Available online at www.sciencedirect.com

ScienceDirect

Journal of Economic Theory 163 (2016) 342–364

JOURNAL OF Economic Theory

www.elsevier.com/locate/jet

Bounded computational capacity equilibrium ☆

Penélope Hernández [a], Eilon Solan [b]

[a] *ERI-CES and Departamento de Análisis Económico, Universidad de Valencia, Campus de Los Naranjos s/n, 46022 Valencia, Spain*

[b] *Department of Statistics and Operations Research, School of Mathematical Sciences, Tel Aviv University, Tel Aviv 6997800, Israel*

Received 5 September 2010; final version received 24 June 2015; accepted 13 February 2016

Available online 20 February 2016

Abstract

A celebrated result of Abreu and Rubinstein (1988) states that in repeated games, when the players are restricted to playing strategies that can be implemented by finite automata and they have lexicographic preferences, the set of equilibrium payoffs is a strict subset of the set of feasible and individually rational payoffs. In this paper we explore the limitations of this result. We prove that if memory size is costly *and* players can use mixed automata, then a folk theorem obtains and the set of equilibrium payoffs is once again the set of feasible and individually rational payoffs. Our result emphasizes the role of memory cost and of mixing when players have bounded computational power.

JEL classification: C72; C73

Keywords: Bounded rationality; Automata; Complexity; Infinitely repeated games; Equilibrium

☆ This work was conducted while the second author was visiting Universidad de Valencia. The first author thanks both the Spanish Ministry of Science and Technology and the European Feder Founds for financial support under project ECO2013-46550-R and Generalitat Valenciana PROMETEOII/2014/054. The second author thanks the Departamento de Análisis Económico at Universidad de Valencia for the hospitality during his visit. The authors thank Elchanan Ben Porath, Ehud Kalai, Ehud Lehrer, two anonymous referees, and the Associate Editor for their useful suggestions. The work of Solan was partially supported by ISF grants 212/09 and 323/13 and by the Google Inter-university center for Electronic Markets and Auctions.

E-mail addresses: Penelope.Hernandez@uv.es (P. Hernández), eilons@post.tau.ac.il (E. Solan).

http://dx.doi.org/10.1016/j.jet.2016.02.007

1. Introduction

The literature on repeated games usually assumes that players have unlimited computational capacity or unbounded rationality. Since in practice this assumption does not hold, it is important to study whether and how its absence affects the predictions of the theory.

One common way of modeling players with bounded rationality is by restricting them to strategies that can be implemented by finite state machines, also called finite automata. The game theoretic literature on repeated games played by finite automata can be roughly divided into two categories. One backed by an extensive literature (e.g., Kalai, 1990, Ben Porath, 1993, Piccione, 1992, Piccione and Rubinstein, 1993, Neyman 1985, 1997, 1998, Neyman and Okada 1999, 2000a, 2000b, Zemel, 1989) that studies games where the memory size of the two players is determined exogenously, so that each player can deviate only to strategies with the given memory size. In the other, Rubinstein (1986), Abreu and Rubinstein (1988), and Banks and Sundaram (1990) study games where the players have lexicographic preferences: each player tries to maximize her payoff, and subject to that she tries to minimize her memory size. Thus, it is assumed that memory is free, and a player would deviate to a significantly more complex strategy if that would increase her profit by one cent. Abreu and Rubinstein (1988) proved that in this case, the set of equilibrium payoffs in two-player games is generally a strict subset of the set of feasible and individually rational payoffs. In fact, it is the set of feasible and individually rational payoffs that can be generated by a *coordinated play*; that is, a sequence of action pairs in which there is a one-to-one mapping between Player 1's actions and Player 2's actions. For example, in the Prisoner's Dilemma that appears in Fig. 1, where each player has two actions, C and D, this set is the union of the two line segments $(3,3)-(1,1)$ and $(3,1)-(1,3)$.

To obtain their result, Abreu and Rubinstein (1988) make two implicit assumptions: (a) memory is costless, and (b) players can use only pure automata. Removing assumption (a) while keeping assumption (b) does not change the set of equilibrium payoffs. Indeed, since the preference of the players is lexicographic, no player can profit by deviating to a larger automaton when memory is costless, so a fortiori she has no profitable deviation when memory is costly. The construction in Abreu and Rubinstein (1988) ensures that a deviation to a smaller automaton yields the deviator a payoff which is close to her min-max value in pure strategies. Therefore, as soon as memory cost is sufficiently small, there is no profitable deviation to a smaller memory as well. We do not know whether and how the set of equilibrium payoffs changes when removing assumption (b) and keeping assumption (a).

Our goal in this paper is to show that if one removes both assumptions (a) and (b), then the result of Abreu and Rubinstein (1988) fails to hold. We will show that if memory is costly (yet memory cost goes to 0) and players can use mixed strategies, then a folk theorem obtains, and the set of equilibrium payoffs includes the set of feasible and individually rational payoffs (w.r.t. the min-max value in pure strategies). We assume for simplicity that the players have additive utility: the utility of a player is the difference between her long-run average payoff and the cost of her computational power.

We thus present a new equilibrium concept that is relevant when memory size matters and each player's set of pure strategies is the set of finite automata. For a given positive real number c, we say that the vector $x \in \mathbb{R}^2$ is a *c-Bounded Computational Capacity equilibrium payoff* (hereafter, BCC for short) if it is an equilibrium payoff when the utility of each player is the difference between her long-run average payoff, and c times the size of its finite state machine.

		Player 2	
		D	C
Player 1	D	$1,1$	$4,0$
	C	$0,4$	$3,3$

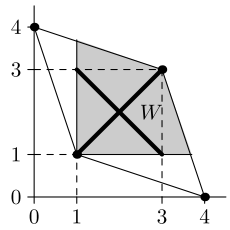

Fig. 1. The Prisoner's Dilemma: the payoff matrix, the feasible and individually rational payoffs (the dark quadrilateral W), and the payoffs that correspond to coordinated play (the two thick lines).

A payoff vector $x \in \mathbb{R}^2$ is a *BCC equilibrium payoff* if it is the limit, as c goes to 0, of c-bounded computational capacity equilibrium payoffs, and the cost of the machines used along the sequence converges to 0.

Interestingly, the definition does not imply that the set of BCC equilibrium payoffs is a subset, nor a superset, of the set of Nash equilibrium payoffs in mixed strategy of the one-shot game.

Our main result is a folk theorem: in two-player games, every feasible and individually rational (w.r.t. the min-max value in pure strategies) payoff vector is a BCC equilibrium payoff in mixed strategies of the one-shot game.

Our proof is constructive. The equilibrium play in the BCC equilibrium that we construct is composed of three phases. The first phase, that is played only once along the equilibrium path, is a punishment phase; in this phase each player plays a strategy that punishes the other player, that is, an action that attains the min-max value in pure strategies of the opponent. As in Abreu and Rubinstein (1988), it is crucial to have the punishment phase on the equilibrium path; otherwise, players can use smaller machines that cannot implement punishment, thereby reducing their computation cost. However, if a machine cannot implement punishment, there is nothing that will deter the other player from deviating. The second phase, called the babbling phase, is also played only once along the equilibrium path. In this phase the players play a predetermined sequence of action pairs. In the third phase, called the regular phase, the players repeatedly play a predetermined periodic sequence of action pairs that approximates the desired target payoff. To implement this phase, the players will use states that were used in the babbling phase. We call those states "reused states". The identity of the reused states is chosen at random at the outset of the game. The role of the babbling phase is twofold. First, it enables one to embed the regular phase within it; second, its structure is designed to simplify complexity calculations. It is long enough to ensure that to learn the states that the other player uses to implement the regular phase, a player needs a much larger automaton than the one that she currently uses. In our construction, the automaton that each player uses is *not* a best response to the automaton that the other player uses when memory cost is 0. In fact, players forgo a possible profit because to achieve this profit they need to significantly increase their memory, which is too costly.

Even though the definition of a BCC equilibrium is theoretically appealing, to prove the folk theorem we use outrageously large automata. For example, the size of the automata that we construct to approximate a target payoff vector by 0.01 is about $(100)^3$.

Our result highlights the difference between lexicographic preferences (as in Abreu and Rubinstein, 1988) and positive albeit low memory cost. When players have lexicographic preferences, they are willing to increase the memory size that they use for the profit of one cent. In particular, if the opponent's automaton reuses some states, and the knowledge of the identity of those reused states is beneficial to the player, then to learn the identity of these states the player

is willing to significantly increase her memory size. When the memory cost is positive, such an increase may not be beneficial. This observation is the key to our construction.

The rest of the paper is organized as follows. Section 2 presents the model and the main result. The proof in the particular case of the Prisoner's Dilemma is presented in Section 3. Comments and open problems appear in Section 4. In the Online Appendix (Hernández and Solan, 2016) we indicate how the proof for the Prisoner's Dilemma should be altered to fit general two-player games.

2. The model and the main result

In this section we define the model, including the concepts of automata, repeated games, and strategies implementable by automata, we describe our solution concept of Bounded Computational Capacity equilibrium, and we state the main result.

2.1. Repeated games

A two-player *repeated game* is given by (1) two finite action sets $\mathcal{A}_1$ and $\mathcal{A}_2$ for the two players, and (2) two payoff functions $u_1 : \mathcal{A}_1 \times \mathcal{A}_2 \to \mathbb{R}$ and $u_2 : \mathcal{A}_1 \times \mathcal{A}_2 \to \mathbb{R}$ for the two players.

The game is played as follows. At each stage $t \in \mathbb{N}$, each player $i \in \{1, 2\}$ chooses an action $a_i^t \in \mathcal{A}_i$ and receives the stage payoff $u_i(a_1^t, a_2^t)$. The goal of each player is to maximize her long-run average payoff $\lim_{t\to\infty} \frac{1}{t} \sum_{j=1}^{t} u_i(a_1^j, a_2^j)$, where $\{(a_1^j, a_2^j), j \in \mathbb{N}\}$ is the sequence of action pairs that were chosen by the players along the game.[1]

The set of feasible payoff vectors is

$$F := \text{conv}\{u(a), a \in \mathcal{A}_1 \times \mathcal{A}_2\}.$$

A *pure strategy* of player i is a function that assigns an action in $\mathcal{A}_i$ to every finite history $h \in \cup_{t=0}^{\infty}(\mathcal{A}_1 \times \mathcal{A}_2)^t$. A *mixed strategy* of player i is a probability distribution over pure strategies.

2.2. Automata

A common way to model a decision maker with bounded computational capacity is as an automaton, which is a finite state machine whose output depends on its current state, and whose evolution depends on the current state and on its input (see, e.g., Neyman, 1985 and Rubinstein, 1986). Formally, an *automaton* P is given by (1) a finite state space Q, (2) a finite set I of inputs, (3) a finite set O of outputs, (4) an output function $f : Q \to O$, (5) a transition function $g : Q \times I \to Q$, and (6) an initial state $q^* \in Q$.

Denote by q^t the automaton's state at stage t. The automaton starts in state $q^1 = q^*$, and at every stage $t \in \mathbb{N}$, as a function of the current state q^t and the current input i^t, the output of the automaton $o^t = f(q^t)$ is determined, and the automaton moves to a new state $q^{t+1} = g(q^t, i^t)$.

The *size* of an automaton P, denoted by $|P|$, is the number of states in Q. Below we will use strategies that can be implemented by automata; in this case the size of the automaton measures the complexity of the strategy.

[1] In general this limit need not exist. Our solution concept will take care of this issue.

2.3. Strategies implemented by automata

Fix a player $i \in \{1, 2\}$. An automaton P, whose set of inputs is the set of actions of player $3-i$ and set of outputs is the set of actions of player i, that is, $I = \mathcal{A}_{3-i}$ and $O = \mathcal{A}_i$, can implement a pure strategy of player i. Indeed, at every stage t, the strategy plays the action $f(q^t)$, and the new state of the automaton $q^{t+1} = g(q^t, a^t_{3-i})$ depends on its current state q^t and on the action a^t_{3-i} that the other player played at stage t. For $i = 1, 2$, we denote an automaton that implements a strategy of player i by P_i. We denote by $\mathcal{P}_i^m$ the set of all automata with m states that implement pure strategies of player i.

When the players use arbitrary strategies, the long-run average payoff needs not exist. However, when both players use strategies that can be implemented by automata, say P_1 and P_2 of sizes p_1 and p_2 respectively, the evolution of the automata follows a (deterministic) Markov chain with $p_1 \times p_2$ states, and therefore the long-run average payoff exists. We denote this average payoff by $\gamma(P_1, P_2) \in \mathbb{R}^2$.

A mixed automaton M is a probability distribution over pure automata.[2] A mixed automaton corresponds to the situation in which the automaton that is used is not known, and there is a belief over which automaton is used. A mixed automaton defines a mixed strategy: at the outset of the game, a pure automaton is chosen according to the probability distribution given by the mixed automaton, and the strategy that the pure automaton defines is executed. We will use only mixed automata whose support is pure automata of a given size m.

When both players use mixed strategies that can be implemented by mixed automata, the expected long-run average payoff exists; it is the expectation of the long-run average payoff of the pure automata that the players play:

$$\gamma(M_1, M_2) := \mathbf{E}_{M_1, M_2}[\gamma(P_1, P_2)].$$

2.4. Bounded computational capacity equilibrium

In the present paper we study games where the utility function of each player takes into account the complexity of the strategy that she uses.

Definition 1. Let $c > 0$. A pair of mixed automata (M_1, M_2) is a *c-BCC equilibrium*, if it is a Nash equilibrium for the utility functions $U_i^c(M_1, M_2) := \gamma_i(M_1, M_2) - c|M_i|$, for $i \in \{1, 2\}$.

If the game has an equilibrium in pure strategies, then the pair of pure automata (P_1, P_2), both with size 1, which repeatedly play the equilibrium actions of the two players, is a c-BCC equilibrium, for every $c > 0$.

The min-max value of player i in pure strategies is

$$v_i := \min_{a_{3-i} \in \mathcal{A}_{3-i}} \max_{a_i \in \mathcal{A}_i} u_i(a_i, a_{3-i}).$$

An action a_{3-i} that attains the minimum is termed a *punishing* action of player $3-i$. The set of *strictly individually rational payoff vectors* (relative to the min-max value in pure strategies) is

$$V := \{x = (x_1, x_2) \in \mathbb{R}^2 \colon x_1 > v_1, x_2 > v_2\}.$$

[2] To emphasize the distinction between automata and mixed automata, we call the former *pure automata*.

To get rid of the dependency of the constant c we define the concept of a *BCC equilibrium payoff*. A payoff vector x is a *BCC equilibrium payoff* if it is the limit, as c goes to 0, of payoffs that correspond to c-BCC equilibria.

Definition 2. A payoff vector $x = (x_1, x_2)$ is a *BCC equilibrium payoff* if for every $c > 0$ there is a c-BCC equilibrium $(M_1(c), M_2(c))$ such that $\lim_{c\to 0} \gamma(M_1(c), M_2(c)) = x$ and $\lim_{c\to 0} c|M_i(c)| = 0$ for $i = 1, 2$.

Every pure equilibrium payoff is a BCC equilibrium payoff (implemented by automata of size 1). Using Abreu and Rubinstein's (1988) proof, one can show that any strictly individually rational payoff (relative to the min-max value in pure strategies) that can be generated by coordinated play is a BCC equilibrium payoff. For the formal statement, assume w.l.o.g. that $|\mathcal{A}_1| \leq |\mathcal{A}_2|$.

Theorem 3. *(See Abreu and Rubinstein, 1988.) Let $\sigma : \mathcal{A}_1 \to \mathcal{A}_2$ be a one-to-one function. Then any payoff vector x in the convex hull of $\{u(a_1, \sigma(a_1)), a_1 \in \mathcal{A}_1\}$ that satisfies $x_i > v_i$ for $i = 1, 2$ is a BCC equilibrium payoff.*

2.5. The main result

Our main result is the following folk theorem, which states that every feasible and strictly individually rational payoff vector is a BCC equilibrium payoff.

Theorem 4. *If the set $F \cap V$ has a nonempty interior, then every vector in $F \cap V$ is a BCC equilibrium payoff.*

Theorem 4 is not a characterization of the set of BCC equilibrium payoffs, because it does not rule out the possibility that a feasible payoff that is not individually rational (relative to the min-max value in pure strategies) is a BCC equilibrium payoff. That is, we do not know whether threats of punishments by a mixed strategy in the one-shot game can be implemented in a BCC equilibrium.

Theorem 4 stands in sharp contrast to the main message of Abreu and Rubinstein (1988) where it is proved that lexicographic preferences, which are equivalent to an infinitesimal cost function c, imply that in equilibrium players follow coordinated play, so that the set of equilibrium payoffs is often strictly smaller than the set of feasible and individually rational payoffs. Our study shows that the result of Abreu and Rubinstein (1988) hinges on two assumptions: (a) memory is costless, and (b) the players use only pure automata. Once we assume that memory is costly and that players may use mixed automata, the set of equilibrium payoffs dramatically changes.

2.6. A detour to Abreu and Rubinstein (1988)

Abreu and Rubinstein (1988) study repeated games in which players have lexicographic preferences and can use only pure automata. They consider both the undiscounted game and the discounted game with a discount factor that is close to 1. A pair of pure automata is an *equilibrium* if (a) no player can profit by deviating to any other pure automaton, and (b) a player who deviates to a smaller automaton loses.

Abreu and Rubinstein (1988) prove that the set of equilibrium payoffs is the set of feasible and individually rational payoff vectors that can be generated by a coordinated play.

In the Prisoner's Dilemma (see Fig. 1) the min-max level of each player is 1, and the punishing action of each player is D. The set of feasible and (weakly) individually rational payoffs is the quadrilateral W with extreme points $(1,1)$, $(1,3\frac{2}{3})$, $(3,3)$ and $(3\frac{2}{3},1)$ (see Fig. 1). The result of Abreu and Rubinstein implies that the set of equilibrium payoffs is the union of the two line segments $(1,1)-(3,3)$ and $(1,3)-(3,1)$.

The argument leading to the result of Abreu and Rubinstein's (1988) are the following.

1. When Player 1 uses an automaton with m states, Player 2's optimization problem reduces to a Markov decision problem with m states, and therefore Player 2's best response is an automaton with at most m states. This implies that in an equilibrium both players use automata of the same size.
2. Each player's equilibrium automaton uses distinct states until it completes one cycle of its states. This follows from a result that says that if the states of one player that are used in any two periods t and t' of equilibrium play are identical, then the average payoff of the opponent between stages t to t' coincides with the average payoff from t' onwards, and therefore also the average payoff from t onwards.[3] Therefore if the cycle starts before all the states for both player are used, each player could modify her machine to skip the stages between stages t and t', thereby lowering the size of her automaton without affecting the long-run average payoff.
3. If in stage t the automaton P_i plays the same action it plays in stage t', then in stage t the automaton P_{3-i} plays the same action it plays in stage t'. Indeed, by Point 2, the automaton P_i uses different states in stages t and t', and these two states are not used in other stages along the cycle. If the automaton P_{3-i} plays differently in stages t and t', then player i can lower the size of her automaton by using the same state in stages t and t', and letting the action of player $3-i$ control the transition out of this state.

Abreu and Rubinstein's equilibrium construction is as follows.

- The players start by implementing a *punishment phase*: both players play the action D for a large number of stages. The states used for this phase are all distinct. Moreover those states are used only at the beginning of the equilibrium play.
- A cycle of action pairs, which is called the *regular phase*, is repeated. The states used in the cycle are distinct from those used during the punishment phase, and are used infinitely many times. Each of those states leads to the first state in the punishment phase if it detects a deviation. The action pairs of the cycle form a coordinated play. This implies that there exists a one-to-one relationship between the action set of Players 1 and 2 in equilibrium.

3. An example

In this section we present and explain the proof of Theorem 4 in the context of the Prisoner's Dilemma. This construction will be formalized in Appendix B and extended to any game in the Online Appendix.

[3] This result is Lemma 2, page 1268, in Abreu and Rubinstein (1988).

Consider the Prisoner's Dilemma game that appears in Fig. 1. We show how to implement the payoff vector $x = (\frac{7}{6}, \frac{19}{6})$ as a BCC equilibrium payoff. This vector can be written as a convex combination of three vectors in the payoff matrix, for example,

$$(\tfrac{7}{6}, \tfrac{19}{6}) = \tfrac{1}{6}(1,1) + \tfrac{2}{6}(3,3) + \tfrac{3}{6}(0,4). \tag{1}$$

The construction depends on a natural number k, that will determine the size of the automata that the players use. This number gets larger as (a) the memory cost decreases, and (b) the target payoff vector x and the actual payoff on the equilibrium path become closer.

The equilibrium play will consist of three phases, as follows.

- A *punishment phase* that consists of k^3 times playing (D, D):

$$Q^* := k^3 \times (D, D).$$

- A *babbling phase* that consists of $2k$ blocks of length k followed by one block of length $k+1$: in odd blocks (except the last one) the players play k times (C, C); in even blocks they play k times (D, D); and in the last block the players play $k+1$ times (C, C).

$$B^* := \sum_{n=1}^{k} \big(k \times (C, C) + k \times (D, D)\big) + (k+1) \times (C, C).$$

- A *regular phase* in which the players repeatedly play actions along which the average payoff is the target payoff x.

$$R^* := 1 \times (D, D) + 2 \times (C, C) + 3 \times (C, D).$$

Formally, the equilibrium play path ω^* is

$$\begin{aligned} \omega^* &:= Q^* + B^* + \sum_{n=1}^{\infty} R^* \\ &= \underbrace{k^3 \times (D, D)}_{\text{Punishment}} + \underbrace{\sum_{n=1}^{k} \big(k \times (C, C) + k \times (D, D)\big) + (k+1) \times (C, C)}_{\text{Babbling}} + \underbrace{\sum_{n=1}^{\infty} R^*}_{\text{Regular}}. \end{aligned} \tag{2}$$

To implement other feasible and individually rational payoff vectors x as BCC equilibria we change the regular phase to contain a cycle of action pairs whose average payoff is close to x.

The roles of the three phases are as follows.

- As in Abreu and Rubinstein (1988), the punishment phase ensures that punishment is on the equilibrium path. Because the players minimize their automaton size, subject to maximizing their payoff, if the punishment phase was off the equilibrium path, players could save states by not implementing it. But if a player cannot implement punishment, the other player may safely deviate, knowing that she will not be punished. In our construction, detectable deviations of the other player will lead the automaton to restart and reimplement ω^*, thereby initiating a long punishment phase. The length of the punishment phase, k^3, is much larger than the length of the babbling phase to ensure that the punishment is severe.

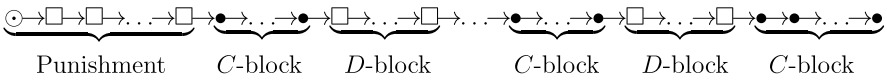

Fig. 2. An implementation of ω_1.

- The babbling phase serves two purposes. First, because it is coordinated, it is not difficult to calculate the *complexity* of ω^* for each player i, that is, the size of the minimal pure automaton of player i that can implement player i's part in ω^*, given that the other player, player $3-i$, plays her part in ω^*. This implies in particular that if a player deviates to a smaller automaton than the one that we will construct, while the other player does not deviate, then there will be a stage in which that player's play deviates from ω^*.
  Second, the states that implement the babbling phase will be reused to implement the regular phase. The identity of the states that are reused will be chosen at random; this is the place where we rely on our usage of mixed automata. In our construction, any deviation in a state that is not reused to implement the regular phase starts a punishment phase. This implies that to profit by a deviation, the player needs to know which states are reused in his opponent's automaton. Since the reused states are chosen at random, learning which states are reused requires a huge automaton, which, due to the memory cost, is too costly.
- On the equilibrium path the regular play will be played repeatedly, so that the long-run average payoff will be the average payoff along R^*, which is $(\frac{7}{6}, \frac{19}{6})$.

We say that a pure automaton P_i of player i is *compatible* with the play ω^* (or that the play ω^* is compatible with the automaton P_i) if, when the other player $3-i$ plays her part in ω^*, the automaton generates the play of player i in ω^*. We will later show that the size of the smallest automaton of Player 1 (resp. Player 2) that is compatible with ω^* is k^3+2k^2+k+1 (resp. k^3+2k^2+k+4), see Corollary 7 (resp. Corollary 8) below.

We now present an automaton for Player 1 with size k^3+2k^2+k+1 that is compatible with ω^*. Denote the states of the automaton that we construct by $Q=\{1,2,\cdots,k^3+2k^2+k+1\}$. The punishment and babbling phases, whose total length is k^3+2k^2+k+1, are

$$\omega_1 = k^3\times(D,D)+\sum_{n=1}^{k}\big(k\times(C,C)+k\times(D,D)\big)+(k+1)\times(C,C).$$

The length of these phases is similar to the size of the automaton that we construct. A naive implementation is to have one state for each action of Player 1 in ω_1: state $q\in Q$ will implement the q'th action pair in ω_1. This implementation is illustrated in Fig. 2, where the initial state is the dotted circle to the left, the white squares correspond to states where the action played is D, and the black circles correspond to states where the action played is C.

It is left to implement the regular phase R^*, in which Player 1 plays once D and 5 times C. One way to do this is as follows (see Fig. 3): When Player 1's automaton is in its last state, state k^3+2k^2+k+1, and Player 2 plays C, Player 1's automaton moves to the last state of the first D-block in the Babbling phase. If Player 2 does not deviate, then the play in the next six stages will indeed be $1\times(D,D)+2\times(C,C)+3\times(C,D)$. In the fifth stage of the following C-block we add a transition that ensures that the regular phase will be repeated: When Player 1's automaton is in the fifth stage of the second C-block and Player 2 plays D, Player 1's automaton moves to the last stage of the first D-block. Thus, three states in Player 1's automaton accept both actions of Player 2: the third, the fourth, and the fifth states of the second C-block. The third and

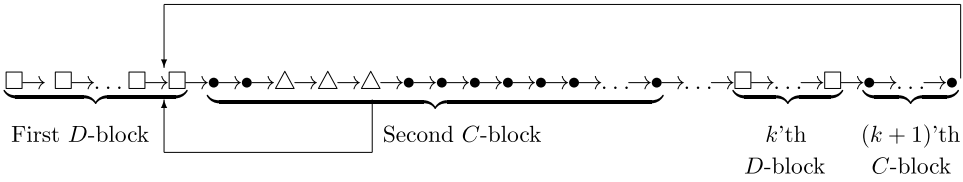

Fig. 3. Implementation of the regular phase.

fourth states lead deterministically to the following state, and the fifth state either continues to the sixth state of the second C-block (if Player 2 plays C) or restarts the regular phase (if Player 2 plays D). We call these three states *accept-all states*. In Fig. 3, the three accept-all states are denoted by triangles.

To ensure that deviations of Player 2 are not profitable, we set all transitions that were not determined so far to initiate a punishment phase, by making Player 1's automaton move to the first state.

Analogously we will define an automaton for Player 2 with $k^3 + 2k^2 + k + 4$ states that is compatible with ω^*. Because the equilibrium play is not a coordinated play, by Abreu and Rubinstein (1988) Player 2 has a profitable deviation. Indeed, Player 2 may skip most of the babbling phase, thereby she reduces the number of states in her automaton while still implementing the same targeted payoff. This can be done as follows. Player 2 uses an automaton with $k^3 + 2k + 5$ states. These states implement naively Player 2's part in the sequence

$$k^3 \times (D, D) + k \times (C, C) + k \times (D, D) + 2 \times (C, C) + 3 \times (C, D),$$

and from the last state the automaton moves to the last state of the D-block.[4]

To be able to execute the deviation described in the previous paragraph, Player 2 must know the identity of the accept-all states in Player 1's automaton.[5] To make this deviation unprofitable Player 1 has to mask the identity of these states.

To achieve this goal, we note that the automaton that we described is only one automaton for Player 1 that is compatible with ω^*. Instead of using the last state of the first D-block and the first five states of the second C-block to implement the regular phase, we could have used the last state of the j'th D-block and the first five states of the $(j+1)$'th C-block for $1 \leq j \leq k-1$. More generally,[6] we could have used the last state of the j'th D-block, the first *three* states of the $(j+1)$'th C-block, and two additional states in the $(j+1)$'th C-block, say, states number h_1 and h_2 (see Fig. 4). The accept-all states would be the third, h_1, and h_2 states of the $(j+1)$'th C-block. When Player 1's automaton is in the first accept-all state (the third state of the $(j+1)$'th C-block) and Player 2 plays D, the automaton will move to the h_1'th state of the $(j+1)$'th C-block; when Player 1's automaton is in the h_1'th state of the $(j+1)$'th C-block and Player 2 plays D, the automaton will move to the h_2'th state of the $(j+1)$'th C-block; and when Player 1's automaton is in the h_2'th state of the $(j+1)$'th C-block and Player 2 plays D, the automaton will move to the last state of the j'th D-block, thereby start a new cycle of the regular phase.

Recognizing that there are many pure automata for Player 1 that are compatible with ω^*, we define a *mixed* automaton for Player 1, which chooses one of these pure automata at random.

[4] Player 2 could deviate to an even smaller automaton to implement this deviation.

[5] In the construction that we described, it is sufficient for Player 2 to know the identity of the third accept-all state.

[6] There are additional pure automata for Player 1 that implement ω^*. We will not use them in our construction.

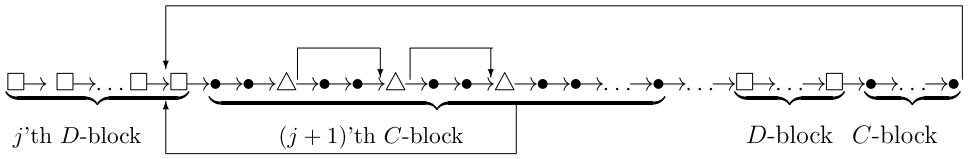

Fig. 4. The j'th D-block and the $(j+1)$'th C-block in P_1.

This ensures that Player 2 will not know the identity of the accept-all states of the realized pure automaton of Player 1.

Similarly, we will define a collection of pure automata for Player 2 with k^3+2k^2+k+4 states that implements Player 2's part of the play ω^* that reuse different states, and a mixed automaton for Player 2 in which the reused states are chosen randomly.

Can Player 2 profit by deviating from her part of ω^* when she faces the mixed automaton of Player 1? The answer is positive: she could enumerate over the three parameters that were chosen at random, namely, j, h_1, and h_2, and for each possible values of these parameters check whether they are the actual values chosen by Player 1.

The only way in which Player 2 can check whether $j = j^*$, $h_1 = h_1^*$, and $h_2 = h_2^*$ is to play D in states number h_1 and h_2 of the $(j+1)$'th C-block, and observe the actions that Player 1 plays in the following stages. If the parameters of Player 1's realized pure automaton are not (j^*, h_1^*, h_2^*), then its automaton will restart, initiating a punishment phase of length k^3. We will show below that for each triplet (j^*, h_1^*, h_2^*) on which Player 2 enumerates, her automaton must devote *distinct* k^3 states to pass the punishment phase (Lemma 15 below). Once Player 2 identified the correct triplet (j^*, h_1^*, h_2^*), she can use this information to increase her average payoff.

If memory were costless, such a deviation would be profitable: When memory is costly this is not necessarily the case. In our construction, the number of pure automata in the support of Player 1's mixed automaton is $O(k)$. Therefore, to learn the parameters (j, h_1, h_2) that Player 1 uses with a nonnegligible probability, say ε, Player 2 needs an automaton of size $O(\varepsilon k^4)$, whose cost is $O(c\varepsilon k^4)$, where c is the cost of each memory cell. Since payoffs are bounded, such a deviation leads to a profit of $O(\varepsilon)$. The size of Player 2's automaton that we described above is $O(k^3)$. Consequently, if $c = O(k^{-3.5})$, the cost of the automaton of Player 2, whose size is $k^3 + 2k^2 + k + 4$, vanishes as k goes to infinity, while the cost of the automaton that with a nonnegligible probability learns the parameters (j, h_1, h_2) goes to infinity. This implies that Player 2 cannot profit by deviating to a larger automaton for certain memory cost.

The discussion in the previous paragraph implies that the two mixed automata that we construct are *not* best response to each other when memory is costless; They are best response to each other when the memory cost is $c = O(k^{-3.5})$.

4. Comments and open problems

4.1. The discounted game

One could study variations of the definition of BCC equilibrium when using the discounted payoff instead of the long-run average payoff.

Given $c > 0$ and a discount factor $\lambda \in (0,1)$, a pair of mixed automata (M_1, M_2) is a (c,λ)-*BCC equilibrium payoff* if it is a Nash equilibrium for the utility functions $U_i^{c,\lambda}(M_1, M_2) =$

$\gamma_i^\lambda(M_1, M_2) - c|M_i|$ for $i = 1, 2$, where $\gamma_i^\lambda(M_1, M_2)$ is the λ-discounted payoff of player i when the players use the mixed automata (M_1, M_2). A vector $x \in \mathbb{R}^2$ is a BCC equilibrium payoff if it is the limit, as c goes to 0 and λ goes to 1, of payoffs that correspond to (c, λ)-BCC equilibria. That is, there is a sequence $(c_n)_{n\in\mathbb{N}}$ and $(\lambda_n)_{n\in\mathbb{N}}$ that converge to 0 and 1, respectively, and for each n there is a (c_n, λ_n)-BCC equilibrium $(M_1^{c_n,\lambda_n}, M_2^{c_n,\lambda_n})$, such that $\lim_{n\to\infty} \gamma^\lambda(M_1^{c_n,\lambda_n}, M_2^{c_n,\lambda_n}) = x$ and $\lim_{n\to\infty} c_n|M_i^{c_n,\lambda_n}| = 0$ for $i = 1, 2$.

Our folk theorem holds for this concept, with the same construction.

4.2. A more general definition of a BCC equilibrium

The definition of the concept of c-BCC equilibrium assumes that the utility of each player is additive, and that the memory cost is linear in the memory size. There are applications where the utility function U_i has a different form.

- Players may disregard the memory cost, but be bounded by the size of memory that they use.

$$U_i(M_1, M_2) = \begin{cases} \gamma_i(M_1, M_2) & |M_i| \le k_i, \\ -\infty & |M_i| > k_i. \end{cases}$$

  This situation occurs, e.g., when players are willing to invest huge amounts of money even if the profit is low, but the available technology does not allow them to increase their memory size beyond some limit. Such a situation may occur, e.g., in the area of code breaking, where countries invest large sums of money to be able to increase the number of other countries' codes that they break, yet they are bounded by technological advances.
- Memory is costly, yet players do not save money by reducing their memory size. That is, a pair of mixed automata (M_1, M_2) is a c-BCC equilibrium if for each player $i \in \{1, 2\}$ and for every pure automaton $P_i \in \text{supp}(M_i)$ one has $\gamma_i(M_i, M_{3-i}) = \gamma_i(P_i, M_{3-i})$, and, if $|P_i| > |M_i|$, one has $\gamma_i(M_i, M_{3-i}) \ge \gamma_i(P_i, M_{3-i}) - c(|P_i| - |M_i|)$. This situation occurs, e.g., when the players are organizations whose size cannot be reduced.

It may be of interest to study the set of equilibrium payoffs for various utility functions U_i, and to see whether and how this set depends on the shape of this function.

4.3. More than two players

The concept of BCC equilibrium payoff is valid for games with any number of players. However, Theorem 4 holds only for two-player games. One crucial point in our construction is that if a deviation is detected, a player is punished for a long (yet finite) period of time by a punishing action. When there are more than two players, the punishing action of, say, Player 1 against Player 2 may be different that the punishing action of Player 1 against Player 3. It is not clear how to construct an automaton that can punish each of the other players, if necessary, and such that all these memory cells will be used on the equilibrium path.

Appendix A. The complexity of a sequence of action pairs

In this section we provide tools to calculate lower bounds to the complexity of sequences and we prove that the complexity of ω^*, which is defined in (2), w.r.t. each of the players is at least the quantities given in Corollaries 7 and 8 below.

When $\omega = (\omega(t))_t$ is a (finite or infinite) sequence of action pairs, we denote by $\text{comp}_i(\omega)$ the complexity of ω w.r.t. player i. We denote by $\omega_i(t)$ player i's action at time t at ω. When P_i is an automaton of player i, we denote by $q_i(t)$ the state of P_i at time t.

The following lemma lists several simple observations that we will use in the sequel. The first property says that if the action that P_i plays in stage t_1 differs from the action it plays in stage t_2, then in those stages it is in different states. The second property says that if P_i is in different states in stages t_1+1 and t_2+1, and if the action pair played in stage t_1 equals the action pair that is played in stage t_2, then P_i must have been in different states already in stages t_1 and t_2. The third property is a generalization of the second property: if P_i is in different states in stages t_1+m and t_2+m, and if the action pair played in stage t_1+l equals the action pair that is played in stage t_2+l for $l \in \{0,1,\ldots,m-1\}$, then P_i must have been in different states already in stages t_1 and t_2. The fourth property says that the complexity of a finite sequence of action pair w.r.t. Player 1 is independent of the action that Player 2 plays in the last stage.

Lemma 5. *Let P_i be a pure automaton of player i that is compatible with ω.*

1. *If $\omega_i(t_1) \neq \omega_i(t_2)$ then $q_i(t_1) \neq q_i(t_2)$.*
2. *If $q_i(t_1+1) \neq q_i(t_2+1)$ and $\omega(t_1) = \omega(t_2)$, then $q_i(t_1) \neq q_i(t_2)$.*
3. *If $q_i(t_1+m) \neq q_i(t_2+m)$ and $\omega(t_1+l) = \omega(t_2+l)$ for every $l \in \{0,1,\ldots,m-1\}$, then $q_i(t_1) \neq q_i(t_2)$.*
4. *If $\omega = (\omega(t))_{t=1}^T$ and $\omega' = (\omega'(t))_{t=1}^T$ are two finite sequences that differ only in the action of Player 2 at stage T, that is, $\omega_i(t) = \omega'_i(t)$ for every $t \in \{1,2,\ldots,T\}$ and every $i \in \{1,2\}$, except $t = T$ and $i = 2$, then the complexity of ω w.r.t. Player 1 is equal to the complexity of ω' w.r.t. Player 1.*

Proof. The first claim holds since the automaton's output is a function of the automaton's state. The second claim follows since the new state of the automaton is a function of the current state and of the other player's action. The third claim follows from the second claim by induction. The fourth claim follows since for a finite sequence, the action of Player 2 in the last stage T does not affect the evolution of the automaton of Player 1 in the first T stages. □

A (finite or infinite) sequence of action pairs $\omega = (\omega(t))_t$ is *coordinated* if $\omega_1(t) = \omega_1(t')$ if and only if $\omega_2(t) = \omega_2(t')$, for every $t \neq t'$. The following result follows from Neyman (1998).

Lemma 6. *Let $\omega = (\omega(t))_{t=1}^T$ be a coordinated sequence of action pairs and let $T_0 \leq T$. If $(\omega(t))_{t=t_2}^T$ is not a prefix of $(\omega(t))_{t=t_1}^T$ for every $t_1 < t_2 \leq T_0$, then $\text{comp}_i(\omega) \geq T_0$ for each player i.*

Proof. Assume to the contrary that the condition of the lemma holds but there is a pure automaton for player i with size less than T_0 that is compatible with ω. By the pigeon hole principle, there are $t_1 < t_2 \leq T_0$ such that $q_i(t_1) = q_i(t_2)$. By Lemma 5(1), $\omega_i(t_1) = \omega_i(t_2)$, and since ω is coordinated we have $\omega_{3-i}(t_1) = \omega_{3-i}(t_2)$. It follows by Lemma 5(2) that $q_i(t_1+1) = q_i(t_2+1)$.

Continuing inductively we deduce that $q_i(t_1+l)=q_i(t_2+l)$ for every l for which $t_2+l\le T$. This implies that $(\omega(t))_{t=t_2}^{T}$ is a prefix of $(\omega(t))_{t=t_1}^{T}$, a contradiction. □

We can now calculate the complexity of the sequence ω^* defined in Eq. (2) w.r.t. both players.

Corollary 7. $\mathrm{comp}_1(\omega^*)\ge k^3+2k^2+k+1$.

Proof. The definition of the complexity of a sequence implies that the complexity of a sequence cannot be lower than the complexity of any of its subsequences. Consider then the prefix ω' of length $T=k^3+2k^2+k+3$ of ω^*, which involves only a coordinated play. For this sequence the condition in Lemma 6 is satisfied for ω^* with $T_0=k^3+2k^2+k+1$, and therefore $\mathrm{comp}_1(\omega')\ge k^3+2k^2+k+1$, as desired. □

Corollary 8. $\mathrm{comp}_2(\omega^*)\ge k^3+2k^2+k+4$.

Proof. Consider the prefix ω' of ω^* of length $T=k^3+2k^2+k+4$. Let ω'' be the sequence ω' after adding the action pair (D,D) at the end, and let ω''' be the sequence ω' after adding the action pair (C,D) at the end. Note that ω''' is a prefix of ω^*, hence $\mathrm{comp}_2(\omega^*)\ge\mathrm{comp}_2(\omega''')$. By Lemma 5(4), $\mathrm{comp}_2(\omega''')=\mathrm{comp}_2(\omega'')$. Apply Lemma 6 to the sequence ω'' with $T_0=k^3+2k^2+k+4$ to deduce that $\mathrm{comp}_2(\omega'')\ge k^3+2k^2+k+4$. The result follows. □

Appendix B. BCC equilibria in the prisoner's dilemma

In the present section the construction described in Section 3 is provided formally, and we prove that it forms a BCC equilibrium. The construction in this case contains all ingredients and complexities of the construction in the general case, yet, because the regular phase is short, there is no need to carry many indices and execute complex computations. In the Online Appendix we generalize this construction to any two-player repeated game.

Consider then the payoff vector

$$x=(x_1,x_2)=\left(\tfrac{7}{6},\tfrac{19}{6}\right)=\tfrac{1}{6}(1,1)+\tfrac{2}{6}(3,3)+\tfrac{3}{6}(0,4). \tag{3}$$

Our construction depends on a parameter k that determines the size of the automata that the players use: Player 1 mixes between pure automata of size k^3+2k^2+k+1 and Player 2 mixes between pure automata of size k^3+2k^2+k+4. Let $k\ge 36$; to facilitate calculations we assume that k is divisible by 4. In particular, the following inequalities, which will be used below, hold: $\min\{x_1-1,x_2-1\}>\frac{6}{k}$ and $k^3>3k^2+2k+8$.

As mentioned before, the equilibrium play will be

$$\omega^*=\underbrace{k^3\times(D,D)}_{\text{Punishment}}+\underbrace{\sum_{n=1}^{k}\big(k\times(C,C)+k\times(D,D)\big)+(k+1)\times(C,C)}_{\text{Babbling}}+\underbrace{\sum_{n=1}^{\infty}R^*}_{\text{Regular}}.$$

B.1. An automaton P_1 for player 1 that is compatible with ω^*

Fix $j\in\{1,2,\dots,k-1\}$ and $h_1,h_2\in\{4,5,\dots,k\}$ such that $h_1\ne h_2$. In this section we provide the formal definition of the pure automaton $P_1=P_1^{j,h_1,h_2}$ for Player 1 with size k^3+2k^2+k+1 that is compatible with ω^* and was described in Section 3.

Denote the states of P_1 by the integers $Q = \{1, 2, \ldots, k^3 + 2k^2 + k + 1\}$, where $q^* = 1$ is the initial state. Divide Q into three sets:

1. $Q^P = \{1, 2, \ldots, k^3\}$ is the set of all states that implement the punishment phase.
2. $Q^C = \left(\bigcup_{n=0}^{k-1}\{k^3 + 2nk + 1, \ldots, k^3 + 2nk + k\}\right) \cup \{k^3 + 2k^2 + 1, \ldots, k^3 + 2k^2 + k + 1\}$ is the set of states in all *C-blocks*.
3. $Q^D = \bigcup_{n=0}^{k-1}\{k^3 + 2nk + k + 1, \ldots, k^3 + 2nk + 2k\}$ is the set of states in all *D-blocks*.

The output function is

$$f(q) = \begin{cases} D & q \in Q^P \cup Q^D, \\ C & q \in Q^C, \end{cases}$$

and the transition function is as follows (see Figs. 2 and 4):

- As long as Player 2 complies with her part of ω^*, the automaton P_1 advances from each state to the following one:

$$g(q, f(q)) = q + 1, \quad 1 \le q < k^3 + 2k^2 + k + 1.$$

- When P_1 is at the last state and Player 2 plays C, the automaton moves to the last state of the j'th D-block:

$$g(k^3 + 2k^2 + k + 1, C) = k^3 + 2jk.$$

- When P_1 is at the third state of the $(j+1)$'th C-block and Player 2 plays D, the automaton moves to state h_1 of the $(j+1)$'th C-block:

$$g(k^3 + 2jk + 3, D) = k^3 + 2jk + h_1.$$

- When P_1 is at state h_1 of the C-block and Player 2 plays D, the automaton moves to state h_2 of the $(j+1)$'th C-block:

$$g(k^3 + 2jk + h_1, D) = k^3 + 2jk + h_2.$$

- When P_1 is at state h_2 of the C-block and Player 2 plays D, the automaton moves to last state of the j'th D-block:

$$g(k^3 + 2jk + h_2, D) = k^3 + 2jk.$$

- All transitions that were not defined above lead to state 1, thereby initiating a punishment phase.

B.2. A mixed automaton $M_1 = M_1(k)$

A mixed automaton M_i of player i is *compatible* with ω^* if all the pure automata in its support are compatible with ω^*.

The pure automaton $P_1 = P_1(j, h_1, h_2)$ that was constructed in Section B.1 depends on three parameters: j, h_1, and h_2. If Player 2 learns the three parameters or a subset thereof, she may have a profitable deviation, either by decreasing the size of her automaton or by implementing a payoff greater than x_2.

D1. As discussed in Section 3, if Player 2 knows j, then she can decrease the size of her automaton by skipping part of the babbling phase.

D2. If Player 2 knows h_1, she would know the distance between the first and second reused states in the $(j+1)$'th C-block. In particular, if instead of following her part in the regular phase $D+2\times C+3\times D$ she would play $D+(2+h_1)\times C+2\times D$, she would create the cycle

$$2\times(D,D)+(2+h_1)\times(C,C)+2\times(C,D),$$

which yields to her (and to Player 1 as well) a higher average payoff.

D3. Similarly, if Player 2 knew h_2 or h_2-h_1, then she could profit by an appropriate deviation in the regular phase.

This discussion implies that Player 1 must mask the parameters j, h_1, and h_2 that she uses. This is done by defining a mixed automaton $M_1=M_1(k)$, which chooses these parameters randomly.

Let $\mathcal{H}=\{(j^d,h_1^d,h_2^d)\colon 1\le d\le \frac{k}{4}\}$ be a collection of $\frac{k}{4}$ triplets that satisfy the following conditions:

A1 $(j^d)_{d=1}^{k/4}$ are distinct elements from $\{1,2,\dots,k-1\}$ and $(h_1^d,h_2^d)_{d=1}^{k/4}$ are distinct elements from $\{4,5,\dots,k\}$.

A2 $h_2^{d_1}-h_1^{d_1}\neq h_2^{d_2}-h_1^{d_2}$ for every distinct $d_1,d_2\in\{1,2,\dots,\frac{k}{4}\}$.

One can define, e.g., $j^d=d$, $h_1^d=3+d$ and $h_2^d=h_1^d+\frac{k}{4}+d$ for every $d\in\{1,2,\dots,\frac{k}{4}\}$.

The mixed automaton $M_1=M_1(k)$ chooses uniformly one of the pure automata in $\mathcal{P}_1:=\{P_1^{j,h_1,h_2},(j,h_1,h_2)\in\mathcal{H}\}$. In particular, all pure automata in the support of M_1 are compatible with ω^* for Player 1, so that M_1 is compatible with ω^* for Player 1 as well.

The most significant implication of Properties (A1)–(A2) is the following. Player 2 may face any of the $\frac{k}{4}$ pure automata in $\mathcal{P}_1$. To deviate, the play of Player 2 must differ from ω^*. Properties (A1) and (A2) ensure that if Player 2 deviates from ω^*, then all pure automata in $\mathcal{P}_1$, except possibly one, will restart within $2k^2+k+1$ stages. That is, with probability close to 1, a deviation from ω^* starts a punishment phase. This observation is the content of the following result.

Lemma 9. *Let $P_1=P_1^{j,h_1,h_2}$ and $P_1'=P_1^{j',h_1',h_2'}$ be two different pure automata in the support of M_1 and let P_2 be any pure automaton of Player 2. Let t be the first stage in which the play under (P_1,P_2) differs from ω^*. Then at least one of the automata P_1 and P_1' restarts before stage $t+2k^2+k+1$.*

Note that since both P_1 and P_1' are compatible with ω^*, the first stage in which the play under (P_1',P_2) differs from ω^* is also t. The Lemma is valid for any strategy of Player 2, not necessarily those implementable by pure automata.

Proof of Lemma 9. Denote by $q(t)$ (resp. $q'(t)$) the state of the automaton P_1 (resp. P_1') at stage t when facing P_2. Denote by $\omega^*(t)$ the action pair at stage t according to ω^*. Then $\omega_2^*(t)$ is the action that Player 2 is supposed to play at stage t according to ω^*.

Since P_1 and P_1' are compatible with ω^*, and since in stage t the play under (P_1,P_2) differs from ω^*, it follows that in stage t the pure automaton P_2 does not play the action $\omega_2^*(t)$. If $q(t)$ (resp. $q'(t)$) is not an accept-all state, then the automaton P_1 (resp. P_1') restarts at stage t, and the lemma follows. Thus, we assume from now on that both $q(t)$ and $q'(t)$ are accept-all states.

In which stages do both P_1 and P_1' visit an accept-all state? During the punishment phase none of these automata visits an accept-all state, and since $j_1 \neq j_1'$, during the implementation of the babbling phase they do not visit accept-all states at the same stage. Thus, only in the regular phase both automata visit accept-all states simultaneously, when implementing the action pairs (C, D). We will show that if P_2 deviates when P_1 implements either one of these action pairs, a punishment phase will ensue in at most $2k^2 + k + 1$ stages.

Suppose first that state $q(t)$ is the h_1'th state of the $(j+1)$'th C-block. Then $q'(t)$ is the h_1''th state of the $(j'+1)$'th C-block. Since P_2 deviates in stage t, it plays C instead of D, so that $q(t+1) = q(t) + 1$ and $q'(t+1) = q'(t) + 1$. The automaton P_1 expects now the sequence $(k - h_1) \times (C, C) + 1 \times (D, D)$ and is going to visit an accept-all state in $h_2 - h_1$ stages. Similarly, the automaton P_1' expects now the sequence $(k - h_1') \times (C, C) + 1 \times (D, D)$ and is going to visit an accept-all state in $h_2' - h_1'$ stages. By (A1)–(A2) we have $h_1 \neq h_1'$ and $h_2 - h_1 \neq h_2' - h_1'$, and therefore no sequence of actions that P_2 can generate is compatible with both automata, hence at least one of them will restart within at most k stages.

The argument is similar if state $q(t)$ is the h_2'th stage of the $(j+1)$'th C-block.

It is left to handle the case in which state $q(t)$ is the third stage of the $(j+1)$'th C-block, in which case state $q'(t)$ is the third stage of the $(j'+1)$'th C-block. The automaton P_1 expects the sequence

$$\omega := (k-3) \times (C, C) + k \times (D, D) + \sum_{n=j+2}^{k} (k \times (C, C) + k \times (D, D)) + (k+1) \times (C, C),$$

and visits two accept-all states in $h_1 - 3$ and $h_2 - 3$ stages. Similarly, the automaton P_1' expects the sequence

$$\omega' := (k-3) \times (C, C) + k \times (D, D) + \sum_{n=j'+2}^{k} (k \times (C, C) + k \times (D, D)) + (k+1) \times (C, C),$$

and visits two accept-all states in $h_1' - 3$ and $h_2' - 3$ stages. By (A1)–(A2) we have $h_1 \neq h_1'$, $h_2 \neq h_2'$, and $j \neq j'$, and therefore no sequence of actions that P_2 can generate is compatible with both automata, hence at least one of them will restart within at most $2k^2 + k + 1$ stages. □

Remark 10. Lemma 9 assumes that both automata start at state 1. The reader can verify that the proof is valid as soon as the two automata start at the same state; that is, it holds whenever $q(1) = q'(1)$.

B.3. An automaton P_2 for player 2 that is compatible with ω^*

As in Section B.1 we define a family of pure automaton for Player 2, which are compatible with ω^* and have size $k^3 + 2k^2 + k + 4$. As for player 1, the automata in the family depend on two parameters, an integer $j \in \{1, 2, \ldots, k-1\}$ and a set $H = \{h_1, h_2, h_3\}$ of three integers that satisfy $1 \leq h_1 < h_2 < h_3 \leq k$.

Let $Q = \{1, 2, \ldots, k^3 + 2k^2 + k + 4\}$ be the set of states of the automaton with $q^* = 1$ the initial state. The sets Q^P, Q^C, and Q^D of the states that implement the punishment phase, the C-blocks, and the D-blocks, respectively, and the output function f, are defined as in Section B.1. The transition function along the coordinated play is

$$g(q, f(q)) = q + 1, \quad 1 \leq q < k^3 + 2k^2 + k + 1.$$

We now add transitions that implement the next three action pairs in ω^*, which are $\omega_5 = (D, D) + 2 \times (C, C)$. To this end we use the last three states of Q.

The action function for these states is given by

$$f(k^3 + 2k^2 + k + 2) = D;\ f(k^3 + 2k^2 + k + 3) = C;\ f(k^3 + 2k^2 + k + 4) = C$$

and the transition is given by

$$g(q, f(q)) = q + 1, \quad k^3 + 2k^2 + k + 1 \leq q < k^3 + 2k^2 + k + 4.$$

The last part of the regular phase, $3 \times (C, D)$, is implemented by reusing states h_1, h_2, and h_3 in the j'th D-block:

$$g(k^3 + 2k^2 + k + 4, C) = k^3 + 2k(j-1) + k + h_1, \tag{4}$$

$$g(k^3 + 2k(j-1) + k + h_1, C) = k^3 + 2k(j-1) + k + h_2, \tag{5}$$

$$g(k^3 + 2k(j-1) + k + h_2, C) = k^3 + 2k(j-1) + k + h_3. \tag{6}$$

Finally, from state $k^3 + 2k(j-1) + k + h_3$ the regular phase should be repeated, so that we define

$$g(k^3 + 2k(j-1) + k + h_3, C) = k^3 + 2k^2 + k + 2.$$

All transitions that are not defined above lead to state 1, The automaton that we just constructed is denoted P_2^{j,h_1,h_2,h_3}. Its accept-all states are states $k^3 + 2k(j-1) + k + h_1$, $k^3 + 2k(j-1) + k + h_2$, and $k^3 + 2k(j-1) + k + h_3$.

B.4. A mixed automaton of player 2

The definition of the mixed strategy M_2 is analog to that of M_1. The pure automaton $P_2 = P_2(j, h_1, h_2, h_3)$ that was constructed in Section B.3 depends on four parameters j, h_1, h_2 and h_3. We will now define a mixed automaton $M_2 = M_2(k)$ that chooses these parameters randomly.

Let $\mathcal{H} = \{(j^d, h_1^d, h_2^d, h_3^d)\colon 1 \leq d < \frac{k}{6}\}$ be a collection of $\frac{k}{6}$ triplets that satisfy the following conditions:

B1 $(j^d)_{d=1}^{k/6}$ are distinct elements from $\{1, 2, \ldots, k-1\}$, and $(h_1^d, h_2^d, h_3^d)_{d=1}^{k/6}$ are distinct elements from $\{1, 2, \ldots, k\}$.

B2 For every distinct $d_1, d_2 \in \{1, 2, \ldots, \frac{k}{6}\}$ the six numbers $h_2^{d_1} - h_1^{d_1}$, $h_2^{d_2} - h_1^{d_2}$, $h_3^{d_1} - h_2^{d_1}$, $h_3^{d_2} - h_2^{d_1}$, $h_3^{d_1} - h_1^{d_1}$, and $h_3^{d_2} - h_1^{d_1}$ are distinct.

One can define, e.g., $j^d = d$, $h_1^d = d$, $h_2^d = 2d + \frac{k}{6}$, and $h_3^d = 3d + 3\frac{k}{6}$, for every $d \in \{1, 2, \ldots, \frac{k}{6}\}$.

The mixed automaton $M_2 = M_2(k)$ chooses uniformly one of the pure automata in $\mathcal{P}_2 := \{P_2^{j,h_1,h_2,h_3}, (j, h_1, h_2, h_3) \in \mathcal{H}\}$. As for Player 1, all pure automata in the support of M_2 are compatible with ω^* for Player 2, so that M_1 is compatible with ω^* for Player 1 as well. The analog of Lemma 9 is the following.

Lemma 11. *Let $P_2 = P_2^{j,h_1,h_2,h_3}$ and $P_2' = P_2^{j',h_1',h_2',h_3'}$ be two different pure automata in the support of M_2 and let P_1 be any pure automaton of Player 1. Let t be the first stage in which the play under (P_1, P_2) differs from ω^*. Then at least one of the automata P_2 and P_2' restarts before stage $t + 2k^2 + k + 1$.*

B.5. (M_1, M_2) is a c-BCC equilibrium

In this section we prove that (M_1, M_2) is a c-BCC equilibrium, provided the cost of memory c is neither too high nor too low. If c is very low, switching to a significantly larger automaton may not be too costly, while if c is very high, the cost of the automaton M_i, which is $c|M_i|$, is high, so that the players will profit by deviating to a small automaton, thereby saving the cost of the automaton. Here we will prove that if $\frac{12}{k^4} < c < \frac{\eta}{4k^3}$ then (M_1, M_2) is a c-BCC equilibrium, where $\eta < \min\{x_1^* - 1, x_2^* - 1\}$.

By construction, the average payoff under (M_1, M_2) is $\gamma(M_1, M_2) = x^* = (\frac{7}{6}, \frac{19}{6})$.

How can a player increase her payoff? To this end she needs to learn which states are the accept-all states of the other player's realized pure automaton. In our construction each state is reused by at most one pure automaton, and therefore learning the identity of the accept-all states essentially means enumerating over all their possible values. There are $O(k)$ different possibilities for accept-all states, and each failed attempt requires (at least) k^3 new states to pass the ensuing punishment phase. It follows that to successfully learn the accept-all states the deviator has to use a memory of size of the order of k^4. The relation between c and k ensures that such a deviation is not profitable. We now turn this intuition into a formal argument.

B.5.1. A lower bound on the size of player 2's automaton that can gain against M_1

In the present section we provide a lower bound on the size of an automaton P_2 of Player 2 that profits when facing M_1. As we will see, the size of such an automaton P_2 will be larger than $O(k^3)$, the complexity of ω^*.

Denote by $(P_1^d)_{d=1}^{k/4}$ the pure automata in the support of M_1. Suppose that the players use the automata (P_1^d, P_2). Denote by $q_2(t; P_1^d)$ the state of the automaton P_2 at stage t when it faces the automaton P_1^d.

If P_2 is not compatible with ω^* for Player 2, then P_1^d restarts whenever a deviation from ω^* is detected, and a punishment phase starts. Denote by t_n^d the stage at the n'th time in which P_1^d visits state 1 when facing P_2, that is, the stage in which the n'th punishment phase starts:

$$\begin{aligned} t_1^d &:= 1, \\ t_{n+1}^d &:= \min\{t > t_n^d : q_1(t) = 1\}, \quad n \geq 1. \end{aligned}$$

By convention, the minimum of an empty set is ∞.

There are two scenarios in which Player 2 may improve her long-run average payoff. One possibility is if there exists n such that $t_n^d < \infty = t_{n+1}^d$. Then t_n^d is the last stage in which the automaton P_1^d restarts. If the play after stage t_n^d is different from ω^*, it means that Player 2 plays as if she knows (some of) the parameters that determine P_1^d, and she might use this information to improve her payoff. Another possibility is that $(t_n^d)_{n\in\mathbb{N}}$ are finite and between two of these stages the average payoff of Player 2 is higher[7] than x_2^*.

This leads us to the following definition. For every $d \in \{1, 2, \ldots, \frac{k}{4}\}$ and every $n \in \mathbb{N}$ let ω_n^d be the play generated from stage n and on under (P_1^d, P_2).

[7] In fact, if $(t_n^d)_{n\in\mathbb{N}}$ are finite then, so that Player 2 improves her payoff, the average payoff between t_n^d and $t_{n+1}^d - 1$ should be higher than x^* infinitely often.

Definition 12. The automaton P_2 *fools* the automaton P_1^d if either one of the following conditions hold when the automata (P_1^d, P_2) face each other.

C1) There is $n_0 \in \mathbb{N}$ such that $t^d_{n_0} < \infty = t^d_{n_0+1}$ and $\omega^d_{n_0} \neq \omega^*$.

C2) $t^d_n < \infty$ for every $n \in \mathbb{N}$, and there is $n_0 \in \mathbb{N}$ such that the average payoff for Player 2 between stages $t^d_{n_0}$ and $t^d_{n_0+1} - 1$ is strictly higher[8] than x_2^*.

Since the punishment phase lowers the average payoff, provided that k is sufficiently large, if Condition C2 holds, then the play between stages $t^d_{n_0}$ and $t^d_{n_0+1} - 1$ is not a prefix of ω^*.

Neither C1 nor C2 imply that the long-run average payoff under (P_1^d, P_2) is higher than x_2^*. Yet, as the next lemma shows, the converse is true: if the long-run average payoff of Player 2 under (P_1^d, P_2) exceeds x_2^*, then P_2 must have fooled P_1^d.

Lemma 13. *If P_2 does not fool P_1^d then $\gamma_2(P_1^d, P_2) \leq x_2^*$.*

Proof. Since both P_1^d and P_2 are automata, $\gamma_2(P_1^d, P_2)$, which is the long-run average payoff of Player 2 under (P_1^d, P_2), exists. Suppose first that $t^d_n < \infty$ for every $n \in \mathbb{N}$. Since P_2 does not fool P_1^d, for every $n \in \mathbb{N}$ the average payoff of Player 2 between stages t^d_n and $t^d_{n+1} - 1$ is at most x_2^*, and therefore $\gamma_2(P_1^d, P_2) \leq x_2^*$.

Suppose now that there is $n_0 \in \mathbb{N}$ such that $t^d_{n_0} < \infty = t^d_{n_0+1}$. Since P_2 does not fool P_1^d, we have $\omega^d_{n_0} = \omega^*$, so that $\gamma_2(P_1^d, P_2) = x_2^*$, and the result follows. □

The following proposition states that to be able to fool L_0 pure automata in the support of M_1, Player 2 must use an automaton of size at least $L_0 k^3$. To profit Player 2 needs to fool many of the pure automata in $\mathcal{P}_1$, hence this result will induce a lower bound on the size of an automaton of Player 2 that profits by deviating.

Proposition 14. *Denote by L_0 the number of pure automata P_1^d, $1 \leq d \leq \frac{k}{4}$, that P_2 fools. Then $|P_2| \geq L_0 k^3$.*

Proof. If condition C1 in Definition 12 holds, we say that P_2 *fools* P_1^d in stages $\{t^d_{n_0}, t^d_{n_0} + 1, \ldots\}$. If condition C2 holds, we say that P_2 *fools* P_1^d in stages $\{t^d_{n_0}, t^d_{n_0} + 1, \ldots, t^d_{n_0+1} - 1\}$. In both cases[9] we set $t^d_* = t^d_{n_0}$, and we say that at stage t^d_* Player 2 *starts to fool* P_1^d. Denote by $R_d = \{q_2(t^d_*; P_1^d), q_2(t^d_* + 1; P_1^d), \cdots, q_2(t^d_* + k^3 - 1; P_1^d)\}$ the k^3 states that P_2 visits at the beginning of the period in which it fools P_1^d. During these stages the automaton P_1^d executes the punishment phase, and the payoff of Player 2 is low.

The following lemma implies Proposition 14.

Lemma 15. *Let $1 \leq d_1 < d_2 \leq \frac{k}{4}$. If P_2 fools both $P_1^{d_1}$ and $P_1^{d_2}$, then $|R_{d_1}| = |R_{d_2}| = k^3$ and $R_{d_1} \cap R_{d_2} = \emptyset$.*

[8] Observe that in this case $t^d_{n_0+1} \geq t^d_{n_0} + k^3$. In fact, a stronger bound can be obtained.

[9] If condition C2 holds, there may be several stages n_0 at which P_2 starts to fool P_1^d. In such a case we choose one of them arbitrarily.

Proof. The first k^3+1 action pairs of ω^* are coordinated, and the (k^3+1)'th action of Player 2 differs from her actions in the first k^3 stages. Lemma 5(3) implies the following:

Fact 1. If P_2 fools P_1^d then the states in R_d are distinct: $|R_d| = k^3$.

Lemma 5(3) also implies the following:

Fact 2. If R_{d_1} and R_{d_2} are not disjoint, then the last state in R_{d_1} coincides with the last state in R_{d_2}, that is, $q_2(t_*^{d_1}+k^3-1;P_1^{d_1}) = q_2(t_*^{d_2}+k^3-1;P_1^{d_2})$.

Indeed, suppose that R_{d_1} and R_{d_2} are not disjoint, and assume that $q_2(t_*^{d_1}+n_1;P_1^{d_1}) = q_2(t_*^{d_2}+n_2;P_1^{d_2})$. We argue that necessarily $n_1=n_2$. This will imply that the last state in R_{d_1} coincides with the last state in R_{d_2}. Assume then to the contrary that, w.l.o.g., $n_1<n_2$. Lemma 5(3) implies that $q_2(t_*^{d_1}+n_1+s;P_1^{d_1}) = q_2(t_*^{d_2}+n_2+s;P_1^{d_2})$ for every s that satisfies $1\le s\le k^3-n_2+1$. Since P_2 fools $P_1^{d_1}$, the action that P_2 plays in state $q_2(t_*^{d_1}+n_1+k^3-n_2+1;P_1^{d_1})$ is D. Since P_2 fools $P_1^{d_2}$, the action that P_2 plays in state $q_2(t_*^{d_2}+n_2+k^3-n_2+1;P_1^{d_2})$ is C. But $q_2(t_*^{d_1}+n_1+k^3-n_2+1;P_1^{d_1}) = q_2(t_*^{d_2}+n_2+k^3-n_2+1;P_1^{d_2})$, a contradiction.

We are now ready to prove that $R_{d_1}\cap R_{d_2}=\emptyset$.

Assume to the contrary that R_{d_1} and R_{d_2} are not disjoint. By Fact 2, the last state in R_{d_1} coincides with the last state in R_{d_2}, that is, $q_2(t_*^{d_1}+k^3-1;P_1^{d_1}) = q_2(t_*^{d_2}+k^3-1;P_1^{d_2})$. Because, for $i=1,2$, at stage $t_*^{d_i}$ the automaton P_2 starts fooling $P_1^{d_i}$, it follows that the play under $(P_1^{d_i},P_2)$ after this stage is different from ω^*. Denote by $t_*^{d_i}+t$ the first stage in which the play under $(P_1^{d_i},P_2)$ differs from ω^*. Lemma 9 implies that at least one of the automata $(P_1^{d_i})_{i=1,2}$, say the automaton $P_1^{d_1}$, restarts before stage $t_*^{d_i}+t+2k^2+k+1$ (see Remark 10). We argue that P_2 does not fool $P_1^{d_1}$, a contradiction.

Indeed, the play from stage $t_*^{d_1}$ until the automaton $P_1^{d_1}$ restarts consists of

- k^3 stages of the punishment phase, in which Player 2's payoff is 1 per stage;
- $2k^2+k+1$ stages of the babbling phase in which her payoff is at most 4 in each stage[10];
- several rounds, say r, of the regular phase, in which her average payoff is x_2 per round;
- if deviation occurs in a regular phase, at most 6 stages in a portion of the regular phase, in which the per-period payoff is at most 4;
- and at most k^2+k+1 stages between the deviation and the stage in which $P_1^{d_1}$ restarts, in which her payoff is at most 4 in each stage.

Thus, Player 2's average payoff between stage $t_*^{d_1}$ and the stage in which $P_1^{d_1}$ restarts is at most

$$\frac{k^3\times 1+(3k^2+2k+8)\times 4+6r\times x_2}{k^3+3k^2+2k+8+6r}, \tag{7}$$

which is strictly lower than x_2 provided

[10] Or at most $2k^2+k+1$ stages, if deviation occurs during the babbling phase.

$$x_2 > \frac{k^3+12k^2+8k+32}{k^3+3k^2+2k+8} > 1+\frac{6}{k},$$

as we claimed. □

The analog of Proposition 14 for Player 1 is the following.

Theorem 16. *Let P_1 be a pure automaton of Player 1, and denote by L_0 the number of pure automata P_2^d that P_1 fools. Then $|P_1| \geq L_0 k^3$.*

B.5.2. A BCC-equilibrium

In this section we argue that the pair of automata (M_1, M_2), which was constructed in Sections B.2 and B.4, is a c-BCC equilibrium, provided k is sufficiently large and $\frac{12}{k^4} < c < \frac{\eta}{4k^3}$. We only prove the claims for Player 2. The claims for Player 1 can be proven analogously. Below we denote the state of an automaton of player i at stage t by $q_i(t)$.

We now prove that Player 2 cannot profit by deviating to an automaton smaller than M_2.

Lemma 17. *Assume that k and c satisfy $\frac{24}{k} < \frac{\eta}{2}$ and $c < \frac{\eta}{2k^3}$. Let P_2' be an automaton for Player 2 with size smaller than k^3+2k^2+k+4. Then $\gamma_2(M_1, P_2') - c|P_2'| \leq \gamma_2(M_1, M_2) - c|M_2|$.*

Proof. Because the complexity of ω^* w.r.t. Player 2 is k^3+2k^2+k+4, the play under (P_1^l, P_2') is not ω^*. By Lemma 15, and because the size of P_2 is smaller than $2k^3$, the automaton P_2' can fool at most one of the automata $(P_1^d)_{d=1}^{k/4}$. Because it cannot generate ω^*, any automaton that P_2 does not fool restarts after at most k^3+2k^2+k+3 stages, so that the average payoff is at most $\frac{k^3}{k^3+2k^2+k+3} + 4\frac{2k^2+k}{k^3+2k^2+k+3}$. It follows that the expected payoff $\gamma_2(M_1, P_2')$ is at most

$$4\frac{1}{k/4} + \frac{\frac{k}{4}-1}{\frac{k}{4}}\left(\frac{k^3}{k^3+2k^2+k+3} + 4\frac{2k^2+k}{k^3+2k^2+k+3}\right) \leq 1+\frac{24}{k} < 1+\frac{\eta}{2}.$$

Because the size of the automaton M_2 is k^3+2k^2+k+4, the gain of reducing the size of automaton from $|M_2|$ to $|P_2'|$ is at most $c(k^3+2k^2+k+3)$. Player 2 does not profit by this deviation as soon as

$$x_2^* \geq 1+\frac{24}{k} + c(k^3+2k^2+k+3),$$

and therefore it is enough to require that

$$x_2^* - 1 > \eta > \frac{24}{k} + c(k^3+2k^2+k+3).$$

The right-hand side inequality holds, provided

$$c < \frac{\eta - \frac{24}{k}}{k^3+2k^2+k+3},$$

so it is enough to require that $c < \frac{\eta}{4k^3}$. □

We finally prove that Player 2 cannot profit by deviating to an automaton larger than M_2.

Lemma 18. *Let P_2' be a pure automaton such that $\gamma_2(M_1, P_2') > x_2$. Then $\gamma_2(M_1, P_2') - c|P_2'| \leq \gamma_2(M_1, M_2) - c|M_2|$, provided $c > \frac{12}{k^4}$.*

Proof. Let L_0 be the number of pure automata $(P_1^d)_{d=1}^{k/4}$ that P_2 fools. Because $\gamma_2(M_1, P_2') > x_2^*$ we have $L_0 \geq 1$. If P_2 fools the realized pure automaton of Player 1, then Player 2's long-run average payoff is at most 4, the maximal payoff in the game. If P_2 does not fool the realized pure automaton of Player 1, then Player 1's long-run average payoff is at most x_2^*. The expected long-run average payoff of Player 2 then satisfies

$$\gamma_2(M_1, P_2') \leq 4\frac{L_0}{\frac{k}{4}} + x_2^*\frac{\frac{k}{4} - L_0}{\frac{k}{4}} < x_2^* + 12\frac{L_0}{k}.$$

By Theorem 16 we have $|P_2'| \geq L_0 k^3$, and therefore

$$\gamma_2(M_1, P_2') < x_2^* + 12\frac{L_0}{k} = x_2^* + 12\frac{L_0 k^3}{k^4} \leq x_2^* + |P_2'| \times \frac{12}{k^4}.$$

Therefore, as soon as $c > \frac{12}{k^4}$ Player 2 does not profit by this deviation. □

To summarize, given the feasible and individually rational payoff vector x^*, we choose $\eta \in (0, \min\{x_1^* - 1, x_2^* - 1\})$. For every $c > 0$ we define $k = k(c)$ by the equality $c = \frac{\eta}{k^{3.5}}$. Then $\frac{12}{k^4} < c < \frac{\eta}{3k^3}$, provided c is small enough (so that $k(c)$ is large enough). The pair of automata $(M_1(k(c)), M_2(k(c)))$ are then c-BCC equilibrium with payoff x^*.

Appendix C. Supplementary material

Supplementary material related to this article can be found online at http://dx.doi.org/10.1016/j.jet.2016.02.007.

References

Abreu, D., Rubinstein, A., 1988. The structure of Nash equilibrium in repeated games with finite automata. Econometrica 56, 1259–1281.

Banks, J., Sundaram, R., 1990. Repeated games, finite automata and complexity. Games Econ. Behav. 2, 97–117.

Ben Porath, E., 1993. Repeated games with finite automata. J. Econ. Theory 59, 17–32.

Hernández, P., Solan, E., 2016. Bounded computational capacity equilibrium – online appendix. Available at www.math.tau.ac.il/~eilons.

Kalai, E., 1990. Bounded rationality and strategic complexity in repeated games. In: Ichiishi, Neyman, Tauman (Eds.), Game Theory and Applications. Academic Press, San Diego, pp. 131–157.

Neyman, A., 1985. Bounded complexity justifies cooperation in the finitely-repeated prisoners' dilemma. Econ. Lett. 19, 227–229.

Neyman, A., 1997. Cooperation, repetition and automata. In: Hart, S., Mas-Colell, A. (Eds.), Cooperation: Game-Theoretic Approaches. In: NATO ASI Series F, vol. 155. Springer-Verlag, pp. 233–255.

Neyman, A., 1998. Finitely repeated games with finite automata. Math. Oper. Res. 23, 513–552.

Neyman, A., Okada, D., 1999. Strategic entropy and complexity in repeated games. Games Econ. Behav. 29, 191–223.

Neyman, A., Okada, D., 2000a. Repeated games with bounded entropy. Games Econ. Behav. 30, 228–247.

Neyman, A., Okada, D., 2000b. Two-person repeated games with finite automata. Int. J. Game Theory 29, 309–325.

Piccione, M., 1992. Finite automata equilibria with discounting. J. Econ. Theory 56, 180–193.

Piccione, M., Rubinstein, A., 1993. Finite automata play a repeated extensive game. J. Econ. Theory 61, 160–168.

Rubinstein, A., 1986. Finite automata play the repeated prisoner's dilemma. J. Econ. Theory 39, 83–96.

Zemel, E., 1989. Small talk and cooperation: a note on bounded rationality. J. Econ. Theory 49, 1–9.