (Dresher, Melvin and Shapley 1950), and recently with general–sum games in (Stein, Ozdaglar, and Parrilo 2008). Instead, when games are not separable, NEs may be not finitely supported (Karlin 1959).

Few computational results are known for continuous games. For the special case of two agents, zero–sum, polynomial utility functions, the computation of an NE can be formulated (Parrilo 2006) as a pair (primal/dual) of semidefinite programming problems (SDP), that can be efficiently solved by convex programming algorithms (Blekherman, Parrilo, and Thomas 2012). With general separable utility functions, non–linear programming tools should be used, without any guarantee of finding the optimal solution and with algorithms that hardly scale (to the best of our knowledge, no work in the literature has dealt with this problem). With arbitrary (non–separable) utility functions, some works deal with the problem of approximate a pure strategy equilibrium by exploiting local search and Monte Carlo methods with normal–form games (Vorobeychik and Wellman 2008) and extensive–form games (Gatti and Restelli 2011), but no work deals with the problem of finding mixed strategy NEs.

In this paper, we focus on continuous stochastic games with switching control. The literature only provides results for finite stochastic games with switching control (Vrieze et al. 1983) and for polynomial continuous stochastic games with single controller (Shah and Parrilo 2007). Our original contributions include the following.

- We provide an iterative SDP based algorithm that converges to a Markov perfect equilibrium (MPE) — the appropriate solution concept for stochastic games — when both the reward and the state transitions are polynomial functions and returns an ϵ–approximate MPE (ϵ–MPE) with $\epsilon \leq \overline{\epsilon}$, where $\overline{\epsilon}$ is given as input.
- We use our algorithm to approximate solutions of non–polynomial games: approximating the reward and state

transitions given as input with polynomials, solving the approximated game, and providing theoretical upper bounds on the quality of the solutions as functions of the approximation error with infinity norm.

- We experimentally evaluate the performance of our algorithm in terms of iterations and compute time.

Game model and solution concepts

A two–player zero–sum *stochastic game* $\mathcal{G}$, introduced in (Shapley 1953), is a tuple (N, S, X, P, R, γ), where: S is a finite set of states ($s \in S$ denotes a generic state), N is the set of agents ($i \in N$ denotes a generic agent), X is the set of actions ($X_{i,s} \subseteq X$ denotes the set of actions available to agent i at state s, and $x_i \in X_{i,s}$ a generic action of agent i), P is a set of maps $p_{s,s'} : \times_{i \in N} X_{i,s} \to [0,1]$ assigning the probability to move from state s to state s' given the actions of all the agents, R is a set of maps $r_s : \times_{i \in N} X_{i,s} \to \mathbb{R}$ assigning each action profile a reward, $\gamma \in (0,1)$ is the temporal discount factor. Without loss of generality, we assume that agent 1 is the max agent. We denote with u_s, where $s \in S$, the utility function of agent 1, while $-u_s$ is the utility function of agent 2. By Bellman equation, u_s is defined as $u_s(\mathbf{x}_1, \mathbf{x}_2) = r_s(x_{1,s}, x_{2,s}) + \sum_{s'} p_{s,s'}(x_{1,s}, x_{2,s}) \cdot u_{s'}(\mathbf{x}_1, \mathbf{x}_2)$, where $\mathbf{x}_i$ is the vector of actions of agent i over all the states and $x_{i,s}$ is the specific action of agent i at state s.

A *continuous* stochastic game is a stochastic game (N, S, X, P, R, γ) where action spaces $X_{i,s}$ are limited subspaces (usually compact) of the Euclidean space and maps $p_{s,s'}$ and r_s are generic functions. We focus on continuous stochastic games with *switching controller*, in which at each state s functions $p_{s,s'}$ depend either on $x_1 \in X_{1,s}$ or on $x_2 \in X_{2,s}$. The agent i who drives the transition at state s is said the *controller* of such state and it is denoted by c_s. Notice that different states can be controlled by different agents. We partition the state space S as $S = S_1 \cup S_2$ where S_i is the set of states where $c_s = i$. When the game is polynomial, we have: $p_{s,s'}(x_{c_s}) = \sum_{k=0}^{m} p_{s,s',k} \cdot (x_{c_s})^k$ and $r_s(x_1, x_2) = \sum_{k=0}^{m} \sum_{j=0}^{m} r_{s,k,j} \cdot (x_1)^k \cdot (x_2)^j$, where m is the maximum degree of all the polynomials and $p_{s,s',k}$, $r_{s,k,j} \in \mathbb{R}$ are coefficients.

A strategy profile $(\boldsymbol{\sigma}_1, \boldsymbol{\sigma}_2)$ specifies the strategy $\sigma_{i,s}$ of each agent i at each state s as a probability measure over the space of actions $X_{i,s}$. An MPE is a strategy profile $(\boldsymbol{\sigma}_1^*, \boldsymbol{\sigma}_2^*)$ where each strategy is conditioned only on the local state of the agent, and such that no agent can improve her utility u_s (or $-u_s$) in any state s by changing her strategy. With zero–sum games, an MPE corresponds to maxmin/minmax strategies. In this paper, we resort also to the ϵ–MPE concept, defined as: a strategy profile $(\boldsymbol{\sigma}_1, \boldsymbol{\sigma}_2)$ is an ϵ–MPE if no agent can improve her utility u_s (or $-u_s$) in some state s more than ϵ by changing her strategy. Obviously, an ϵ–MPE with $\epsilon = 0$ is an MPE. Furthermore, while an MPE may not exist with continuous games, it is always possible to find ϵ–MPEs for some ϵ.

Algorithm 1 Iterative Nash approximation

1: assign $\hat{u}_s = 0$ for every $s \in S$
2: **repeat**
3: $\quad[\hat{\mathbf{u}}, \boldsymbol{\sigma}_2^{S_1}] =$ solve $\text{PS}_1(\hat{\mathbf{u}})$
4: $\quad[\ \ , \boldsymbol{\sigma}_1^{S_1}] =$ solve $\text{DS}_1(\hat{\mathbf{u}})$
5: $\quad[\hat{\mathbf{u}}, \boldsymbol{\sigma}_1^{S_2}] =$ solve $\text{PS}_2(\hat{\mathbf{u}})$
6: $\quad[\ \ , \boldsymbol{\sigma}_2^{S_2}] =$ solve $\text{DS}_2(\hat{\mathbf{u}})$
7: $\quad$assign $\overline{\boldsymbol{\sigma}}_1 = (\boldsymbol{\sigma}_1^{S_1}, \boldsymbol{\sigma}_1^{S_2})$ and $\overline{\boldsymbol{\sigma}}_2 = (\boldsymbol{\sigma}_2^{S_1}, \boldsymbol{\sigma}_2^{S_2})$
8: $\quad$calculate $\overline{\mathbf{u}}$ with $(\overline{\boldsymbol{\sigma}}_1, \overline{\boldsymbol{\sigma}}_2)$
9: $\quad\mathbf{u}_1^* =$ solve BR_1 with $\overline{\boldsymbol{\sigma}}_2 = (\boldsymbol{\sigma}_2^{S_1}, \boldsymbol{\sigma}_2^{S_2})$
10: $\quad\mathbf{u}_2^* =$ solve BR_2 with $\overline{\boldsymbol{\sigma}}_1 = (\boldsymbol{\sigma}_1^{S_1}, \boldsymbol{\sigma}_1^{S_2})$
11: **until** $\max\{\|\mathbf{u}_1^* - \overline{\mathbf{u}}\|_\infty, \|\overline{\mathbf{u}} - \mathbf{u}_2^*\|_\infty\} > \overline{\epsilon}$

Equilibrium computation with polynomial games

Here, we describe the algorithm converging to an MPE with polynomial games. For the sake of clarity, we initially describe the algorithm omitting details on the SDPs the algorithm uses. Details are provided later.

Algorithm

The procedure is summarized in Algorithm 1. The algorithm uses auxiliary utilities $\hat{\mathbf{u}}$. As shown below, these utilities converge to the utilities at the equilibrium, denoted by $\mathbf{u}^*$.

Initially, (Steps 1 and 2) the algorithm initializes $\hat{u}_s = 0$ for every $s \in S$. Then, the algorithm repeats Steps 3–11 until an $\overline{\epsilon}$–MPE has been found where $\overline{\epsilon}$ is given as input.

At first, the algorithm finds the optimal strategies in the states S_1 controlled by agent 1 when the utilities of the states $s \in S_2$ are fixed to $\hat{u}_s$, and assigns the returned optimal utility values of states $s \in S_1$ to $\hat{u}_s$. This is accomplished into two steps: in Step 3, the optimal strategy of agent 2 is computed by solving an SDP called PS_1, while in Step 4, the optimal strategy of agent 1 is computed by solving an SDP called DS_1. PS_1 is the primal problem, while DS_1 is the dual problem. (As we will discuss in the following section, strong duality holds for these two problems.) The problem PS_1:

$$\textbf{(PS}_1\textbf{)} \quad \min \quad \sum_{s \in S_1} u_s$$

$$\mathbb{E}_{x_2 \sim \sigma_{2,s}}[r_s(x_1, x_2)] + \gamma \sum_{s' \in S_1} u_{s'} \cdot p_{s,s'}(x_1) + \gamma \sum_{s' \in S_2} \hat{u}_{s'} \cdot p_{s,s'}(x_1) \leq u_s \quad \forall s \in S_1,\ x_1 \in X_{1,s} \qquad (1)$$

$$\sigma_{2,s} \text{ is a probability measure on } X_{2,s} \quad \forall s \in S_1 \qquad (2)$$

where $\mathbb{E}[\cdot]$ denotes the expectation of '$\cdot$'. Notice that (1) contains infinitely many constraints, one for each value of $x_1 \in X_{1,s}$. PS_1 returns the optimal strategy $\sigma_{2,s}^{S_1}$ and the optimal utilities $\overline{u}_s$ in states $s \in S_1$ given that utilities of states S_2 are fixed and equal to $\hat{u}_s$. Utilities $\overline{u}_s$ are assigned to $\hat{u}_s$ for $s \in S_1$. Similarly, the dual problem DS_1 is:

$$\textbf{(DS}_1\textbf{)} \quad \max \quad \left(\sum_{s \in S_2} z_s - \gamma \cdot \sum_{s' \in S_1} \mathbb{E}_{x_2 \sim \sigma_{2,s}}[p_{s,s'}(x_2) \cdot \hat{u}_{s'}] \right)$$

$$\begin{array}{lll}\sum_{s\in S_2}\mathbb{E}_{x_2\sim\sigma_{2,s}}[\mathbf{1}_{s=s'}-\gamma p_{s,s'}(x_2)] & = 1 & \forall s\in S_2\\ -z_s-\mathbb{E}_{x_2\sim\sigma_{2,s}}[r_s(x_1,x_2)] & \geq 0 & \forall s\in S_2, x_1\in X_{1,s}\\ \\ \sigma_{2,s} \text{ is a probability measure on } X_{2,s} & & \forall s\in S_2\end{array}$$

where z_s are auxiliary variables, and $\mathbf{1}_{s=s'}$ is equal to 1 when $s=s'$ and 0 otherwise. As above, DS_1 contains infinite constraints one for each value of $x_2\in X_{2,s}$. DS_1 returns the optimal strategy $\sigma_{1,s}^{S_1}$ in states S_1.

Then, in Steps 5 and 6, the algorithm repeats Steps 3 and 4 for the states S_2 by solving two SDPs, called PS_2 and DS_2 and omitted here being similar to PS_1 and DS_1, respectively. These programs return the optimal strategies $\sigma_{1,s}^{S_2}$, $\sigma_{2,s}^{S_2}$ and the optimal utilities $\overline{u}_s$ in states S_2 given $\hat{u}_s$ at $s\in S_1$. Utilities $\overline{u}_s$ are assigned to $\hat{u}_s$ for $s\in S_2$.

In the next steps, the current solution is considered as an ϵ–MPE and the value of ϵ is estimated as follows: given the joint strategy at the current iteration, the algorithm computes the utilities and compares them w.r.t. the utilities provided by each agent's best response. The utilities provided by the current solution $(\overline{\boldsymbol{\sigma}}_1,\overline{\boldsymbol{\sigma}}_2)$ with $\overline{\boldsymbol{\sigma}}_1=(\boldsymbol{\sigma}_1^{S_1},\boldsymbol{\sigma}_1^{S_2})$ and $\overline{\boldsymbol{\sigma}}_2=(\boldsymbol{\sigma}_2^{S_1},\boldsymbol{\sigma}_2^{S_2})$ can be easily obtained by solving the following linear system (Step 8):

$$\mathbb{E}_{\substack{x_1\sim\overline{\sigma}_{1,s}\\ x_2\sim\overline{\sigma}_{2,s}}}\left[r_s(x_1,x_2)+\gamma\textstyle\sum_{s'\in S}\overline{u}_{s'}\cdot p_{s,s'}(x_1)\right]=\overline{u}_s \quad \forall s\in S_1$$

$$\mathbb{E}_{\substack{x_1\sim\overline{\sigma}_{1,s}\\ x_2\sim\overline{\sigma}_{2,s}}}\left[r_s(x_1,x_2)+\gamma\textstyle\sum_{s'\in S}\overline{u}_{s'}\cdot p_{s,s'}(x_2)\right]=\overline{u}_s \quad \forall s\in S_2$$

The problem of computing agent i's best response given $\overline{\boldsymbol{\sigma}}_{-i}$ is a continuous MDP with polynomial reward and transition functions (and therefore it admits an optimal pure strategy). This problem can be formulated, as shown later, as an SDP. Thus, we have two SDPs, called BR_1 and BR_2 for agent 1 and agent 2, respectively. BR_1 (Step 9) is formulated as:

$$\textbf{(BR}_1\textbf{)}\quad \min \quad \textstyle\sum_{s\in S}u_s$$

$$\begin{array}{lll}\mathbb{E}_{x_2\sim\overline{\sigma}_{2,s}}[r_s(x_1,x_2)]+\gamma\sum_{s'\in S}u_{s'}\cdot & & \forall s\in S_1, \quad (3)\\ \cdot p_{s,s'}(x_1) & \leq u_s & x_1\in X_{1,s}\end{array}$$

$$\begin{array}{lll}\mathbb{E}_{x_2\sim\overline{\sigma}_{2,s}}[r_s(x_1,x_2)+\gamma\sum_{s'\in S}u_{s'}\cdot & & \forall s\in S_2, \quad (4)\\ \cdot p_{s,s'}(x_2)] & \leq u_s & x_1\in X_{1,s}\end{array}$$

BR_2 (Step 9) is defined similarly and then omitted. Call $\mathbf{u}_1^*$ the vector of utilities returned by BR_1 and $\mathbf{u}_2^*$ the vector of utilities returned by BR_2. Call $\overline{\mathbf{u}}$ the vector of utilities $\overline{u}_s$. The ϵ value of the current strategy profile $(\overline{\boldsymbol{\sigma}}_1,\overline{\boldsymbol{\sigma}}_2)$ is $\max_i\{\|\mathbf{u}_i^*-\overline{\mathbf{u}}\|_\infty\}$, that is, the maximum loss of all the agents over all the states. The algorithm terminates if $\epsilon\leq\overline{\epsilon}$.

We can state the following theorem.

Theorem 1. *Given polynomial reward and transition functions, Algorithm 1 returns an $\overline{\epsilon}$–MPE.*

Proof. The proposed algorithm needs non–negative reward functions. This assumption can be easily satisfied (without changing the equilibrium strategies) by adding a constant to the reward functions, so that all the rewards get strictly positive. We observe that the solution of the problems PS_1 and PS_2 are monotonically increasing in $\hat{u}_s$. That is, given $\hat{u}'_s$ and $\hat{u}''_s$ such that $\hat{u}'_s\leq\hat{u}''_s\leq u_s^*$, where u_s^* are the utilities at the equilibrium, for every $s\in S$ and called $\overline{u}'_s$ the solution of PS_i when the input is $\hat{u}'_s$ and $\overline{u}''_s$ the solution of PS_i when the input is $\hat{u}''_s$, we have that $\overline{u}'_s\leq\overline{u}''_s\leq u_s^*$. As a result, when reward functions are non–negative, starting from $\hat{u}_s=0$ for every $s\in S$ we have a sequence of $\hat{u}_s$ that is monotonically increasing as long $\hat{u}_s^i\neq u_s^*$. Therefore, the algorithm converges to an MPE. The algorithm stops when no agent, given the strategy of the opponent, can gain more than $\overline{\epsilon}$. Therefore, the algorithm returns an $\overline{\epsilon}$–MPE. □

Differently from finite switching–control games (Vrieze et al. 1983), when games are continuous there is no guarantee that the algorithm converges by a finite number of steps. At each iteration, it is possible to check whether or not there is an MPE with the current support, i.e., the set of actions played with non–zero probability in $(\overline{\boldsymbol{\sigma}}_1,\overline{\boldsymbol{\sigma}}_2)$. (When games are finite, this problem can be formulated as a linear programming problem; it can be easily shown that with continuous polynomial games such problem can be formulated as an SDP.) Since with finite games the number of supports is finite, it is possible to guarantee the termination by finite time. The same approach cannot be used with continuous games, the number of supports being infinite.

Semidefinite programming formulation

We show how PS_1 can be formulated as an SDP (the formulations of DS_1, PS_2, and DS_2 are similar and therefore they will be omitted here). At first, we rewrite constraint (1) by considering that r_s and $p_{s,s'}$ are polynomial functions and by expanding the expected value operator $\mathbb{E}[\cdot]$:

$$\begin{aligned}u_s-\sum_{k=0}^{m}\sum_{j=0}^{m}r_{s,k,j}\cdot(x_1)^k\cdot\left(\int_{X_{2,s}}\sigma_{2,s}(x_2)\cdot(x_2)^j dx_2\right)\\ -\gamma\sum_{s'\in S}u'_s\cdot\sum_{k=0}^{m}p_{s,s',k}\cdot(x_1)^k\geq 0 \qquad \forall s\in S_1, x_1\in X_{1,s}\end{aligned}$$

We can substitute $\int_{X_{2,s}}\sigma_{2,s}(x_2)\cdot(x_2)^j dx_2$ with the moment $\mu_{2,s,j}$ of the j–th order of $\sigma_{2,s}$, obtaining:

$$\begin{aligned}u_s-\sum_{k=0}^{m}\sum_{j=0}^{m}r_{s,k,j}\cdot(x_1)^k\cdot\mu_{2,s,j}\\ -\gamma\sum_{s'\in S}u_{s'}\cdot\sum_{k=0}^{m}p_{s,s',k}\cdot(x_1)^k\geq 0 \qquad \forall s\in S_1, x_1\in X_{1,s}\end{aligned}$$

Call $\boldsymbol{\mu}_{2,s}$ the vector of $\mu_{2,s,j}$ with $j\in\{0,\ldots,m\}$ (higher order moments are not constrained). PS_1 is:

$$\textbf{(PS}_1\textbf{)}\quad \min \quad \textstyle\sum_{s\in S_1}u_s$$

$$\begin{array}{lll}u_s-\sum_{k=0}^{m}\sum_{j=0}^{m}r_{s,k,j}\cdot(x_1)^k\cdot & & \forall s\in S_1\\ \cdot\mu_{2,s,j}-\gamma\sum_{s'\in S}u_{s'}\cdot & & \\ \cdot\sum_{k=0}^{m}p_{s,s',k}\cdot(x_1)^k & \in\mathcal{P}^+(x_1\in X_{1,s}) & \\ \\ \boldsymbol{\mu}_{2,s} & \in\mathcal{M}(X_{2,s}) & \forall s\in S_1\end{array}$$

where $\mathcal{P}^+(x_1\in X_{1,s})$ is the space of univariate polynomials in x_1 that are non–negative on $X_{1,s}$ and $\mathcal{M}(X_{1,s})$

is the space of the moment vectors of well defined probability measures on $X_{1,s}$. Both these constraints (i.e., non–negativeness of univariate polynomials and correspondence of moment vectors to well defined probability measures) can be coded as semidefinite programming constraints. Call:

$$\begin{aligned}
\mathbf{u} &= [u_1 \ u_2 \ \dots \ u_{|S|}]^T \\
[x]_m &= [(x)^0 \ (x)^1 \ (x)^2 \ \dots \ (x)^m]^T \\
R_s^T \cdot [x]_m &= \sum_{k=0}^{m} r_{s,k} \cdot (x)^k \\
\mathbf{u}^T \cdot P_{1,s}^T \cdot [x]_m &= \sum_{s' \in S_1} u_{s'} \cdot \sum_{k=0}^{m} p_{s,s',k} \cdot (x)^k \\
\hat{\mathbf{u}}^T \cdot \hat{P}_{1,s}^T \cdot [x]_m &= \sum_{s' \in S_2} \hat{u}_{s'} \cdot \sum_{k=0}^{m} \hat{p}_{s,s',k} \cdot (x)^k
\end{aligned}$$

Call $\mathcal{H}$ the following operator returning an Hankel matrix:

$$\mathcal{H}: \begin{bmatrix} a_1 \\ a_2 \\ a_3 \\ \vdots \\ a_{2n-1} \end{bmatrix} \longrightarrow \begin{bmatrix} a_1 & a_2 & \dots & a_n \\ a_2 & a_3 & \dots & a_{n+1} \\ \vdots & \vdots & \ddots & \vdots \\ a_n & a_{n+1} & \dots & a_{2n-1} \end{bmatrix}$$

and $\mathcal{H}^*$ the following adjoint operator:

$$\mathcal{H}^*: \begin{bmatrix} a_{1,1} & a_{1,2} & \dots & a_{1,n} \\ a_{1,2} & a_{2,2} & \dots & a_{2,n} \\ \vdots & \vdots & \ddots & \vdots \\ a_{1,n} & a_{2,n} & \dots & a_{n,n} \end{bmatrix} \longrightarrow \begin{bmatrix} a_{1,1} \\ 2a_{1,2} \\ a_{2,2} + 2a_{1,3} \\ \vdots \\ a_{n,n} \end{bmatrix}$$

Define the following matrices as:

$$L_1 = \begin{bmatrix} I_{m \times m} \\ \mathbf{0}_{1 \times m} \end{bmatrix}, L_2 = \begin{bmatrix} \mathbf{0}_{1 \times m} \\ I_{m \times m} \end{bmatrix}$$

When $X_{1,s} = [0,1]$ with $s \in S$, PS_1 can be written as:

$$\min \textstyle\sum_{s \in S_1} u_s$$

$$\begin{aligned}
&\mathcal{H}^*(Z_s + \frac{1}{2}(L_1 W_s L_2^T + L_2 W_s L_1^T) \\
&\quad -L_2 W_s L_2^T) - u_s - R_s \boldsymbol{\mu}_{i,s} && = 0 \quad \forall s \in S_1 \quad (5) \\
&\quad -\gamma(P_{1,s}\mathbf{u} + \hat{P}_{1,s}\hat{\mathbf{u}}) \\
&Z_s, W_s && \succeq 0 \quad \forall s \in S_1 \quad (6) \\
&\mathcal{H}(\boldsymbol{\mu}_{i,s}) && \succeq 0 \quad \forall s \in S_1 \quad (7) \\
&\frac{1}{2}(L_1^T \mathcal{H}(\boldsymbol{\mu}_{i,s}) L_2 + L_2^T \mathcal{H}(\boldsymbol{\mu}_{i,s}) L_1) && \succeq 0 \quad \forall s \in S_1 \quad (8) \\
&\quad -L_2^T \mathcal{H}(\boldsymbol{\mu}_{i,s}) L_2 \\
&\mu_{i,s,0} && = 1 \quad \forall s \in S_1 \quad (9) \\
&\mathcal{H}(\boldsymbol{\mu}'_{i,s}) && \succeq 0 \quad \forall s \in S_1 \quad (10) \\
&\mathcal{H}(\boldsymbol{\mu}_{i,s}) - \mathcal{H}(\boldsymbol{\mu}'_{i,s}) && \succeq 0 \quad \forall s \in S_1 \quad (11)
\end{aligned}$$

where $\boldsymbol{\mu}'_{i,s} = [\mu_{i,s,1} \ \mu_{i,s,2} \ \dots \ \mu_{i,s,m+1}]$, W_s and Z_s are matrices of auxiliary variables, and "$\succeq 0$" means semidefinite positive. Constraints (5) and (6) translate constraint (1) in SDP fashion; constraints (7)–(9) translate constraint (2); constraints (10) and (11) are accessory for the resolution of PS_1, but necessary for finding well defined $\mu_{i,s,m+1}$ that are needed for the strategy recovery as described in the following section. The proof that strong duality holds with this SDP easily follows from the satisfaction of Slater's constraint qualification and that it is bounded from below, a similar proof can be found in (Shah and Parrilo 2007).

Strategy recovery

The SDP programs discussed in the previous sections return strategies defined in the space of moments. In order to recovery the strategy in the space of actions we can use the methods discussed in (Karlin and Shapley 1953; Schmeiser and Devroye 1988; Shohat and Tamarkin 1943). Since the number of moments of $\mu_{i,s,h}$ of $\sigma_{i,s}$ is finite, we can always define a finite support $\Psi_{i,s}$, where $x_{i,s,h}$ is the h-th value of $\Psi_{i,s}$. For the sake of presentation, in the following we will omit indices i, s from $\mu_{i,s,h}$, $x_{i,s,h}$, and $\sigma_{i,s}$.

The strategy can be recovered by, at first, solving the following linear equation system:

$$\begin{bmatrix} \mu_0 & \mu_1 & \cdots & \mu_{\frac{m}{2}+1} \\ \mu_1 & \mu_2 & \cdots & \mu_m \\ \vdots & \vdots & \ddots & \vdots \\ \mu_{\frac{m}{2}+1} & \mu_{\frac{m}{2}+2} & \cdots & \mu_m \end{bmatrix} \cdot \begin{bmatrix} b_0 \\ b_1 \\ \vdots \\ b_{m-1} \end{bmatrix} = - \begin{bmatrix} \mu_{\frac{m}{2}+1} \\ \mu_{\frac{m}{2}+2} \\ \vdots \\ \mu_{m+1} \end{bmatrix}$$

and finding the vector of coefficients b_j. Notice that the above (Hankel) matrix of the moments is semidefinite positive due to constraint (7) and therefore the solution of the above linear system is unique. The actions in the support are the roots x_h of the following univariate polynomial:

$$x^m + b_{m-1} \cdot x^{m-1} + \dots + b_1 \cdot x + b_0 = 0.$$

The probabilities $\sigma(x_h)$ associated with actions x_h can be found by solving the non–singular system of Vandermonde:

$$\textstyle\sum_{h=1}^{m} \sigma(x_h) \cdot x_h^j = \mu_j \quad 0 \le j \le m-1$$

Best response computation

Given strategy $\overline{\boldsymbol{\sigma}}_2$, the problem BR_1 to find the agent 1's best response is:

$$\min \textstyle\sum_{s \in S} u_s$$

$$\begin{aligned}
&u_s - \textstyle\sum_{h \in \Psi_{2,s}} \overline{\sigma}_{2,s,h} \sum_{k=0}^{m} && \forall s \in S_1 \\
&\quad \cdot \sum_{j=0}^{m} r_{s,k,j} \cdot (x_1)^k \cdot (\overline{x}_{2,s,h})^j - \\
&\quad -\gamma \textstyle\sum_{s' \in S} u'_s \cdot \sum_{k=0}^{m} p_{s,s',k} \cdot (x_1)^k && \in \mathcal{P}^+(X_{1,s}) \\
&u_s - \textstyle\sum_{h \in \Psi_{2,s}} \overline{\sigma}_{2,s,h} \sum_{k=0}^{m} \sum_{j=0}^{m} r_{s,k,j} && \forall s \in S_2 \\
&\quad \cdot (x_1)^k \cdot (\overline{x}_{2,s,h})^j - \gamma \textstyle\sum_{h \in \Phi_{2,s}} \overline{\sigma}_{2,s,h} \\
&\quad \cdot \textstyle\sum_{s' \in S} u'_s \sum_{k=0}^{m} p_{s,s',k} \cdot (\overline{x}_{2,s,h})^k && \in \mathcal{P}^+(X_{2,s})
\end{aligned}$$

Call:

$$\overline{R}_s^T \cdot [x_s]_{d_s} = \sum_{h \in \Psi_{2,s}} \overline{\sigma}_{2,s,h} \sum_{k=0}^{m} \cdot \sum_{j=0}^{m} r_{s,j,k} \cdot (\overline{x}_{2,s,h})^j \cdot (x_1)^k$$

$$\mathbf{u}^T \cdot \overline{P}_s^T \cdot [x_s]_{d_s} = \sum_{h \in \Psi_{2,s}} \overline{\sigma}_{2,s,h} \sum_{s' \in S} u'_s \sum_{k=0}^{m} p_{s,s',j} \cdot (\overline{x}_{2,s,h})^k \cdot (x_1)^0$$

when $X_s = [0,1]$ the above mathematical program can be formulated as the following SDP:

$$\min \textstyle\sum_{s \in S} u_s$$

$$\mathcal{H}^*(Z_s + \tfrac{1}{2}(L_1 W_s L_2^T + L_2 W_s L_2^T) - \\ -L_2 W_s L_2^T) - u_s - \overline{R}_s - \gamma P_s \mathbf{u} = 0 \quad \forall s \in S_1 \quad (12)$$

$$\mathcal{H}^*(Z_s + \tfrac{1}{2}(L_1 W_s L_2^T + L_2 W_s L_2^T) - \\ -L_2 W_s L_2^T) - u_s - \overline{R}_s - \gamma \overline{P}_s \mathbf{u} = 0 \quad \forall s \in S_2 \quad (13)$$

$$W_s, Z_s \succeq 0 \quad \forall s \in S \quad (14)$$

constraints (12) translate constraints (3), while constraints (13) translate constraints (4); W_s and Z_s are matrices of auxiliary variables. BR_2 can be similarly defined.

Equilibrium approximation with non–polynomial games

In case of non–polynomial games, we approximate reward and state transitions with polynomial functions and then we apply Algorithm 1. If maximum approximation errors are small we can expect to find solutions that produces nearly–optimal outcomes. Notice that, it is known from approximation theory that continuous functions defined over compact subsets of a d–dimensional Euclidean space can be approximated to arbitrary accuracy with a degree–n polynomial (Cheney 1982). In this section, we provide theoretical bounds on how the equilibrium approximation accuracy is affected by using reward and state transition functions that are polynomial approximations of the original ones.

For the sake of presentation, in the following we use the contract form $f_s^{\boldsymbol{\sigma}_1,\boldsymbol{\sigma}_2}$ to denote the expected value of a function f in state s with respect to the strategy profile $(\boldsymbol{\sigma}_1, \boldsymbol{\sigma}_2)$:

$$f_s^{\boldsymbol{\sigma}_1,\boldsymbol{\sigma}_2} = \mathbb{E}_{\substack{x_1 \sim \boldsymbol{\sigma}_{1,s} \\ x_2 \sim \boldsymbol{\sigma}_{2,s}}} f_s(x1, x2)$$

Given a stochastic game with reward function $r_\cdot(\cdot,\cdot)$ and state transition probabilities $p_{\cdot,\cdot}(\cdot)$, we consider their polynomial approximations $\tilde{r}_\cdot(\cdot,\cdot)$ and $\tilde{p}_{\cdot,\cdot}(\cdot)$ with the following maximum error bounds:[1]

$$\|r_s(\cdot,\cdot) - \tilde{r}_s(\cdot,\cdot)\|_\infty \leq \delta \qquad \forall s \in S \qquad (15)$$

$$\|p_{s,s'}(\cdot) - \tilde{p}_{s,s'}(\cdot)\|_\infty \leq \rho \qquad \forall s, s' \in S \quad (16)$$

[1] Given an arbitrary univariate function p_s, it is possible to find the best m–degree polynomial approximation $\tilde{p}_s$ (i.e., the m–degree polynomial minimizing ρ) with (Remez 1935). The optimal approximation of r_s is harder, it being a bivariate function. In this case, the algorithm presented in (Caliari, de Marchi, and Vianello 2008) can be used to find a good polynomial approximation $\tilde{r}_s$ of r_s.

We define an ϵ_s–MPE as a strategy profile such that no agent can gain more than ϵ in state s by deviating from her strategy, while in the other state utilities can be arbitrary. We state the following theoretical results. The first result is similar to (Chen, Deng, and Teng 2006), but stronger.

Theorem 2. *Given game $\mathcal{G}$ and absorbing state $\underline{s}$, the MPE strategy profile $(\tilde{\boldsymbol{\sigma}}_1, \tilde{\boldsymbol{\sigma}}_2)$ when agents' utility functions are $\tilde{u}_{\underline{s}}(\cdot,\cdot)$ and $-\tilde{u}_{\underline{s}}(\cdot,\cdot)$, respectively, is an $\epsilon_{\underline{s}}$–MPE with $\epsilon_{\underline{s}} \leq 2\delta$ when agents' utility functions are $u_{\underline{s}}(\cdot,\cdot)$ and $-u_{\underline{s}}(\cdot,\cdot)$.*

Proof. Call $\boldsymbol{\sigma}_1^*$ the best response of agent 1 to the $\boldsymbol{\sigma}_2$ of agent 2. We can compute the upper bound over the expected utility loss of agent 1 (recalling that, by definition, for every $\boldsymbol{\sigma}_1$ we have $\tilde{u}_{\underline{s}}^{\tilde{\boldsymbol{\sigma}}_1,\tilde{\boldsymbol{\sigma}}_2} - \tilde{u}_{\underline{s}}^{\boldsymbol{\sigma}_1,\tilde{\boldsymbol{\sigma}}_2} \geq 0$):

$$\begin{aligned}
\epsilon_{\underline{s}} &= u_{\underline{s}}^{\sigma_1^*,\tilde{\boldsymbol{\sigma}}_2} - u_{\underline{s}}^{\tilde{\boldsymbol{\sigma}}_1,\tilde{\boldsymbol{\sigma}}_2} = r_{\underline{s}}^{\sigma_1^*,\tilde{\boldsymbol{\sigma}}_2} - r_{\underline{s}}^{\tilde{\boldsymbol{\sigma}}_1,\tilde{\boldsymbol{\sigma}}_2} \\
&\leq r_{\underline{s}}^{\sigma_1^*,\tilde{\boldsymbol{\sigma}}_2} - \tilde{r}_{\underline{s}}^{\sigma_1^*,\tilde{\boldsymbol{\sigma}}_2} + \tilde{r}_{\underline{s}}^{\tilde{\boldsymbol{\sigma}}_1,\tilde{\boldsymbol{\sigma}}_2} - r_{\underline{s}}^{\tilde{\boldsymbol{\sigma}}_1,\tilde{\boldsymbol{\sigma}}_2} \\
&\leq |r_{\underline{s}}^{\sigma_1^*,\tilde{\boldsymbol{\sigma}}_2} - \tilde{r}_{\underline{s}}^{\sigma_1^*,\tilde{\boldsymbol{\sigma}}_2}| + |\tilde{r}_{\underline{s}}^{\tilde{\boldsymbol{\sigma}}_1,\tilde{\boldsymbol{\sigma}}_2} - r_{\underline{s}}^{\tilde{\boldsymbol{\sigma}}_1,\tilde{\boldsymbol{\sigma}}_2}| \\
&\leq \|r_{\underline{s}}(\cdot,\cdot) - \tilde{r}_{\underline{s}}(\cdot,\cdot)\|_\infty + \|\tilde{r}_{\underline{s}}(\cdot,\cdot) - r_{\underline{s}}(\cdot,\cdot)\|_\infty \\
&\leq 2\delta
\end{aligned}$$

The same reasoning can be applied to agent 2, obtaining the same upper bound. Hence, the theorem is proved. □

We generalize the above theorem for generic states when $\rho = 0$. Given state s, call $l(s)$ the largest number of actions needed to reach an absorbing state from s. We first introduce the following lemma which bounds the difference between the two utility functions $u_s^{\boldsymbol{\sigma}_1,\boldsymbol{\sigma}_2}$ and $\tilde{u}_s^{\boldsymbol{\sigma}_1,\boldsymbol{\sigma}_2}$ for any state s and any strategy profile $(\boldsymbol{\sigma}_1, \boldsymbol{\sigma}_2)$.

Lemma 3. *(With $\rho = 0$.) Given a tree–based game $\mathcal{G}$, for any state s and for any strategy profile $(\boldsymbol{\sigma}_1, \boldsymbol{\sigma}_2)$, the maximum absolute difference between utility of agent 1 computed according to $u(\cdot,\cdot)$ and utility of agent 1 computed according to $\tilde{u}(\cdot,\cdot)$ is less than $\delta\frac{1-\gamma^{l(s)+1}}{1-\gamma}$: $\|u_s^{\boldsymbol{\sigma}_1,\boldsymbol{\sigma}_2} - \tilde{u}_s^{\boldsymbol{\sigma}_1,\boldsymbol{\sigma}_2}\|_\infty \leq \delta\frac{1-\gamma^{l(s)+1}}{1-\gamma}$.*

Proof. Given the bound on the reward functions in Eq. (15) we can write:

$$\begin{aligned}
\|u_s^{\boldsymbol{\sigma}_1,\boldsymbol{\sigma}_2} - \tilde{u}_s^{\boldsymbol{\sigma}_1,\boldsymbol{\sigma}_2}\|_\infty &\leq \|r_s^{\boldsymbol{\sigma}_1,\boldsymbol{\sigma}_2} - \tilde{r}_s^{\boldsymbol{\sigma}_1,\boldsymbol{\sigma}_2}\|_\infty \\
&\quad + \gamma \| \sum_{s'} p_{s,s'}^{\boldsymbol{\sigma}_c} \left(u_{s'}^{\boldsymbol{\sigma}_1,\boldsymbol{\sigma}_2} - \tilde{u}_{s'}^{\boldsymbol{\sigma}_1,\boldsymbol{\sigma}_2} \right) \|_\infty \\
&\leq \delta + \gamma \max_{s'} \|u_{s'}^{\boldsymbol{\sigma}_1,\boldsymbol{\sigma}_2} - \tilde{u}_{s'}^{\boldsymbol{\sigma}_1,\boldsymbol{\sigma}_2}\|_\infty
\end{aligned}$$

By the recursive application of the above formula starting from an absorbing state $\underline{s}$ up to state s, we obtain: $\|u_s^{\boldsymbol{\sigma}_1,\boldsymbol{\sigma}_2} - \tilde{u}_s^{\boldsymbol{\sigma}_1,\boldsymbol{\sigma}_2}\|_\infty \leq \delta \sum_{i=0}^{l(s)} (\gamma)^i = \delta\frac{1-(\gamma)^{l(s)+1}}{1-\gamma}$. □

Now, we are ready to extend Theorem 2 to the case of tree–based games.

Theorem 4. *(With $\rho = 0$.) Given a tree–based game $\mathcal{G}$ and a generic state s, the MPE strategy profile $(\tilde{\boldsymbol{\sigma}}_1, \tilde{\boldsymbol{\sigma}}_2)$ of the subgame whose root node is s when agents' utility functions are $\tilde{u}(\cdot,\cdot)$ and $-\tilde{u}(\cdot,\cdot)$, respectively, is an ϵ_s–MPE with $\epsilon_s \leq 2\delta\frac{1-\gamma^{l(s)+1}}{1-\gamma}$ of such subgame when agents' utility functions are $u(\cdot,\cdot)$ and $-u(\cdot,\cdot)$.*

Proof. For every state s' in the subgame whose root node s, call $\boldsymbol{\sigma}_2^*(\boldsymbol{\sigma}_2)$ the best strategy of agent 1 found by backward induction when agent 2 plays $\boldsymbol{\sigma}_2$. By following the same line of Theorem 2's proof, we can compute the upper bound over the expected utility loss of agent 1 at s as a function of the difference of the two utility functions $u_s(\cdot,\cdot)$ and $\tilde{u}_s(\cdot,\cdot)$, which have been bounded in Lemma 3:

$$\begin{aligned}\epsilon_s &= u_s^{\boldsymbol{\sigma}_1^*,\tilde{\boldsymbol{\sigma}}_2} - u_s^{\tilde{\boldsymbol{\sigma}}_1,\tilde{\boldsymbol{\sigma}}_2} \\ &\leq u_s^{\boldsymbol{\sigma}_1^*,\tilde{\boldsymbol{\sigma}}_2} - \tilde{u}_s^{\boldsymbol{\sigma}_1^*,\tilde{\boldsymbol{\sigma}}_2} + \tilde{u}_s^{\tilde{\boldsymbol{\sigma}}_1,\tilde{\boldsymbol{\sigma}}_2} - u_s^{\tilde{\boldsymbol{\sigma}}_1,\tilde{\boldsymbol{\sigma}}_2} \\ &\leq 2\|u_s(\cdot,\cdot) - \tilde{u}_s(\cdot,\cdot)\|_\infty \\ &\leq 2\delta\frac{1-\gamma^{l(s)+1}}{1-\gamma}\end{aligned}$$

The same result can be obtained for the loss of agent 2. □

Now, we focus on graph–based games.

Corollary 5 *(With $\rho = 0$.) The bound stated in Theorem 4 can be generalized to graph–based games $\mathcal{G}$ as follows:*

$$\epsilon \leq \lim_{l(s)\to+\infty} 2\delta\frac{1-\gamma^{l(s)+1}}{1-\gamma} = \frac{2\delta}{1-\gamma}$$

Notice that the above bound is state independent and is meaningful (i.e., $\epsilon < 1$) only when $2\delta + \gamma < 1$.

Call $\Phi(s) = \sum_{s'\in \text{FH}(s)} \tilde{u}_{s'}^{\tilde{\boldsymbol{\sigma}}_1,\tilde{\boldsymbol{\sigma}}_1} - \sum_{s'\in \text{LH}(s)} \tilde{u}_{s'}^{\tilde{\boldsymbol{\sigma}}_1,\tilde{\boldsymbol{\sigma}}_1}$ where sets $\text{FH}(s)$ and $\text{LH}(s)$ are defined as follows. Call q_s the number of states s' reachable from s with a single action. $\text{FH}(s)$ contains the $\lfloor q_s/2 \rfloor$ states with the highest $\tilde{u}_{s'}^{\tilde{\boldsymbol{\sigma}}_1,\tilde{\boldsymbol{\sigma}}_1}$, while $\text{LH}(s)$ contains the $\lfloor q_s/2 \rfloor$ states with the lowest $\tilde{u}_{s'}^{\tilde{\boldsymbol{\sigma}}_1,\tilde{\boldsymbol{\sigma}}_1}$. Call $\overline{\Phi} = \max_s\{\Phi(s)\}$.

Similarly to what has been done in Lemma 3, in the following we bound the difference between the two utility functions $u_s^{\boldsymbol{\sigma}_1,\boldsymbol{\sigma}_2}$ and $\tilde{u}_s^{\boldsymbol{\sigma}_1,\boldsymbol{\sigma}_2}$ for any state s and any strategy profile $(\boldsymbol{\sigma}_1,\boldsymbol{\sigma}_2)$ when $\delta = 0$ and $\rho > 0$.

Lemma 6. *(With $\delta = 0$.) Given a tree–based game $\mathcal{G}$, for any state s and for any strategy profile $(\boldsymbol{\sigma}_1,\boldsymbol{\sigma}_2)$, the maximum absolute difference between utility of agent 1 computed according to $u(\cdot,\cdot)$ and utility of agent 1 computed according to $\tilde{u}(\cdot,\cdot)$ is less than $\delta\frac{1-\gamma^{l(s)+1}}{1-\gamma}$: $\|u_s^{\boldsymbol{\sigma}_1,\boldsymbol{\sigma}_2} - \tilde{u}_s^{\boldsymbol{\sigma}_1,\boldsymbol{\sigma}_2}\|_\infty \leq \delta\frac{1-\gamma^{l(s)+1}}{1-\gamma}$.*

Proof. Given the bound on the transition probability functions in Eq. (16), we obtain:

$$\begin{aligned}\|u_s^{\boldsymbol{\sigma}_1,\boldsymbol{\sigma}_2} - \tilde{u}_s^{\boldsymbol{\sigma}_1,\boldsymbol{\sigma}_2}\|_\infty &\leq \|r_s^{\boldsymbol{\sigma}_1,\boldsymbol{\sigma}_2} - \tilde{r}_s^{\boldsymbol{\sigma}_1,\boldsymbol{\sigma}_2}\|_\infty \\ &\quad +\gamma\|\sum_{s'}\left(p_{s,s'}^{\boldsymbol{\sigma}_c} u_{s'}^{\boldsymbol{\sigma}_1,\boldsymbol{\sigma}_2} - \tilde{p}_{s,s'}^{\boldsymbol{\sigma}_c}\tilde{u}_{s'}^{\boldsymbol{\sigma}_1,\boldsymbol{\sigma}_2}\right)\|_\infty \\ &= \gamma\|\sum_{s'}\left(p_{s,s'}^{\boldsymbol{\sigma}_c} u_{s'}^{\boldsymbol{\sigma}_1,\boldsymbol{\sigma}_2} - p_{s,s'}^{\boldsymbol{\sigma}_c}\tilde{u}_{s'}^{\boldsymbol{\sigma}_1,\boldsymbol{\sigma}_2}\right) \\ &\quad +\sum_{s'}\left(p_{s,s'}^{\boldsymbol{\sigma}_c}\tilde{u}_{s'}^{\boldsymbol{\sigma}_1,\boldsymbol{\sigma}_2} - \tilde{p}_{s,s'}^{\boldsymbol{\sigma}_c}\tilde{u}_{s'}^{\boldsymbol{\sigma}_1,\boldsymbol{\sigma}_2}\right)\|_\infty \\ &\leq \gamma\|\sum_{s'} p_{s,s'}^{\boldsymbol{\sigma}_c}\left(u_{s'}^{\boldsymbol{\sigma}_1,\boldsymbol{\sigma}_2} - \tilde{u}_{s'}^{\boldsymbol{\sigma}_1,\boldsymbol{\sigma}_2}\right)\|_\infty \\ &\quad +\gamma\|\sum_{s'}\left(p_{s,s'}^{\boldsymbol{\sigma}_c} - \tilde{p}_{s,s'}^{\boldsymbol{\sigma}_c}\right)\tilde{u}_{s'}^{\boldsymbol{\sigma}_1,\boldsymbol{\sigma}_2}\|_\infty \\ &\leq \gamma\max_{s'}\|u_{s'}^{\boldsymbol{\sigma}_1,\boldsymbol{\sigma}_2} - \tilde{u}_{s'}^{\boldsymbol{\sigma}_1,\boldsymbol{\sigma}_2}\|_\infty + \gamma\rho\Phi(s) \\ &\leq \gamma\rho\overline{\Phi} + \gamma\max_{s'}\|u_{s'}^{\boldsymbol{\sigma}_1,\boldsymbol{\sigma}_2} - \tilde{u}_{s'}^{\boldsymbol{\sigma}_1,\boldsymbol{\sigma}_2}\|_\infty\end{aligned}$$

By the recursive application of the above formula starting from an absorbing state $\underline{s}$ up to state s, we obtain: $\|u_s^{\boldsymbol{\sigma}_1,\boldsymbol{\sigma}_2} - \tilde{u}_s^{\boldsymbol{\sigma}_1,\boldsymbol{\sigma}_2}\|_\infty \leq \gamma\rho\overline{\Phi}\frac{1-(\gamma)^{l(s)+1}}{1-\gamma}$. □

Theorem 7. *(With $\delta = 0$.) Given a tree–based game $\mathcal{G}$ and a generic state s, the MPE strategy profile $(\tilde{\boldsymbol{\sigma}}_1,\tilde{\boldsymbol{\sigma}}_2)$ of the subgame whose root node is s when agents' utility functions are $\tilde{u}(\cdot,\cdot)$ and $-\tilde{u}(\cdot,\cdot)$, respectively, is an ϵ_s–MPE with $\epsilon_s \leq 2\gamma\rho\overline{\Phi}\frac{1-\gamma^{l(s)+1}}{1-\gamma}$ of such subgame when agents' utility functions are $u(\cdot,\cdot)$ and $-u(\cdot,\cdot)$.*

Proof. By Theorem 2, we compute the upper bound over the expected utility loss of agent 1 at s as a function of the loss of agent 1 in the states directly reachable from s:

$$\begin{aligned}\epsilon_s &= u_s^{\boldsymbol{\sigma}_1^*,\tilde{\boldsymbol{\sigma}}_2} - u_s^{\tilde{\boldsymbol{\sigma}}_1,\tilde{\boldsymbol{\sigma}}_2} \\ &\leq 2\|u_s(\cdot,\cdot) - \tilde{u}_s(\cdot,\cdot)\|_\infty \\ &\leq 2\gamma\rho\overline{\Phi}\frac{1-(\gamma)^{l(s)+1}}{1-\gamma}\end{aligned}$$

The same result can be obtained for the loss of agent 2. □

Now, we focus on graph–based games.

Corollary 8. *(With $\delta = 0$.) The bound stated in Theorem 7 can be generalized to graph–based games $\mathcal{G}$ as follows:*

$$\epsilon_s \leq \lim_{l(s)\to+\infty} 2\gamma\rho\overline{\Phi}\frac{1-(\gamma)^{l(s)+1}}{1-\gamma} = \frac{2\gamma\rho\overline{\Phi}}{1-\gamma}$$

As previously, the above bound is state independent and is significative (i.e., $\epsilon < 1$) only when $\gamma(1 + 2\rho\overline{\Phi}) < 1$.

Taken together, the results exposed above allow us to state the bound for the general case, in which $\delta, \rho \geq 0$:

Theorem 9. *Given a tree–based game $\mathcal{G}$ and a generic state s, the MPE strategy profile $(\tilde{\boldsymbol{\sigma}}_1,\tilde{\boldsymbol{\sigma}}_2)$ of the subgame whose root node is s when agents' utility functions are $\tilde{u}(\cdot,\cdot)$ and $-\tilde{u}(\cdot,\cdot)$, respectively, is an ϵ_s–MPE with $\epsilon_s \leq 2(\delta + \gamma\rho\overline{\Phi})\frac{1-(\gamma)^{l(s)+1}}{1-\gamma}$ of such subgame when agents' utility functions are $u(\cdot,\cdot)$ and $-u(\cdot,\cdot)$.*

The proof is easy, the two bounds being additive in our problem. Similarly, we obtain:

Corollary 10. *The bound stated in Theorem 9 can be generalized to graph–based games $\mathcal{G}$ as follows:*

$$\epsilon_s \leq \lim_{l(s)\to+\infty} 2(\delta + \gamma\rho\overline{\Phi})\frac{1-(\gamma)^{l(s)+1}}{1-\gamma} = \frac{2(\delta + \gamma\rho\overline{\Phi})}{1-\gamma}$$

Notice that this bound is meaningful (i.e., < 1) only when $2\delta + \gamma(1 + 2\rho\overline{\Phi}) < 1$.

Finally, we collect all the above results and we consider the situation in which an $\tilde{\epsilon}$–MPE computed on the approximated game $\tilde{\mathcal{G}}$ is used in the original game $\mathcal{G}$.

Theorem 11. *Given a game $\mathcal{G}$ and a generic state s, an $\tilde{\epsilon}$–MPE strategy profile $(\tilde{\boldsymbol{\sigma}}_1,\tilde{\boldsymbol{\sigma}}_2)$ of the approximated game $\tilde{\mathcal{G}}$, is an ϵ^*–MPE for game $\mathcal{G}$, with $\epsilon^* \leq \max_s\{\epsilon_s\} + \tilde{\epsilon}$.*

Again, the proof stems from the additivity of the two bounds and the definition of ϵ–MPE.

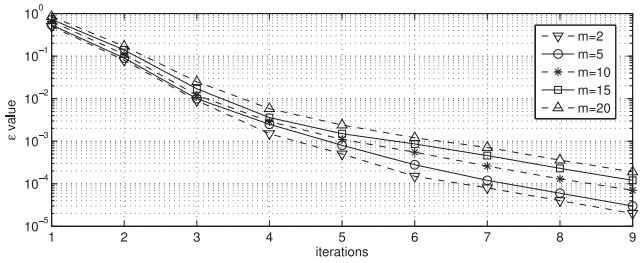

Figure 1: Average ϵ value.

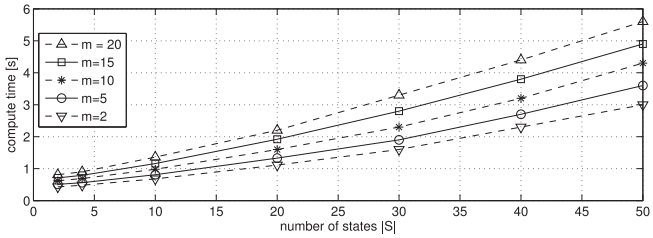

Figure 2: Average compute time per single SDP.

Experimental evaluation

We implemented our algorithm with Matlab 7.12 calling Yalmip R20120109 (Lofberg 2004) and SDPT3 v.4 (K.C. Toh and Tutuncu 1999) to solve SDPs. We conducted the experiments on a UNIX computer with dual quadcore 2.33GHz CPUs and 16GB RAM.

We preliminarily evaluated the efficiency of our algorithm with medium/small polynomial games. We randomly generated 10 game instances with $|S| \in \{2, 4, 10, 20, 30, 40, 50\}$ and $m \in \{2, 5, 10, 15, 20\}$ as follows. Without loss of generality, $r_{s,k,j}$ has been uniformly drawn from $[-1, 1]$ except $r_{i,0,0}$ that is set equal to $r_{i,0,0} = m \cdot (m+1)/2$ to guarantee that the reward functions are always positive on $[0, 1]\times[0, 1]$. We generated $p_{s,s',i}$ by exploiting SDP programming: we formulated the space of feasible polynomials ($p_{s,s',i} \geq 0$, $\sum_{s'} p_{s,s',i} = 1$) as an SDP and we randomly selected a feasible point of such space. The reward functions have been normalized such that $\max_s\{u_s\} = 1$ and $\min_s\{u_s\} = 0$.

We applied our algorithm to the above experimental setting. Fig. 1 shows how ϵ varies with the number of iterations (the plot is semi–log). Fixed the iteration, for each value of m we report the value of ϵ averaged over the instances with all the different $|S|$. At every iteration, ϵ is monotonically decreasing with m and the ratio between ϵ with $m = 20$ (max used degree) and ϵ with $m = 2$ (min used degree) is always less than 10 and hence the performances with different m are close. ϵ decreases exponentially with the number of iterations and gets very small — in $[10^{-5}, 10^{-4}]$ — even after few iterations. Then, we evaluated the computation time. About 98% of the compute time per iteration is required by the resolution of the SDPs (PS_i, DS_i, and BR_i require approximately the same compute time). In Fig. 2 we report how the average compute time needed by Yalmip and SDPT3 to solve a single SDP (precisely, PS_1) varies with $|S|$ for different m. The average compute time remains short even with $|S| = 50$ and $m = 20$, and, therefore, with small/medium instances, the algorithm scales very well finding ϵ–MPEs with very small ϵ by short compute time.

Finally, we preliminarily evaluated the effectiveness of our theoretical bounds for non–polynomial games by evaluating, with Caliari et al. (2008), the approximation error δ with infinity norm for different classes of bivariate functions (their approximation is harder than that of univariate functions). In Tab. 1, we report, for different m, the average δ with three different function classes (exp, sin, linear piecewise) by generating randomly 30 instances per class (10 per n). $\delta \approx 10^{-5}$ for every n when $m = 20$ with smooth functions (exp/sin), while $\delta \geq 10^{-2}$ with continuous but not differentiable functions (piecewise). Anyway, also with these last functions the error and hence the upper bound on ϵ is reasonably small.

Conclusions

We studied the problem to find and approximate an MPE with continuous two–player zero–sum stochastic games with switching control. We provided an algorithm based on SDP that converges to an MDP when the game is polynomial. When instead the game is non–polynomial, we approximate the reward and the transition functions minimizing the error with infinity norm and then we apply our algorithm. In this case, we provide theoretical guarantees over the value of ϵ of the ϵ–MPE to which the algorithm converges. Finally, we experimentally evaluated our algorithm, showing that with small/medium games it is efficient and that the approximation error with infinity norm is small.

As future works, we aim at studying the scalability of the algorithm with large instances, evaluating our theoretical bounds with a variety of function classes, improving them, and extending the algorithm to solve general–sum games. In addition, we will remove the assumption of switching controller and we will explore both verification (Gatti and Panozzo 2012) and computation (Gatti and Iuliano 2011) of perfection–based solution concepts with continuous actions.

References

Blekherman, G.; Parrilo, P. A.; and Thomas, R. 2012. *Polynomial optimization, sums of squares and applications*. SIAM.

Caliari, M.; de Marchi, S.; and Vianello, M. 2008. Padua2d: Lagrange interpolation at padua points on bivariate domains. *ACM TRANS MATH SOFTWARE* 35(3):1–11.

Chen, X.; Deng, X.; and Teng, S.-h. 2006. Computing Nash Equilibria: Approximation and Smoothed Complexity. *FOCS* 603–612.

Chen, X.; Deng, X.; and Teng, S.-H. 2009. Settling the complexity of computing two–player Nash equilibria. *J ACM* 56(3):14:1–14:57.

Cheney, E. 1982. *Introduction to approximation theory*. AMER MATH SOC.

class	n	polynomial degree (m)				
		2	5	10	15	20
$\sum_{j=1}^{n} \alpha_j \cdot \exp(\sum_{i=1}^{2} \beta_{i,j} \cdot (x_i - \gamma_{i,j}))$	1	$1.86{\cdot}10^{-1}$	$1.65{\cdot}10^{-2}$	$1.34{\cdot}10^{-5}$	$1.80{\cdot}10^{-5}$	$1.05{\cdot}10^{-5}$
	5	$2.83{\cdot}10^{-1}$	$0.59{\cdot}10^{-2}$	$2.03{\cdot}10^{-4}$	$6.41{\cdot}10^{-5}$	$1.31{\cdot}10^{-4}$
	10	$2.95{\cdot}10^{-1}$	$1.60{\cdot}10^{-2}$	$1.69{\cdot}10^{-4}$	$4.52{\cdot}10^{-5}$	$6.01{\cdot}10^{-5}$
$\sum_{j=1}^{n} \alpha_j \cdot \sin(\sum_{i=1}^{2} \beta_{i,j} \cdot (x_i - \gamma_{i,j}))$	1	$2.46{\cdot}10^{-1}$	$6.60{\cdot}10^{-3}$	$2.17{\cdot}10^{-4}$	$5.87{\cdot}10^{-5}$	$2.20{\cdot}10^{-5}$
	5	$2.52{\cdot}10^{-1}$	$6.80{\cdot}10^{-3}$	$2.16{\cdot}10^{-4}$	$5.62{\cdot}10^{-5}$	$4.72{\cdot}10^{-5}$
	10	$3.24{\cdot}10^{-1}$	$1.24{\cdot}10^{-2}$	$2.24{\cdot}10^{-4}$	$6.06{\cdot}10^{-5}$	$5.77{\cdot}10^{-5}$
$\sum_{j=1}^{n+1} \sum_{k=1}^{n+1} \sum_{i=1}^{2} \beta_{i,k,j} x_i x_i \in \mathcal{D}_{i,k}$	1	$1.68{\cdot}10^{-1}$	$1.00{\cdot}10^{-1}$	$2.88{\cdot}10^{-2}$	$3.67{\cdot}10^{-2}$	$3.85{\cdot}10^{-2}$
	5	$2.91{\cdot}10^{-1}$	$1.33{\cdot}10^{-1}$	$5.41{\cdot}10^{-2}$	$4.51{\cdot}10^{-2}$	$4.34{\cdot}10^{-2}$
	10	$2.51{\cdot}10^{-1}$	$1.26{\cdot}10^{-1}$	$6.20{\cdot}10^{-2}$	$3.99{\cdot}10^{-2}$	$4.26{\cdot}10^{-2}$

Table 1: Approximation error δ with different classes of non–polynomial functions.

Dresher, Melvin, S. K., and Shapley, L. S. 1950. Polynomial Games. In *Contributions to the Theory of Games*. 161–180.

Fudenberg, D., and Tirole, J. 1991. *Game Theory*. The Mit Press.

Gatti, N., and Iuliano, C. 2011. Computing an extensive–form perfect equilibrium in two–player games. In *AAAI*, 669–674.

Gatti, N., and Panozzo, F. 2012. New results on the verification of nash refinements for extensive–form games. In *AAMAS*, in publication.

Gatti, N., and Restelli, M. 2011. Equilibrium Approximation in Extensive–Form Simulation–Based Games. In *AAMAS*, 199–206.

Glicksberg, I. L. 1952. A further generalization of the kakutani fixed point theorem, with application to nash equilibrium points. *AMER MATH SOC* 3(1):170–174.

Karlin, S., and Shapley, L. S. 1953. Geometry of Moment Spaces. *MEM AM MATH SOC* 12:105.

Karlin, S. 1951. Continuous games. *NAT ACAD SCIE USA* 37(4):220–223.

Karlin, S. 1959. Mathematical Methods and Theory in Games, Programming, and Economics. *Vol. I: Matrix games, programming, and mathematical economics. Vol. II: The Theory of infinite games* I + II.

K.C. Toh, M. T., and Tutuncu, R. 1999. Sdpt3 - a matlab software package for semidefinite programming. *Optimization Methods and Software* (11):545–581.

Lofberg, J. 2004. YALMIP : a toolbox for modeling and optimization in MATLAB. *CACSD* (4):284 – 289.

Parrilo, P. A. 2006. Polynomial Games and Sum of Squares Optimization. *CDC* 2855 – 2860.

Remez, Y. L. 1935. On a method of tchebycheff type approximation of functions. *UKRAIN ANN*.

Rosen, J. B. 1987. Existence and Uniqueness of Equilibrium Point for Concave n-Person Games. *ECONOMETRICA* 33:520–533.

Schmeiser, B., and Devroye, L. 1988. Non-Uniform Random Variate Generation. *J AM STAT ASSOC* 83(403):906.

Shah, P., and Parrilo, P. A. 2007. Polynomial Stochastic Games via Sum of Squares Optimization. *CDC* 745–750.

Shapley, L. S. 1953. Stochastic Games. *NAT ACAD SCIE USA* 39(10):1095–1100.

Shoham, Y., and Leyton-Brown, K. 2010. *Multiagent Systems: Algorithmic, Game-Theoretic, and Logical Foundations*.

Shohat, J., and Tamarkin, J. 1943. The Problem of Moments. *American Mathematical Society Mathematical surveys* 1.

Stein, N. D.; Ozdaglar, A.; and Parrilo, P. A. 2008. Separable and Low-Rank Continuous Games. *INT J GAME THEORY* 37(4):475–504.

Vorobeychik, Y., and Wellman, M. P. 2008. Stochastic search methods for Nash equilibrium approximation in simulation–based games. In *AAMAS*, 1055–1062.

Vrieze, O. J.; Tijs, S. H.; Raghavan, T. E. S.; and Filar, J. A. 1983. A Finite Algorithm for the Switching Control Stochastic Game. *OR SPEKTRUM* 5(1):15–24.