An Approximate Subgame-Perfect Equilibrium Computation Technique for Repeated Games

Andriy Burkov and **Brahim Chaib-draa**
DAMAS Laboratory, Laval University,
Quebec, Canada G1K 7P4
{burkov,chaib}damas.ift.ulaval.ca

Abstract

This paper presents a technique for approximating, up to any precision, the set of subgame-perfect equilibria (SPE) in repeated games with discounting. The process starts with a single hypercube approximation of the set of SPE payoff profiles. Then the initial hypercube is gradually partitioned on to a set of smaller adjacent hypercubes, while those hypercubes that cannot contain any SPE point are gradually withdrawn. Whether a given hypercube can contain an equilibrium point is verified by an appropriate mixed integer program. A special attention is paid to the question of extracting players' strategies and their representability in form of finite automata.

Introduction

In multiagent systems, each agent's strategy is optimal if it maximizes that agent's utility function, subject to the constraints induced by the respective strategies of the other agents. Game theory provides a compact yet sufficiently rich form of representing such strategic interactions. Repeated games (Osborne and Rubinstein 1999; Mailath and Samuelson 2006) are a formalism permitting modeling long-term strategic interactions between multiple selfish optimizers.

Probably the most known example of a repeated game is Prisoner's Dilemma (Figure 1). In this game, there are two

		Player 2 C	Player 2 D
Player 1	C	$2, 2$	$-1, 3$
	D	$3, -1$	$0, 0$

Figure 1: The payoff matrix of Prisoner's Dilemma.

players, and each of them can make two actions: C or D. When those players simultaneously perform their actions, the pair of actions induces a numerical payoff obtained by each player. The game then passes to the next stage, where it can be played again by the same pair of players.

Game theory assumes that the goal of each player is to play optimally, i.e., to maximize its utility function given the strategies of the other players. When the *a priori* information about all players' strategies and their real strategic preferences coincide, we talk about equilibria.

A pair of "Tit-For-Tat" (TFT) strategies is a well-known example of equilibrium in Repeated Prisoner's dilemma. TFT consists of starting by playing C. Then, each player should play the same action as the very recent action played by its opponent. Indeed, such history dependent equilibrium brings to each player a higher average payoff, than that of another, stationary, equilibrium of the repeated game (a pair of strategies that prescribe to play the stage-game Nash equilibrium (D, D) at every stage). An algorithmic construction of such strategies, given an arbitrary repeated game, is challenging. For the case where the utility function is given by the average payoff, (Littman and Stone 2005) propose a simple and efficient algorithm that constructs equilibrium strategies in two-player repeated games. On the other hand, when the players discount their future payoffs with a discount factor, a pair of TFT strategies is still an equilibrium only for certain values of the discount factor. (Judd, Yeltekin, and Conklin 2003) propose an approach for computing equilibria for different discount factors, but their approach is limited to pure strategies, and, as we will discuss below, has several other important limitations.

In this paper, we present an algorithmic approach to the problem of computing equilibria in repeated games when the future payoffs are discounted. Our approach is more general than that of (Littman and Stone 2005), because it allows an arbitrary discounting, and is free of four major limitations of the algorithm of (Judd, Yeltekin, and Conklin 2003). Furthermore, our algorithm finds only those strategies that can be adopted by artificial agents. The latter are usually characterized by a finite time to compute their strategies and a finite memory to implement them. To our knowledge, this is the first time that all these goals are achieved simultaneously.

The remainder of the paper is structured as follows. First, we formally state the problem. Then, we survey the previous work, by pointing out its limitations. Next, we present our novel ASPECT algorithm for approximately solving discounted repeated games and for extracting strategies. We then state our main theoretical result and give an overview of a number of experimental results. We conclude with a short discussion.

Problem Statement

Stage-Game

A *stage-game* is a tuple $(N, \{A_i\}_{i\in N}, \{r_i\}_{i\in N})$. In a stage-game, there is a finite set N of individual players ($|N| \equiv n$). Player $i \in N$ has a finite set A_i of *(pure) actions*. Each player i chooses a certain action $a_i \in A_i$; the resulting vector $a \equiv (a_i)_{i\in N}$ forms an *action profile* that belongs to the set of action profiles $A \equiv \times_{i\in N} A_i$. The action profile is then executed and the corresponding stage-game *outcome* is realized. A player specific *payoff function* r_i specifies player i's payoffs for different game outcomes. A bijection is typically assumed between the set of action profiles and the set of game outcomes. In this case, a player's payoff function is defined as the mapping $r_i : A \mapsto \mathbb{R}$.

Given $a \in A$, $r(a) \equiv (r_i(a))_{i\in N}$ is called a *payoff profile*. A *mixed action* α_i of player i is a probability distribution over its actions, i.e., $\alpha_i \in \Delta(A_i)$. A *mixed action profile* is a vector $\alpha \equiv (\alpha_i)_{i\in N}$. We denote by $\alpha_i^{a_i}$ and α^a respectively the probability to play action a_i by player i and the probability that the outcome a will be realized by α, i.e., $\alpha^a \equiv \prod_i \alpha_i^{a_i}$. Payoff functions can be extended to mixed action profiles by taking expectations.

Let $-i$ stand for "all players except i". A *(Nash) equilibrium in a stage-game* is a mixed action profile α, s.t. for each player i and $\forall \alpha_i' \in \Delta(A_i)$, the following holds:

$$r_i(\alpha) \geq r_i(\alpha_i', \alpha_{-i}), \text{ where } \alpha \equiv (\alpha_i, \alpha_{-i}).$$

Repeated Game

In a repeated game, the same stage-game is played in *periods* (or *stages*) $t = 0, 1, 2, \ldots$. When the number of game periods is not known in advance and can be infinite, the repeated game is called *infinite*. This is the scope of the present paper.

The set of the repeated game *histories up to period* t is given by $H^t \equiv \times_t A$. The set of *all possible histories* is given by $H \equiv \bigcup_{t=0}^{\infty} H^t$. For instance, a history $h^t \in H^t$ is a stream of outcomes realized in the repeated game, starting from period 0 up to period $t-1$: $h^t \equiv (a^0, a^1, a^2, \ldots, a^{t-1})$. A *(mixed) strategy of player* i is a mapping $\sigma_i : H \mapsto \Delta(A_i)$. A *pure strategy* is a strategy that puts weight 1 on only one pure action at any $h \in H$. A *strategy profile* is a vector $\sigma \equiv (\sigma_i)_{i\in N}$. We denote by Σ_i the set of strategies of player i, and by $\Sigma \equiv \times_{i\in N}\Sigma_i$ the set of strategy profiles.

A *subgame* is a repeated game that continues after a certain history. Imagine a subgame induced by a history h^t. Given a strategy profile σ, the behavior of players in this subgame after a history h^τ is identical to the behavior in the original repeated game after the history $h^t \cdot h^\tau$, a concatenation of two histories. For a pair (σ, h), the *subgame strategy profile induced by* h is denoted as $\sigma|_h$.

An *outcome path* in the repeated game is a possibly infinite stream $\mathbf{a} \equiv (a^0, a^1, \ldots)$ of action profiles. A finite prefix of length t of an outcome path corresponds to a history in H^{t+1}. At each repeated game run, a strategy profile σ induces a certain outcome path $\mathbf{a}$.

Let σ be a strategy profile. The *discounted average payoff of σ for player i* is defined as

$$u_i^\gamma(\sigma) \equiv (1-\gamma)\, \mathrm{E}_{\mathbf{a}\sim\sigma} \sum_{t=0}^{\infty} \gamma^t r_i(a^t),$$

where $\gamma \in [0, 1)$ is the *discount factor*, which can be interpreted as the probability that the repeated game will continue after each period. We define the *payoff profile induced by* σ as $u^\gamma(\sigma) \equiv (u_i^\gamma(\sigma))_{i\in N}$.

The strategy profile σ is a *(Nash) equilibrium* if, for each player i and its strategies $\sigma_i' \in \Sigma_i$,

$$u_i^\gamma(\sigma) \geq u_i^\gamma(\sigma_i', \sigma_{-i}), \text{ where } \sigma \equiv (\sigma_i, \sigma_{-i}).$$

A strategy profile σ is a *subgame-perfect equilibrium* (SPE) in the repeated game, if for all histories $h \in H$, the subgame strategy profile $\sigma|_h$ is an equilibrium in the subgame.

Strategy Profile Automata

The strategies for artificial agents usually should have a finite representation. Let $M \equiv (Q, q^0, f, \tau)$ be an *automaton implementation of a strategy profile* σ. It consists of a set of states Q, with the initial state $q^0 \in Q$; of a profile of decision functions $f \equiv (f_i)_{i\in N}$, where $f_i : Q \mapsto \Delta(A_i)$ is the decision function of player i; and of a transition function $\tau : Q \times A \mapsto Q$, which identifies the next state of the automaton given the current state and the action profile.

Let $|M|$ denote the number of states of automaton M. If $|M|$ is finite, such automaton is called *finite*. (Kalai and Stanford 1988) showed that any SPE can be approximated with a finite automaton. They defined the notion of an approximate SPE as follows. For an *approximation factor* $\epsilon > 0$, a strategy profile $\sigma \in \Sigma$ is an *ϵ-equilibrium*, if for each player i and $\forall \sigma_i' \in \Sigma_i$, $u_i^\gamma(\sigma) \geq u_i^\gamma(\sigma_i', \sigma_{-i}) - \epsilon$, where $\sigma \equiv (\sigma_i, \sigma_{-i})$. A strategy profile $\sigma \in \Sigma$ is a *subgame-perfect ϵ-equilibrium* (SPϵE) in a repeated game, if $\forall h \in H$, $\sigma|_h$ is an ϵ-equilibrium in the subgame induced by h.

Theorem 1 ((Kalai and Stanford 1988)). *Consider a repeated game with the parameters γ and ϵ. For any SPE σ, there exists a finite automaton M, s.t. $|u_i^\gamma(\sigma) - u_i^\gamma(M)| < \epsilon$, for all $i \in N$, and M induces an SPϵE.*

Problem Statement

Let $U^\gamma \subset \mathbb{R}^n$ be the set of SPE payoff profiles in a repeated game with the discount factor γ. Let $\Sigma^{\gamma,\epsilon} \subseteq \Sigma$ be the set of SPϵE strategy profiles. The problem of an approximate subgame-perfect equilibrium computation is stated as follows: find a set $W \supseteq U^\gamma$ with the property that for any $v \in W$, one can find a finite automaton M inducing a strategy profile $\sigma^M \in \Sigma^{\gamma,\epsilon}$, s.t. $v_i - u_i^\gamma(M) \leq \epsilon, \forall i \in N$.

Note that the goal of any player is to maximize its payoff, while the strategy is a means. So, we set out with a goal to find a set W that does not omit any SPE payoff profile. Ideally, the set W has to be as small as possible. This latter property is enforced by our second goal, which is to be capable of constructing, for any payoff profile $v \in W$, a finite automaton that can approximately induce that payoff profile.

Previous Work

The work on equilibrium computation can be divided into three main groups. The algorithms of the first group solve the problem of computing one or several stage-game equilibria using only the payoff matrix. The discount factor is implicitly assumed to be zero (Lemke and Howson 1964; Porter, Nudelman, and Shoham 2008).

At the other extremity, there are algorithms that assume γ to be arbitrarily close to 1. For instance, in two-player repeated games, this permits efficiently construct automata inducing SPE strategy profiles (Littman and Stone 2005).

The algorithms of the third group (Cronshaw 1997; Judd, Yeltekin, and Conklin 2003) aim at computing equilibria by assuming that γ is a fixed value in the open interval $(0, 1)$. These algorithms are based on the concept of *self-generating sets*. Let us briefly present it here. Let V^γ denote the set of pure SPE payoff profiles one wants to identify. Let $BR_i(\alpha)$ be a stage-game best response of player i to the mixed action profile $\alpha \equiv (\alpha_i, \alpha_{-i})$:

$$BR_i(\alpha) \equiv \max_{a_i \in A_i} r_i(a_i, \alpha_{-i}).$$

Define the map B^γ on a set $W \subset \mathbb{R}^n$:

$$B^\gamma(W) \equiv \bigcup_{(a,w)\in A\times W} (1-\gamma)r(a) + \gamma w,$$

where w is the *continuation promise* that verifies, for all i:

$$(1-\gamma)r_i(a) + \gamma w_i - (1-\gamma)r_i(BR_i(a), a_{-i}) - \gamma \underline{w}_i \geq 0,$$

and $\underline{w}_i \equiv \inf_{w\in W} w_i$. (Abreu, Pearce, and Stacchetti 1990) show that the largest fixed point of $B^\gamma(W)$ is V^γ.

Any numerical implementation of $B^\gamma(W)$ requires an efficient representation of the set W in a machine. (Judd, Yeltekin, and Conklin 2003) use convex sets for approximating both W and $B^\gamma(W)$ as an intersection of a finite number of hyperplanes. The main limitations of this approach are as follows: (*i*) it assumes the existence of at least one pure action stage-game equilibrium; (*ii*) it permits computing only pure action SPE payoff and strategy profiles; (*iii*) it cannot find SPE strategy profiles implementable by finite automata; (*iv*) it relies on availability of public correlation (Mailath and Samuelson 2006).

In the next section, we present our novel ASPECT algorithm (for *Approximate Subgame-Perfect Equilibrium Computation Technique*) for approximating the set of SPE payoff profiles. The set of payoff profiles W returned by our initial complete formulation of ASPECT includes, among others, all pure strategy SPE as well as all stationary mixed strategy SPE. However, it can omit certain equilibrium points and, therefore, W does not entirely contain U^γ. In a subsequent section, we will propose an extension of ASPECT capable of completely approximating U^γ, by assuming public correlation.

Our ASPECT Algorithm

The fixed point property of the map B^γ and its relation to the set of SPE payoff profiles can be used to approximate the latter. The idea is to start by a set W that entirely contains U^γ, and then to iteratively eliminate all points $w' \in W$, for which $\nexists (w, \alpha) \in W \times \Delta(A_1) \times \ldots \times \Delta(A_n)$, such that

$$\begin{aligned} w' = (1-\gamma)r(\alpha) + \gamma w \text{ and } (1-\gamma)r_i(\alpha) + \gamma w_i \\ -(1-\gamma)r_i(BR_i(\alpha), \alpha_{-i}) - \gamma \underline{w}_i \geq 0,\ \forall i. \end{aligned} \quad (1)$$

Algorithm 1 outlines the basic structure of ASPECT. It starts with an initial approximation W of the set of SPE payoff profiles U^γ. The set W, in turn, is represented by a union of disjoint hypercubes belonging to the set C. Each hypercube $c \in C$ is identified by its origin $o^c \in \mathbb{R}^n$ and by the hypercube side length l. Initially, C contains only one hypercube c, whose origin o^c is set to be a vector $(\underline{r})_{i\in N}$; the side length l is set to be $l = \bar{r} - \underline{r}$, where $\underline{r} \equiv \min_{a,i} r_i(a)$ and $\bar{r} \equiv \max_{a,i} r_i(a)$. Therefore, initially W entirely contains U^γ.

Input: r, a payoff matrix; γ, ϵ, the parameters.
1: Let $l \equiv \bar{r} - \underline{r}$ and $o^c \equiv (\underline{r})_{i\in N}$;
2: Set $C \leftarrow \{(o^c, l)\}$;
3: **loop**
4: Set ALLCUBESCOMPLETED $\leftarrow$ TRUE;
5: Set NOCUBEWITHDRAWN $\leftarrow$ TRUE;
6: **for each** $c \equiv (o^c, l) \in C$ **do**
7: Let $\underline{w}_i \equiv \min_{c\in C} o^c_i$; set $\underline{w} \leftarrow (\underline{w}_i)_{i\in N}$;
8: **if** CUBESUPPORTED(c, C, $\underline{w}$) is FALSE **then**
9: Set $C \leftarrow C \backslash \{c\}$;
10: **if** $C = \varnothing$ **then**
11: **return** FALSE;
12: Set NOCUBEWITHDRAWN $\leftarrow$ FALSE;
13: **else**
14: **if** CUBECOMPLETED(c) is FALSE **then**
15: Set ALLCUBESCOMPLETED $\leftarrow$ FALSE;
16: **if** NOCUBEWITHDRAWN is TRUE **then**
17: **if** ALLCUBESCOMPLETED is FALSE **then**
18: Set $C \leftarrow$ SPLITCUBES(C);
19: **else**
20: **return** C.

Algorithm 1: The basic structure of ASPECT.

Each iteration of ASPECT (the loop, line 3) consists of verifying, for each hypercube c, whether it has to be eliminated from the set C (procedure CUBESUPPORTED). If c does not contain any point w' satisfying the conditions of Equation (1), this hypercube is withdrawn from the set C. If, by the end of a certain iteration, no hypercube was withdrawn, each remaining hypercube is split into 2^n disjoint hypercubes with side $l/2$ (procedure SPLITCUBES). The process continues until, for each remaining hypercube, a stopping criterion is satisfied (procedure CUBECOMPLETED).

The CUBESUPPORTED Procedure

Computing the set of all equilibria is a challenging task. To the best of our knowledge, there is no algorithm capable of even approximately solving this problem. When the action profiles are allowed to be mixed, their one by one enumeration[1] is impossible. Furthermore, a deviation of one player from a mixed action can only be detected by the others if the deviation is done in favor of an out-of-support action[2].

We solve the two aforementioned problems as follows. We first define a special mixed integer program (MIP). We then let the solver decide on which actions to be included

[1] As, for example, in (Judd, Yeltekin, and Conklin 2003).

[2] The *support of a mixed action* α_i is a set $A_i^{\alpha_i} \subseteq A_i$, which contains all pure actions to which α_i assigns a non-zero probability.

into the mixed action support of each player, and what probability has to be assigned to those actions. Note that player i is only willing to randomize according to a suggested mixture α_i, if it is indifferent over the pure actions in the support of that mixture. The trick is to specify different continuation promises for different actions in the support, such that the expected payoff of each action remains bounded by the dimensions of the hypercube. Algorithm 2 defines the procedure CUBESUPPORTED for ASPECT.

Input: $c \equiv (o^c, l)$, a hypercube; C, a set of hypercubes; $\underline{w}$ a vector of payoffs.

1: **for each** $\tilde{c} \equiv (o^{\tilde{c}}, l) \in C$ **do**

2: Solve the following mixed integer program:

Decision variables: $w_i(a_i) \in \mathbb{R}$, $w'_i(a_i) \in \mathbb{R}$, $y_i^{a_i} \in \{0,1\}$, $\alpha_i^{a_i} \in [0,1]$ for all $i \in \{1,2\}$ and for all $a_i \in A_i$.

Objective function: $\min f \equiv \sum_i \sum_{a_i} y_i^{a_i}$.

Subject to constraints:

$$(1) \quad \forall i: \ \sum_{a_i} \alpha_i^{a_i} = 1;$$

For all i and for all $a_i \in A_i$:

$$(2) \quad \alpha_i^{a_i} \leq y_i^{a_i},$$

$$(3) \quad w'_i(a_i) = (1-\gamma) \sum_{a_{-i}} \alpha_{-i}(a_{-i}) r_i(a_i, a_{-i}) + \gamma w_i(a_i),$$

$$(4) \quad o_i^c y_i^{a_i} \leq w'_i(a_i) \leq l y_i^{a_i} + o_i^c,$$

$$(5) \quad \underline{w}_i - \underline{w}_i y_i^{a_i} + o_i^{\tilde{c}} y_i^{a_i} \leq w_i(a_i) \leq (\underline{w}_i + l) - (\underline{w}_i + l) y_i^{a_i} + (o_i^{\tilde{c}} + l) y_i^{a_i}.$$

3: **if** a solution is found **then return** $w_i(a_i)$ and $\alpha_i^{a_i}$ for all $i \in \{1,2\}$ and for all $a_i \in A_i$.

4: **return** FALSE

Algorithm 2: CUBESUPPORTED for mixed strategies.

The procedure CUBESUPPORTED of Algorithm 2 verifies whether a given hypercube c has to be kept in the set of hypercubes. If yes, CUBESUPPORTED returns a mixed action profile α and the corresponding continuation promise payoffs for each action in the support of α_i. Otherwise, the procedure returns FALSE. The indifference of player i between the actions in the support of α_i is (approximately) secured by the constraint (4) of the MIP. In an optimal solution of the MIP, any binary indicator variable $y_i^{a_i}$ can only be equal to 1 if a_i is in the support of α_i. Therefore, each $w'_i(a_i)$ is either bounded by the dimensions of the hypercube, if $a_i \in A_i^{\alpha_i}$, or, otherwise, is below the origin of the hypercube.

Note that the above MIP is only linear in the case of two players. For more than two players, the problem becomes non-linear, because α_{-i} is now given by a product of decision variables α_j, for all $j \in N \backslash \{i\}$. Such optimization problems are known to be very difficult to solve (Saxena, Bonami, and Lee 2008). We solved all linear MIP problems defined in this paper using CPLEX (IBM, Corp. 2009) together with OptimJ (ATEJI 2009).

Computing Strategies

Our ASPECT algorithm, defined in Algorithm 1, returns the set of hypercubes C, such that the union of these hypercubes gives W, a set that contains a certain subset of U^γ. Intuitively, each hypercube represents all those strategy profiles that induce similar payoff profiles. Therefore, one can view hypercubes as states of an automaton. Algorithm 3 constructs an automaton M that implements a strategy profile that approximately induces any payoff profile $\tilde{v} \in W$.

Input: C, a set of hypercubes, such that W is their union; $\tilde{v} \in W$, a payoff profile.

1: Find a hypercube $c \in C$, which $\tilde{v}$ belongs to; set $Q \leftarrow \{c\}$ and $q^0 \leftarrow c$;

2: **for each** player i **do**

3: Find $\underline{w}^i = \min_{w \in W} w_i$ and a hypercube $c^i \in C$, which $\underline{w}^i$ belongs to;

4: Set $Q \leftarrow Q \cup \{c^i\}$;

5: Set $f \leftarrow \varnothing \mapsto \times_i \Delta(A_i)$;

6: Set $\tau \leftarrow \varnothing \mapsto C$.

7: **loop**

8: Pick a hypercube $q \in Q$, for which $f(q)$ is not defined, or **return** $M \equiv (Q, q^0, f, \tau)$ if there is no such hypercubes.

9: Apply the procedure CUBESUPPORTED(q) and obtain a (mixed) action profile α and continuation payoff profiles $w(a)$ for all $a \in \times_i A_i^{\alpha_i}$.

10: Define $f(q) \equiv \alpha$.

11: **for each** $a \in \times_i A_i^{\alpha_i}$ **do**

12: Find a hypercube $c \in C$, which $w(a)$ belongs to, set $Q \leftarrow Q \cup \{c\}$;

13: Define $\tau(q, a) \equiv c$.

14: **for each** i and **each** $a^i \in (A \backslash A_i^{\alpha_i}) \times_{j \in N \backslash \{i\}} A_i^{\alpha_i}$ **do**

15: Define $\tau(q, a^i) \equiv c^i$.

Algorithm 3: Algorithm for constructing an automaton M that approximately induces the given payoff profile v.

Algorithm 3 starts with an empty set of states Q. Then it puts N punishment states[3] into this set, one for each player (lines 3–4). The transition and the decision function for any state that has just been put into Q remain undefined. From the set Q of automaton states, Algorithm 3 then iteratively picks some state q, for which the transition and the decision function have not yet been defined (line 8). Then the procedure CUBESUPPORTED is applied to state q, and a mixed action profile α and continuation payoff profiles $w(a)$ for all $a \in \times_{i \in N} A_i^{\alpha_i}$ are obtained. This mixed action profile α will be played by the players when automation enters into state q during game play (line 10). For each $w(a)$, a hypercube $c \in C$ is found, which $w(a)$ belongs to. Those hypercubes are also put into the set of states Q (line 12) and the transition function for state q is finally defined (lines 13 and 15). Algorithm 3 terminates when, for all $q \in Q$, the transition function and the decision function have been defined.

The Stopping Criterion

The values of the flags NOCUBEWITHDRAWN and ALLCUBESCOMPLETED of Algorithm 1 determine whether AS-

[3] The punishment state for player i is the automaton state, which is based on the hypercube that contains a payoff profile v, such that $v_i = \underline{w}_i$.

PECT should stop and return the solution. At the end of each iteration, the flag ALLCUBESCOMPLETED is only TRUE if for each remaining hypercube $c \in C$, CUBECOMPLETED(c) is TRUE. The procedure CUBECOMPLETED(c) verifies that (*i*) the automaton that starts in the state given by c induces an SPϵE, and (*ii*) the payoff profile induced by this automaton is ϵ-close to o^c. Both conditions can be verified by dynamic programming: assume the remaining agents' strategies fixed and use value iteration to compute, for each player $i \in N$, both the values of following the strategy profile and the values of deviations.

The Main Theorem

Here, we present the main theoretical result of the paper.

Theorem 2. *For any repeated game, discount factor γ and approximation factor ϵ, (*i*) ASPECT (Algorithms 1, 2) terminates in finite time, (*ii*) the set of hypercubes C, at any moment, contains at least one hypercube, (*iii*) for any input $\tilde{v} \in W$, Algorithm 3 terminates in finite time and returns a finite automaton M that satisfies: (1) the strategy profile σ^M implemented by M induces the payoff profile v, s.t. $\tilde{v}_i - v_i \leq \epsilon, \forall i \in N$, and (2) the maximum payoff g_i that each player i can achieve by unilaterally deviating from σ^M is such that $g_i - v_i \leq \epsilon$.*

The proof of Theorem 2 relies on the six lemmas given below. Due to limited space, we give only a brief intuition behind the proofs of certain of them.

Lemma 1. *At any point of execution of ASPECT, C contains at least one hypercube.*

Proof. According to (Nash 1950), any stage-game has at least one equilibrium. Let v be a payoff profile of a certain Nash equilibrium in the stage-game. For the hypercube c that contains v, the procedure CUBESUPPORTED will always return TRUE, because, for any γ, v satisfies the two conditions of Equation (1), with $w' = w = v$ and α being a mixed action profile that induces v. Therefore, c will never be withdrawn. □

Lemma 2. ASPECT *will reach an iteration, such that* NOCUBEWITHDRAWN *is* TRUE*, in finite time.*

Proof. Because the number of hypercubes is finite, the procedure CUBESUPPORTED will terminate in finite time. For a constant l, the set C is finite and contains at most $\lceil (\bar{r} - \underline{r})/l \rceil$ elements. Therefore, after a finite time, there will be an iteration of ASPECT, such that for all $c \in C$, CUBESUPPORTED(c) returns TRUE. □

Lemma 3. *Let C be a set of hypercubes at the end of a certain iteration of ASPECT, such that* NOCUBEWITHDRAWN *is* TRUE*. For all $c \in C$, Algorithm 3 terminates in finite time and returns a complete finite automaton.*

Proof. By observing the definition of Algorithm 3, the proof follows from the fact that the number of hypercubes and, therefore, the possible number of the automaton states is finite. Furthermore, the automaton is complete, because the fact that NOCUBEWITHDRAWN is TRUE implies that for each hypercube $c \in C$, there is a mixed action α and a continuation payoff profile w belonging to a certain hypercube $c' \in C$. Consequently, for each state q of the automaton, the functions $f(q)$ and $\tau(q)$ will be defined. □

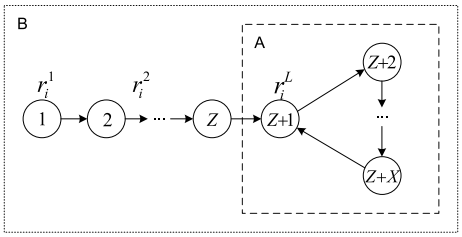


Figure 2: A generic equilibrium graph for player i.

Lemma 4. *Let C be the set of hypercubes at the end of a certain iteration of ASPECT, such that* NOCUBEWITHDRAWN *is* TRUE*. Let l be the current value of the hypercube side length. For every $c \in C$, the strategy profile σ^M, implemented by the automaton M that starts in c, induces the payoff profile $v \equiv u^\gamma(\sigma^M)$, such that $o_i^c - v_i \leq \frac{\gamma l}{1-\gamma}$, $\forall i \in N$.*

Proof. Here, we give only the intuition. The proof is built on the fact that when player i is following the strategy prescribed by the automaton constructed by Algorithm 3, this process can be reflected by an *equilibrium graph*, as the one shown in Figure 2. The graph represents the initial state followed by a non-cyclic sequence of states (nodes 1 to Z) followed by a cycle of X states (nodes $Z+1$ to $Z+X$). The labels over the nodes are the immediate expected payoffs collected by player i in the corresponding states. A generic equilibrium graph contains one non-cyclic and one cyclic part. For two successive nodes q and $q+1$ of the equilibrium graph we have:

$$o_i^q \leq (1-\gamma) r_i^q + \gamma w_i^q \leq o_i^q + l,$$
$$o_i^{q+1} \leq w_i^q \leq o_i^{q+1} + l,$$

where o_i^q, r_i^q and w_i^q stand respectively for (*i*) the payoff of player i in the origin of the hypercube behind the state q, (*ii*) the immediate expected payoff of player i for playing according to $f_i(q)$ or for deviating inside the support of $f_i(q)$, and (*iii*) the continuation promise payoff of player i for playing according to the equilibrium strategy profile in state q.

By means of the developments based on the properties of the sum of geometric series, one can derive the equation for v_i, the long-term expected payoff of player i for passing through the equilibrium graph infinitely often. Further developments allow us to conclude that $o_i^c - v_i \leq \frac{\gamma l}{1-\gamma}$. □

Lemma 5. *Let C be the set of hypercubes at the end of a certain iteration of ASPECT, such that* NOCUBEWITHDRAWN *is* TRUE*. Let l be the current value of the hypercube side length. For every $c \in C$, the maximum payoff g_i that each player i can achieve by unilaterally deviating from the strategy profile σ^M implemented by an automaton M that starts in c and induces the payoff profile $v \equiv u^\gamma(\sigma^M)$ is such that $g_i - v_i \leq \frac{2l}{1-\gamma}$, $\forall i \in N$.*

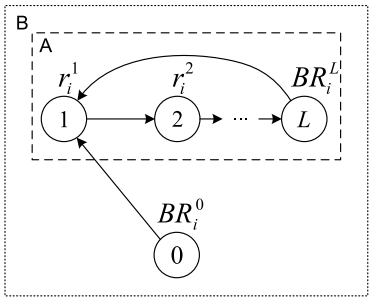


Figure 3: A generic deviation graph for player i.

Proof. The proof of this lemma is similar to that of the previous lemma. The difference is that, now, one has to consider *deviation graphs* for player i as the one depicted in Figure 3. A *deviation graph for player* i is a finite graph, which reflects the optimal behavior for player i assuming that the behaviors of the other players are fixed and are given by a finite automaton. The nodes of the deviation graph correspond to the states of the automaton. The labels over the nodes are the immediate expected payoffs collected by player i in the corresponding states. A *generic deviation graph for player* i (Figure 3) is a deviation graph that has one cyclic and one non-cyclic part. In the cyclic part (subgraph A), player i follows the equilibrium strategy or deviations take place inside the support of the prescribed mixed actions (nodes 1 to $L-1$, with node 1 corresponding to the punishment state for player i). In the last node of the cyclic part (node L), an out-of-the-support deviation takes place. The non-cyclic part of the generic deviation graph contains a single node corresponding to the state, where the initial out-of-the-support deviation of player i from the SPE strategy profile occurs.

Similar developments allow us to find, now for a deviation graph, an expression for g_i, the expected long-term payoff of the optimal deviation for player i. Then, one can similarly derive that $g_i - v_i \leq \frac{2l}{1-\gamma}$. □

Lemma 6. ASPECT *terminates in finite time.*

Proof. The hypercube side length l is reduced by half every time that no hypercube was withdrawn by the end of an iteration of ASPECT. Therefore, and by Lemma 2, any given value of l will be reached after a finite time. By Lemmas 4 and 5, ASPECT, in the worst case, terminates when l becomes lower than or equal to $\frac{\epsilon(1-\gamma)}{2}$. □

On combining the above lemmas we obtain Theorem 2.

Extensions

The assumption that all continuation payoff profiles for hypercube c are contained within a certain hypercube $c' \in C$ (Algorithm 2) makes the optimization problem linear and, therefore, easier to solve. However, this restricts the set of equilibria that can be approximated using this technique. Furthermore, examining prospective continuation hypercubes one by one is computational time-consuming: the worst-case complexity of one iteration of ASPECT is $O(|C|^2)$, assuming that solving one MIP takes a unit time.

Clustered Continuations

We can improve in both complexity and range of solutions by considering *clustered continuations*. A cluster is a convex hyperrectangle consisting of one or more adjacent hypercubes. The set S of hyperrectangular clusters is obtained from the set of hypercubes, C, by finding a smaller set, such that the union of its elements is equal to the union of the elements of C. Each $s \in S$ is identified by its origin $o^s \in \mathbb{R}^n$ and by the vector of side lengths $l^s \in \mathbb{R}^n$. When the continuations are given by convex clusters, the set of solutions can potentially be richer because, now, continuation payoff profiles for one hypercube can be found in different hypercubes (within one convex cluster). This also permits reducing the the worst-case complexity of iterations of ASPECT to $O(|C||S|) \leq O(|C|^2)$.

The CUBESUPPORTED procedure with clustered continuations will be different from that given by Algorithm 2 in a few details. Before line 1, the set of clusters S has first to be computed. To do that, a simple greedy approach can, for example, be used. Then, the line 1 has to be replaced by

"**for each** $s \equiv (o^s, l^s) \in S$ **do**".

Finally, the constraint (5) of line 2 has to be replaced by the following one:

$$\underline{w}_i - \underline{w}_i y_i^{a_i} + o_i^s y_i^{a_i} \leq w_i(a_i) \leq (\underline{w}_i + l) - (\underline{w}_i + l) y_i^{a_i} + (o_i^s + l_i^s) y_i^{a_i}.$$

A more general formulation of the MIP could allow the continuations for different action profiles to belong to different clusters. This would, however, again result in a hard non-linear MIP. The next extension permits preserving in W all SPE payoff profiles (i.e., $W \supseteq U^\gamma$) while maintaining the linearity of the MIP for the two-player repeated games.

Public Correlation

By assuming the set of continuation promises to be convex, one does not need to use multiple clusters to contain continuation promises in order to improve the range of solutions. Moreover, such an assumption guarantees that *all* realizable SPE payoff profiles will be preserved in W. A convexification of the set of continuation payoff profiles can be done in different ways, one of which is *public correlation*. In practice, this implies the existence of a certain random signal observable by all players after each repeated game iteration, or that a communication between players is available (Mailath and Samuelson 2006).

Algorithm 4 contains the definition of the CUBESUPPORTED procedure that convexifies the set of continuation promises. The definition is given for two players. The procedure first identifies $\mathrm{co}\, W$, the smallest convex set containing all hypercubes of the set C (procedure GETHALFPLANES). This convex set is represented as a set P of half-planes. Each element $p \in P$ is a vector $p \equiv (\phi^p, \psi^p, \lambda^p)$, s.t. the inequality $\phi^p x + \psi^p y \leq \lambda^p$ identifies a half-plane in a two-dimensional space. The intersection of these half-planes gives $\mathrm{co}\, W$. In our experiments, in order to construct the set P, we used the Graham scan (Graham 1972).

The procedure CUBESUPPORTED defined in Algorithm 4 differs from that of Algorithm 2 in the following aspects. It does not search for continuation payoffs in different hyper-

cubes by examining them one by one and, therefore, it does not iterate. Instead, it convexifies the set W and searches for continuation promises for the hypercube c inside $\operatorname{co} W$. The definition of the MIP is also different. New indicator variables, z^{a_1,a_2}, for all pairs $(a_1, a_2) \in A_1 \times A_2$, are introduced. The new constraint (6), jointly with the modified objective function, verify that z^{a_1,a_2} is only equal to 1 whenever both $y_1^{a_1}$ and $y_2^{a_2}$ are equal to 1. In other words, $z^{a_1,a_2} = 1$, only if $a_1 \in A_1^{\alpha_1}$ and $a_2 \in A_2^{\alpha_2}$. Constraint (7) verifies that $(w_1(a_1), w_2(a_2))$, the continuation promise payoff profile, belongs to $\operatorname{co} W$ if and only if $(a_1, a_2) \in A_1^{\alpha_1} \times A_2^{\alpha_2}$. Note that in the constraint (7), M stands for a sufficiently large number. This is a standard trick for relaxing any constraint.

Input: $c \equiv (o^c, l)$, a hypercube; C, a set of hypercubes.

1: $P \leftarrow$ GETHALFPLANES(C);

2: Solve the following linear MIP:

Decision variables: $w_i(a_i) \in \mathbb{R}$, $w_i'(a_i) \in \mathbb{R}$, $y_i^{a_i} \in \{0,1\}$, $\alpha_i^{a_i} \in [0,1]$ for all $i \in \{1,2\}$ and for all $a_i \in A_i$; $z^{a_1,a_2} \in \{0,1\}$ for all pairs $(a_1, a_2) \in A_1 \times A_2$;
Obj. function: $\min f \equiv \sum_{(a_1,a_2)\in A_1\times A_2} z^{a_1,a_2}$;
Subject to constraints:

(1) $\sum_{a_i} \alpha_i^{a_i} = 1,\ \forall i \in \{1,2\}$;
For all $i \in \{1,2\}$ and for all $a_i \in A_i$:
(2) $\alpha_i^{a_i} \le y_i^{a_i}$,
(3) $w_i'(a_i) = (1-\gamma)\sum_{a_{-i}} \alpha_{-i}(a_{-i}) r_i(a_i, a_{-i}) + \gamma w_i(a_i)$,
(4) $o_i^c y_i^{a_i} \le w_i'(a_i) \le l y_i^{a_i} + o_i^c$,
(5) $\underline{w}_i - \underline{w}_i y_i^{a_i} \le w_i(a_i) \le (\underline{w}_i + l) - (\underline{w}_i + l) y_i^{a_i} + \bar{r} y_i^{a_i}$;
$\forall a_1 \in A_1$ and $\forall a_2 \in A_2$:
(6) $y_1^{a_1} + y_2^{a_2} \le z^{a_1,a_2} + 1$;
$\forall p \equiv (\phi^p, \psi^p, \lambda^p) \in P$ and $\forall (a_1, a_2) \in A_1 \times A_2$:
(7) $\phi^p w_1(a_1) + \psi^p w_2(a_2) \le \lambda^p z^{a_1,a_2} + M - M z^{a_1,a_2}$.

3: **if** a solution is found **then return** $w_i(a_i)$ and $\alpha_i^{a_i}$ for all $i \in \{1,2\}$ and for all $a_i \in A_i$.

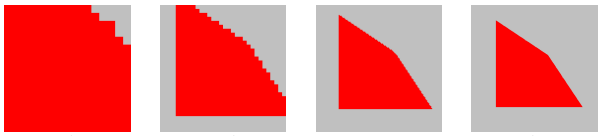
4: **return** FALSE

Algorithm 4: CUBESUPPORTED with public correlation.

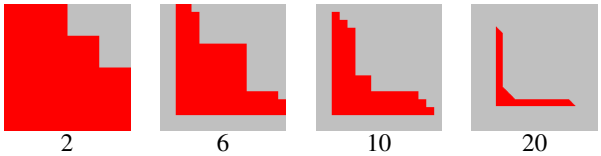
Experimental Results

In this section, we outline the results of our experiments with a number of well-known games, for which the payoff matrices are standard and their equilibrium properties have been extensively studied: Prisoner's Dilemma, Duopoly (Abreu 1988), Rock-Paper-Scissors, and Battle of the Sexes.

The graphs in Figure 4 reflect, for two different values of γ, the evolution of the set of SPE payoff profiles computed by ASPECT extended with public correlation in Prisoner's Dilemma. Here and below, the vertical and the horizontal axes of each graph correspond respectively to the payoffs of Players 1 and 2. Each axis is bounded respectively by $\bar{r}$ and $\underline{r}$. The numbers under the graphs are iterations of the algorithm. The red (darker) regions reflect the hypercubes



(a) $\gamma = 0.7$, $\epsilon = 0.01$



(b) $\gamma = 0.3$, $\epsilon = 0.01$

Figure 4: The evolution of the set of SPE payoff profiles in Prisoner's Dilemma with public correlation.

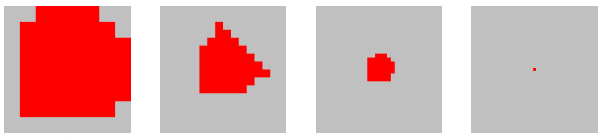


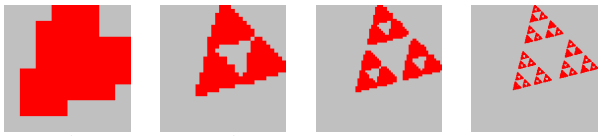
Figure 5: The evolution of the set of SPE payoff profiles in Rock-Paper-Scissors with $\gamma = 0.7$ and $\epsilon = 0.01$.

that remain in the set C by the end of the corresponding iteration. One can see in Figure 4a that when γ is sufficiently large, the algorithm maintains a set that converges towards the set F^* of feasible and individually rational payoff profiles (Mailath and Samuelson 2006), the largest possible set of SPE payoff profiles in any repeated game.

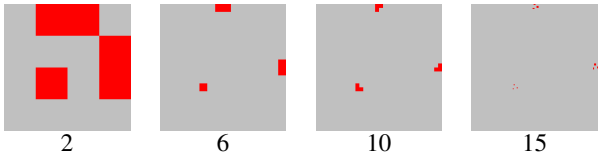
Rock, Paper, Scissors (RPS) is a symmetrical zero-sum game. In the repeated RPS game, the point $(0,0)$ is the only possible SPE payoff profile. It can be realized by a stationary strategy profile prescribing to each player to sample actions from a uniform distribution. The graphs in Figure 5 certify the correctness of the approach in this case.

Battle of the Sexes with payoff profiles $(2,1)$, $(1,2)$ and $(0,0)$ is the game that has two pure action stage-game equilibria, $(1,2)$ and $(2,1)$, and one mixed stage-game equilibrium with payoff profile $(2/3, 2/3)$. When γ is sufficiently close to 0, the set of SPE payoff profiles computed by ASPECT converges towards these three points (Figure 6b), which is the expected behavior. As γ grows, the set of SPE payoff profiles becomes larger (Figure 6a). We also observed that when the value of γ becomes sufficiently close to 1, the set of SPE payoff profiles converges towards F^* and eventually includes the point $(3/2, 3/2)$ that maximizes the Nash product (the product of players' payoffs).

In the game of Duopoly (Abreu 1988), we have made another remarkable observation: ASPECT, limited to pure strategies, preserves the point $(10, 10)$ in the set of SPE payoff profiles. (Abreu 1988) showed that this point can only be in the set of SPE payoff profiles, if $\gamma > 4/7$. Our exper-



(a) $\gamma = 0.45$, $\epsilon = 0.01$



(b) $\gamma = 0.05$, $\epsilon = 0.01$

Figure 6: The evolution of the set of SPE payoff profiles in Battle of the Sexes.

ϵ	l	Iterations	Time
0.025	0.008	55	1750
0.050	0.016	41	770
0.100	0.031	28	165
0.200	0.063	19	55
0.300	0.125	10	19
0.500	0.250	5	15

Table 1: The performance of ASPECT, extended with clustered continuations, in the repeated Battle of the Sexes.

iments confirm that. Moreover, the payoff profile $(0, 0)$ of the *optimal penal code* does also remain there. Algorithm 3 also returns an automaton that generates the payoff profile $(10, 10)$. This automaton induces a strategy profile, which is equivalent to the optimal penal code based strategy profile proposed by (Abreu 1988). To the best of our knowledge, this the first time that optimal penal code based strategies, which so far were only proven to exist (in the general case), were algorithmically computed.

Finally, the numbers in Table 1 demonstrate how different values of the approximation factor ϵ impact the performance of ASPECT for mixed action equilibria with clustered continuations in terms of (*i*) number of iterations until convergence (column 3) and (*ii*) time, in seconds, spent by the algorithm to find a solution (column 4).

Discussion

We have presented an approach for approximately computing the set of subgame-perfect equilibrium (SPE) payoff profiles and for deriving strategy profiles that induce them in repeated games. For the setting without public correlation, our ASPECT algorithm returns the richest set of payoff profiles among all existing algorithms: it returns a set that contains all stationary equilibrium payoff profiles, all non-stationary pure action SPE payoff profiles, and a subset of non-stationary mixed action SPE payoff profiles. In the presence of public correlation, our extended algorithm is capable of approximating the set of all SPE payoff profiles.

In this work, we adopted an assumption that the values of the discount factor, γ, and the hypercube side length, l, are the same for all players. ASPECT can readily be modified to incorporate player specific values of both parameters.

The linearity of the MIP problem of the CUBESUPPORTED procedure is preserved only in the two-player case. In more general cases, a higher number of players or a presence of multiple states in the environment are sources of non-linearity. This latter property, together with the presence of integer variables, require special techniques to solve the problem; this constitutes subject for future research.

References

Abreu, D.; Pearce, D.; and Stacchetti, E. 1990. Toward a theory of discounted repeated games with imperfect monitoring. *Econometrica* 1041–1063.

Abreu, D. 1988. On the theory of infinitely repeated games with discounting. *Econometrica* 383–396.

ATEJI. 2009. OptimJ – A Java language extension for optimization. `http://www.ateji.com/optimj.html`.

Cronshaw, M. 1997. Algorithms for finding repeated game equilibria. *Computational Economics* 10(2):139–168.

Graham, R. 1972. An efficient algorith for determining the convex hull of a finite planar set. *Inform. Proc. Letters* 1(4):132–133.

IBM, Corp. 2009. IBM ILOG CPLEX Callable Library Version 12.1 C API Reference Manual. `http://www-01.ibm.com/software/integration/optimization/cplex/`.

Judd, K.; Yeltekin, S.; and Conklin, J. 2003. Computing supergame equilibria. *Econometrica* 71(4):1239–1254.

Kalai, E., and Stanford, W. 1988. Finite rationality and interpersonal complexity in repeated games. *Econometrica* 56(2):397–410.

Lemke, C., and Howson, J. 1964. Equilibrium points of bimatrix games. *SIAM J. Appl. Math.* 413–423.

Littman, M., and Stone, P. 2005. A polynomial-time nash equilibrium algorithm for repeated games. *Decision Support Systems* 39(1):55–66.

Mailath, G., and Samuelson, L. 2006. *Repeated games and reputations: long-run relationships*. Oxford University Press, USA.

Nash, J. 1950. Equilibrium points in n-person games. In *Proceedings of the National Academy of the USA*, volume 36(1).

Osborne, M., and Rubinstein, A. 1999. *A course in game theory*. MIT press.

Porter, R.; Nudelman, E.; and Shoham, Y. 2008. Simple search methods for finding a Nash equilibrium. *Games Econ. Behav.* 63(2):642–662.

Saxena, A.; Bonami, P.; and Lee, J. 2008. Disjunctive cuts for non-convex mixed integer quadratically constrained programs. *Lecture Notes in Computer Science* 5035:17.