Proceedings of the Twenty-Third International Joint Conference on Artificial Intelligence

Iterated Boolean Games

Julian Gutierrez and **Paul Harrenstein** and **Michael Wooldridge**
Department of Computer Science
University of Oxford

Abstract

Iterated games are well-known in the game theory literature. We study *iterated Boolean games*. These are games in which players repeatedly choose truth values for Boolean variables they have control over. Our model of iterated Boolean games assumes that players have goals given by formulae of Linear Temporal Logic (LTL), a formalism for expressing properties of state sequences. In order to model the strategies that players use in such games, we use a finite state machine model. After introducing and formally defining iterated Boolean games, we investigate the computational complexity of their associated game-theoretic decision problems as well as semantic conditions characterising classes of LTL properties that are preserved by pure strategy Nash equilibria whenever they exist.

1 Introduction

Playing a game more than once against the same opponent can have a dramatic effect on which outcomes of the game can be sustained as equilibria [Osborne and Rubinstein, 1994, pp.133–161]. To take a classic example in the literature, in the one-shot Prisoner's Dilemma there is a unique pure Nash equilibrium in which both players defect, leading to payoffs that are worse for both players than the payoffs they would have obtained had they cooperated. However, cooperation cannot be ensured in the one-shot Prisoner's Dilemma. If, instead, the same players repeatedly meet each other then cooperation can be sustained—leading to equilibria (outcomes) which can be better than those of the one-shot version of the game. Cooperation is rationally sustainable because the players will meet in the future, and will thereby have the opportunity to *punish* each other for non-cooperation.

We study iterated versions of *Boolean games*. The basic idea of a Boolean game [Harrenstein *et al.*, 2001] is that each player i is associated with a goal, represented as a logical formula γ_i, and player i's main purpose is to ensure that γ_i is satisfied. The strategies and choices for each player i are defined with respect to a set of Boolean variables Φ_i, drawn from an overall set of variables Φ. Player i is assumed to have unique control over variables Φ_i, in that it can assign truth values to these variables in any way it chooses. Strategic concerns arise in Boolean games as the satisfaction of player i's goal γ_i can depend on the variables controlled by other players.

In the version of Boolean games that we study, it is assumed that players interact over an *infinite* series of rounds, where at each round each player makes an assignment to the variables under its control. Goals are expressed as formulae of *Linear Temporal Logic* (LTL), a well-known formalism for expressing properties of distributed and concurrent systems [Emerson, 1990; Manna and Pnueli, 1992; 1995]. Formulae of LTL are essentially predicates over infinite sequences of states. Thus, whether a player's goal is or is not satisfied may depend not just on how players act on one round, but how they act in all future rounds.

Players in Boolean games can be understood as nondeterministic computer programs, and the model thus has great relevance to multi-agent systems research. As players can model programs, building strategies in Boolean games corresponds to synthesising computer systems from their logical specifications, for instance as the modules for control and synchronisation of concurrent processes [Kupferman *et al.*, 2000; Vardi, 2008; Pnueli and Rosner, 1988].

The paper contains three main contributions. Firstly, it formalises iterated Boolean games. Moreover, it provides a finite state machine representation (an *operational* model) along with a temporal logic theory (a *denotational* model) for the strategies and players. This dual operational/denotational model for iterated Boolean games allows us to investigate computational properties regarding the games, such as whether a collection of strategies forms a Nash equilibrium.

Secondly, we study the complexity of various decision problems. Model checking for iterated Boolean games is in PSPACE and thus no harder than model checking for LTL [Sistla and Clarke, 1985]. Synthesis is 2EXPTIME-complete, matching the complexity of synthesis for 'rational systems' [Fisman *et al.*, 2010]. Moreover, checking whether a strategy profile is a pure Nash equilibrium and whether a game has at least one pure Nash equilibrium are complete problems for PSPACE and 2EXPTIME, respectively.

Thirdly, we give *Folk Theorems* for iterated Boolean games. These theorems provide semantic characterisations of LTL properties that are satisfied in equilibrium outcomes. Some of these Folk Theorems completely characterise games for which some questions can be answered more efficiently; in particular, synthesis can be done in PSPACE.

Related work Equilibria in Boolean games have been studied before (*e.g.*, [Dunne *et al.*, 2008; Bonzon *et al.*, 2009; Grant *et al.*, 2011]) but, to our knowledge, not in models where the players are assumed to interact for an infinite number of rounds, which is the setting our results pertain to.

However, there are many games in the computer science literature where the plays have infinite length; see, *e.g.*, [Chatterjee and Henzinger, 2012; Grädel and Ummels, 2008] for surveys on the topic. These games are most usually played by two players (together with a hostile environment) who display independent, distributed, and concurrent behaviour. In our setting, we are interested in situations where there are multiple players who do not necessarily have opposing objectives: in fact cooperation is a desirable behaviour in our setting.

Folk Theorems for other games are mostly found in the game theory literature rather than in computer science or multi-agent systems research. Yet there are a few exceptions. In [Fisman *et al.*, 2010] a similar question is asked for a problem—rational synthesis—that is essentially one of the decision problems for iterated Boolean games. Other results on synthesis can be found too, see, *e.g.,* [Chatterjee *et al.*, 2008; Ummels, 2006; Lustig and Vardi, 2009; Kupferman *et al.*, 2000]. In most of this literature the emphasis is on the question of whether a particular LTL formula can be synthesised. With the Folk Theorems, on the other hand, the focus is on semantic representations of complete classes of properties (LTL properties in this case) which can be rationally sustained—and hence possibly synthesised.

2 Preliminaries

Let $\Phi = \{p, q, \ldots\}$ be a (finite, fixed, non-empty) set of Boolean variables and L_0 the set of formulae of classical propositional logic constructed over Φ. A *valuation* v is a subset of variables, that is, $v \subseteq \Phi$, where it is understood that v assigns truth to all the variables it contains and falsity to the others. Let V be the set of valuations for Φ. For formulae $\varphi \in L_0$ we use $v \models \varphi$ to indicate that v satisfies φ.

Linear Temporal Logic (LTL): LTL extends classical propositional logic with modal tense operators $\mathbf{X}$ ("next"), $\mathbf{F}$ ("eventually"), $\mathbf{G}$ ("always"), and $\mathbf{U}$ ("until"), which can be used for expressing properties of sequences. We take $\mathbf{X}$ and $\mathbf{U}$ as our atomic operators, and define the remaining LTL connectives based on these. The syntax of LTL is defined with respect to a set Φ of Boolean variables as follows:

$$\varphi ::= \top \mid p \mid \neg\varphi \mid \varphi \vee \varphi \mid \mathbf{X}\,\varphi \mid \varphi\,\mathbf{U}\,\varphi$$

where $p \in \Phi$. The remaining classical and LTL connectives are then defined in the standard way; in particular, we have $\mathbf{F}\,\varphi = \top\,\mathbf{U}\,\varphi$, and $\mathbf{G}\,\varphi = \neg\,\mathbf{F}\,\neg\varphi$. Given a set of variables Ψ, let $L(\Psi)$ be the set of LTL formulae over Ψ.

A core concept in the semantics for LTL is that of a *run*. A run $\rho\colon \mathbb{N} \to V$ is a function that assigns a valuation $\rho[t]$ to every time point $t \in \mathbb{N}$, indicating which Boolean variables are true at time t.[1] We interpret formulae of LTL with respect to pairs (ρ, t), where ρ is a run and $t \in \mathbb{N}$ is a temporal index into ρ. Formally, the semantics of LTL formulae is as follows:

$$\begin{array}{lll}
(\rho,t) \models \top & & \\
(\rho,t) \models p & \text{iff} & p \in \rho[t] \\
(\rho,t) \models \neg\varphi & \text{iff} & \text{it is not the case that } (\rho,t) \models \varphi \\
(\rho,t) \models \varphi \vee \psi & \text{iff} & (\rho,t) \models \varphi \text{ or } (\rho,t) \models \psi \\
(\rho,t) \models \mathbf{X}\,\varphi & \text{iff} & (\rho,t+1) \models \varphi \\
(\rho,t) \models \varphi\,\mathbf{U}\,\psi & \text{iff} & \text{for some } t' \geq t\colon\ ((\rho,t') \models \psi \text{ and} \\
 & & \qquad \text{for all } t \leq t'' < t'\colon (\rho,t'') \models \varphi).
\end{array}$$

If $(\rho, 0) \models \varphi$, we also write $\rho \models \varphi$ and say that ρ *satisfies* φ. We say that φ and ψ are *equivalent* if for all runs ρ we have $\rho \models \varphi$ if and only if $\rho \models \psi$. An LTL formula $\varphi \in L$ is *satisfiable* if there is some run satisfying φ.

3 The game model

Variables, Control, and Choices: Each player i controls a (possibly empty) subset Φ_i of the overall set of Boolean variables. That is, player i has the unique ability to choose the value (either true or false) of each variable $p \in \Phi_i$ at each round of the game. A *choice* for agent $i \in N$ is a subset v_i of the propositional variables under his control, *i.e.,* $v_i \subseteq \Phi_i$. At every round of the game, each player makes such a choice. Let V_i be the set of choices for agent i. A *choice vector* $\vec{v}$ is a collection of choices, one for each player, that is, $\vec{v} = (v_1, \ldots, v_n)$. A choice vector uniquely defines an overall valuation for Φ, and vice versa. We will generally abuse notation by treating choice vectors as valuations and valuations as choice vectors. Every choice v_i and every choice vector $\vec{v}$ is associated with a characterising formula $\chi_{v_i}^{\Phi_i} = \bigwedge_{p \in \Phi_i \cap v_i} p \wedge \bigwedge_{p \in \Phi_i \setminus v} \neg p$ and $\chi_{\vec{v}}^{\Phi} = \bigwedge_{i \in N} \chi_{v_i}^{\Phi}$.Thus, $\vec{v} \models \chi_{\vec{w}}^{\Phi}$ if and only if $\vec{v} \cap \Phi = \vec{w} \cap \Phi$. We will omit the superscripts whenever Φ and Φ_i are clear from the context.

Iterated Boolean Games: An *Iterated Boolean Game* (hereafter sometimes just called a "game") is a structure:

$$G = (N, \Phi, \Phi_1, \ldots, \Phi_n, \gamma_1, \ldots, \gamma_n)$$

where $N = \{1, \ldots, n\}$ is a set of players, $\Phi = \{p, q, \ldots\}$ is a finite set of Boolean variables[2], $\Phi_i \subseteq \Phi$ is the set of Boolean variables under the unique control of player i, and $\gamma_i \in L$ is the LTL goal of player i.

There is one requirement on these structures: the sets of Boolean variables $\Phi_1, \ldots, \Phi_n$ must form a partition of Φ, that is, $\Phi_i \cap \Phi_j = \emptyset$ for all $i \neq j \in N$ and $\Phi = \Phi_1 \cup \cdots \cup \Phi_n$.

The *outcomes* of a game are given by the runs $\rho \in R$. Each player i is associated with a strict preference relation $\succ_i$ over the runs. Every relation $\succ_i$ is binary in the sense that player i strictly prefers runs that satisfy its goal γ_i over runs that do not and is indifferent otherwise, that is, for all runs ρ and ρ',

$$\rho \succ_i \rho' \text{ if and only if both } \rho \models \gamma_i \text{ and } \rho' \not\models \gamma_i.$$

Moreover, $\rho \succsim_i \rho'$ means that it is not the case that $\rho' \succ_i \rho$.

[1] We use square brackets for parameters referring to time points.

[2] Arbitrary finite domains can be represented by Boolean domains—*viz.*, as finite numbers are represented by binary ones.

Strategies: In a one-shot Boolean game, a strategy for a player i is simply a single choice $v_i \in V_i$. This is, of course, not the case for iterated Boolean games. In iterated Boolean games, a strategy for player i is a function that makes a choice in each round of the game on the basis of the history of the game to date. Formally, we can understand such strategies as functions $f_i : V^* \to V_i$, where V^* denotes the set of finite sequences over V. In order to study computational problems for games, it is desirable to have a *finite* representation for strategies: here we model strategies as *deterministic finite state machines*. Such representations are widely used to study repeated games in the game theory literature [Osborne and Rubinstein, 1994, p.140–143] and are of course very natural from a computational point of view. Formally, a *machine strategy* σ_i for player i in a game $G = (N, \Phi, \Phi_1, \ldots, \Phi_n, \gamma_1, \ldots, \gamma_n)$ is given by a structure:

$$\sigma_i = (Q_i, q_i^0, \delta_i, \tau_i)$$

where Q_i is a finite, non-empty set of *states*, q_i^0 is the *initial* state, $\delta_i : Q_i \times V \to Q_i$ is a *state transition function*, and $\tau_i : Q_i \to V_i$ is a *choice function*. Let Σ_i denote the class of strategies for player i. It is implicit above that δ_i is a total function, and so, for any given state $q \in Q_i$ we must have $\delta_i(q, v)$ defined for all $2^{|\Phi|}$ valuations $v \in V$, resulting in machine strategies of size exponential in $|\Phi|$.

To obtain a more compact representation, we may allow the transition function δ_i to be a *partial* function. Such a partial strategy $\sigma_i = (Q_i, q_i^0, \delta_i, \tau_i)$ then represents the (complete) strategy $\sigma' = (Q_i \cup \{q_i^d\}, q_i^0, \delta', \tau_i')$, where q_i^d is an additional "default state" to which δ' in state q and valuation $\vec{v}$ maps, in case $\delta(q, v)$ is undefined. Moreover, $\tau'(q_i^d) = \emptyset$. This will relieve us of the need to explicitly define $\delta_i(q, v)$ for every v: we only need to define $\delta_i(q, v)$ for the valuations v of interest.

A *strategy profile* $\vec{\sigma}$ is an n-tuple of strategies, one for each player i, that is, $\vec{\sigma} = (\sigma_1, \ldots, \sigma_n)$. For $S \subseteq N$, define $(\vec{\sigma}_{-S}, \vec{\sigma}_S')$ as the strategy profile $\vec{\sigma}''$ such that $\sigma_i'' = \sigma_i$ if $i \notin S$ and $\sigma_i'' = \sigma_i'$ otherwise. If $S = \{i\}$ we omit the curly braces and write $(\vec{\sigma}_{-i}, \sigma_i')$ rather than $(\vec{\sigma}_{-\{i\}}, \sigma_{\{i\}}')$.

Because strategies are deterministic, each $\vec{\sigma}$ induces a unique run, denoted by $\rho(\vec{\sigma})$. To define $\rho(\vec{\sigma})$, we need some notation. First, a *state vector* of a strategy profile is a tuple $\vec{q} = (q_1, \ldots, q_n)$, where, for all $i \in N$, we have $q_i \in Q_i$. With each point of time t we associate both a state vector denoted by $\vec{q}[t] = (q_1[t], \ldots, q_n[t])$ and an outcome denoted by $\vec{v}[t] = (v_1[t], \ldots, v_n[t])$. The *history* $h(\vec{\sigma})$ of $\vec{\sigma}$ is an infinite sequence of interleaved state vectors and outcomes:

$$h(\vec{\sigma}) = \vec{q}[0] \xrightarrow{\vec{v}[0]} \vec{q}[1] \xrightarrow{\vec{v}[1]} \cdots$$

where:

$$\vec{q}[0] = (q_1^0, \ldots, q_n^0)$$
$$\vec{v}[0] = (\tau_1(q_1^0), \ldots, \tau_n(q_n^0)),$$

and for all $t \in \mathbb{N}$ such that $t > 0$ we have:

$$\vec{q}[t] = (\delta_1(q_1[t-1], \vec{v}[t-1]), \ldots, \delta_n(q_n[t-1], \vec{v}[t-1]))$$
$$\vec{v}[t] = (\tau_1(q_1[t]), \ldots, \tau_n(q_n[t]))$$

Note that machine strategies provide an *operational* model for the behaviour/interactions of the players in a game. The following lemma will prove very useful and is a consequence of standard expressivity results (see, e.g. [Thomas, 1990; Diekert and Gastin, 2008]) for LTL with respect to languages accepted by deterministic ω-automata.

Lemma 1 *For every satisfiable LTL formula φ there is a strategy profile $\vec{\sigma} = (\sigma_1, \ldots, \sigma_n)$ such that $\rho(\vec{\sigma}) \models \varphi$.*

As LTL can express every finite star-free ω-regular language and a run $\rho(\vec{\sigma})$ for a strategy profile $\vec{\sigma}$ can be seen as an ω-regular word over the valuations V over Φ, it can readily be appreciated that for every strategy profile $\vec{\sigma}$ there is an LTL formula $\varphi(\vec{\sigma})$ such that for all runs ρ,

$$\rho \models \varphi(\vec{\sigma}) \text{ if and only if } \rho = \rho(\vec{\sigma}).$$

Then, the LTL formula $\varphi(\vec{\sigma})$ exactly characterises the unique run $\rho(\vec{\sigma})$ induced by the strategy profile $\vec{\sigma}$ (the formula $\varphi(\vec{\sigma})$ holds on $\rho(\vec{\sigma})$ and only on $\rho(\vec{\sigma})$). The formula $\varphi(\vec{\sigma})$, however, may be exponential in the size of $\vec{\sigma}$. We thus also have a lemma which ensures that every run $\rho(\vec{\sigma})$ is characterised by an LTL formula that is polynomial in the size of $\vec{\sigma}$.

For each player i and each state $q_i \in Q_i$ introduce a new ("fresh") Boolean variable which, assuming $\Phi, Q_1, \ldots, Q_n$ to be pairwise disjoint, we will also denote by q_i. Moreover, we use $Q = Q_1 \cup \cdots \cup Q_n$ to refer to the entire set of these new variables. For every player i and every (complete) strategy σ_i we define an LTL formula $th(\sigma_i) \in L(\Phi \cup Q)$ as follows:

$$th(\sigma_i) = \mathit{INIT}(\sigma_i) \wedge \mathit{TRANS}(\sigma_i) \wedge \mathit{INVAR}(\sigma_i) \wedge \mathit{VAL}(\sigma_i)$$

where,

$$\mathit{INIT}(\sigma_i) = q_i^0,$$
$$\mathit{TRANS}(\sigma_i) = \mathbf{G} \bigwedge_{\delta_i(q_i, \vec{v}) = q_i'} \left((\chi_{\vec{v}}^{\Phi} \wedge q_i) \to \mathbf{X}\, q_i' \right),$$
$$\mathit{INVAR}(\sigma_i) = \mathbf{G} \bigvee_{q_i \in Q_i} \left(q_i \wedge \bigwedge_{q_i' \neq q_i} \neg q_i' \right),$$
$$\mathit{VAL}(\sigma_i) = \mathbf{G} \bigwedge_{q_i \in Q_i} \left(q_i \to \chi_{\tau(q_i)}^{\Phi} \right).$$

Let furthermore $TH(\vec{\sigma}) = \bigwedge_{i \in N} th(\sigma_i)$ for strategy profiles $\vec{\sigma} = (\sigma_1, \ldots, \sigma_n)$. Observe that $th(\sigma_i)$ is polynomial in the size of σ_i; for partial strategies a similar formula can be defined, still polynomial in size. Intuitively, $th(\sigma_i)$ is the LTL theory of strategy σ_i: the property $\mathit{INIT}(\sigma_i)$ encodes the initial state q_i^0, $\mathit{TRANS}(\sigma_i)$ encodes the transition relation δ_i, $\mathit{INVAR}(\sigma_i)$ ensures that the strategy is in exactly one state at any given time, while $\mathit{VAL}(\sigma_i)$ encodes the choice function τ_i. Thus, $th(\sigma_i)$ provides a *denotational* model for σ_i with respect to the semantics of LTL. In particular, for all $\rho \colon \mathbb{N} \to 2^{\Phi \cup Q}$ satisfying $TH(\vec{\sigma})$ it holds that $\rho[t] \cap \Phi = \rho(\vec{\sigma})[t]$ for all $t \in \mathbb{N}$, *i.e.*, the run ρ restricted to Φ coincides with $\rho(\vec{\sigma})$.

Formulae of the form $th(\sigma_i)$—which characterise the players' behaviour—are satisfiable, as shown next.

Lemma 2 *Let $\vec{\sigma} = (\sigma_1, \ldots, \sigma_i)$ be a strategy profile and let $S \subseteq N$. Then, $\bigwedge_{i \in S} th(\sigma_i)$ is satisfiable.*

Based on the same construction, the following lemma holds.

Lemma 3 *Let $\vec{\sigma}$ be a strategy profile in the game*

$$G = (N, \Phi, \Phi_1, \ldots, \Phi_n, \gamma_1, \ldots, \gamma_n).$$

Then, for all $S \subseteq N$ and all $\varphi \in L(\Phi)$, we have that

$\bigwedge_{i \in S} th(\sigma_i) \wedge \varphi$ *is satisfiable implies* $\rho(\vec{\sigma}'_{-S}, \vec{\sigma}_S) \models \varphi$ *for some* $\vec{\sigma}'$.

Nash Equilibrium: Let us now define the well-known notion of (pure strategy) Nash equilibrium for iterated Boolean games, which will allow us—in the next section—to ask different computational questions about these games. A strategy profile $\vec{\sigma}$ is defined to be a *(pure strategy) Nash equilibrium* if for all players $i \in N$ and for all strategies $\sigma'_i \in \Sigma_i$ we have

$$\rho(\vec{\sigma}) \succsim_i \rho(\vec{\sigma}_{-i}, \sigma'_i).$$

Let $NE(G)$ denote the set of Nash equilibria of game G.

Due to the binary character of the players' preferences, our definition implies that the strategy profile $\vec{\sigma}$ is a Nash equilibrium if every player whose goal is not satisfied by the run $\rho(\vec{\sigma})$ cannot unilaterally deviate and get its goal achieved.

Observation 4 *Strategy profile $\vec{\sigma}$ is a (pure strategy) Nash equilibrium if and only if for every player $i \in N$,*

$$\textit{either } \rho(\vec{\sigma}) \models \gamma_i \textit{ or for all } \sigma'_i \in \Sigma_i\textit{: } \rho(\vec{\sigma}_{-i}, \sigma'_i) \not\models \gamma_i.$$

4 Complexity

We now consider the computational complexity of problems relating to Nash equilibria in iterated Boolean games. We first establish an upper bound on the complexity of a problem that underpins many of others that we will study later on:

Given: Game G, profile $\vec{\sigma}$, and LTL formula φ.
MODEL CHECKING: Is it the case that $\rho(\vec{\sigma}) \models \varphi$?

This problem is not quite the same as the "standard" LTL model checking problem, which is known to be PSPACE-complete ("truth in an R-structure" [Sistla and Clarke, 1985, p.741]), and so we need to establish the upper bound.

Proposition 5 MODEL CHECKING *is in PSPACE.*

Proof sketch: We reduce MODEL CHECKING to LTL satisfiability checking, which is known to be in PSPACE [Sistla and Clarke, 1985]. Let $\varphi \in L(\Phi)$. To check whether $\rho(\vec{\sigma}) \models \varphi$, we simply ask whether $TH(\vec{\sigma}) \wedge \varphi$ is satisfiable. Recall that the size of $TH(\vec{\sigma})$ is polynomial in the size of $\vec{\sigma}$. It suffices to show that, $TH(\vec{\sigma}) \wedge \varphi$ is satisfiable if and only if $\rho(\vec{\sigma}) \models \varphi$. The implication from left to right is immediate by Lemma 3 (for $S = N$). For the opposite direction, by Lemma 2, we have $\rho(\vec{\sigma}') \models TH(\vec{\sigma})$ for some $\vec{\sigma}'$. Lemma 3 then yields $\rho(\vec{\sigma}') \models \varphi$ and the result follows. ∎

With this result in place, we can move on to consider problems specifically related to equilibria in iterated Boolean games. First, consider the following problem:

Given: Game G, strategy profile $\vec{\sigma}$.
MEMBERSHIP: Is it the case that $\vec{\sigma} \in NE(G)$?

Using Observation 4, and the construction $th(\sigma_i)$ for strategies σ_i—the denotational model for machine strategies—that was given before, we have the following result.

Proposition 6 MEMBERSHIP *is PSPACE-complete.*

Proof sketch: We give an algorithm that is in PSPACE and solves MEMBERSHIP for iterated Boolean games:

```
1. for i := 1 to n do
2.   if ρ(σ⃗) ⊭ γ_i then
3.     if γ_i ∧ ⋀_{j∈N\{i}} th(σ_j) is satisfiable then
4.       return "no"
5.     end-if
6.   end-if
7. end-for
8. return "yes"
```

Because of Proposition 5, we know that line (2) can be solved using a PSPACE oracle for MODEL CHECKING; moreover, line (3) uses a PSPACE oracle for satisfiability.

The algorithm checks whether there is some player i such that $\rho(\vec{\sigma}) \not\models \gamma_i$ and $\gamma_i \wedge \bigwedge_{j \in N \setminus \{i\}} th(\sigma_j)$ is satisfiable. It outputs "no" if this is the case and "yes," otherwise. Soundness follows immediately from Lemma 3 and Observation 4.

For hardness, reduce LTL satisfiability to MEMBERSHIP. Let φ be an LTL formula in $L(\Phi)$ and let ρ^0 be the run such that $\rho^0[t] = \emptyset$ for all $t \in \mathbb{N}$. Without loss of generality, we may assume that $\rho^0 \not\models \varphi$. Define G as the one-player iterated Boolean game in which player i has φ as goal. Let $\sigma_i = (Q_i, q_i^0, \delta_i, \tau_i)$ be the *partial* machine strategy with $Q_i = \{q_i^0\}$ and $\delta_i = \tau_i = \emptyset$. Thus, $\rho(\sigma_i) = \rho^0$ and $\rho(\sigma_i) \not\models \varphi$. Clearly, σ_i is polynomial in the size of φ. It can now readily be appreciated that φ is satisfiable if and only if $\sigma_i \notin NE(G)$. As PSPACE is closed under complement, the result follows. ∎

Thus, checking membership is no more complex than LTL model checking. However, other apparently closely related problems turn out to be much harder. Consider the following:

Given: Game G, LTL formula $\varphi \in L$.
E-NASH: Does $\exists \vec{\sigma} \in NE(G).\ \rho(\vec{\sigma}) \models \varphi$ hold?

Given: Game G, LTL formula $\varphi \in L$.
A-NASH: Does $\forall \vec{\sigma} \in NE(G).\ \rho(\vec{\sigma}) \models \varphi$ hold?

Given: Game G.
NON-EMPTINESS: Is it the case that $NE(G) \neq \emptyset$?

E-NASH is basically the *rational synthesis* problem for pure Nash equilibrium as first analysed by Fisman *et al*, whose complexity is 2EXPTIME-complete (Theorem 2 in Fisman *et al* [Fisman *et al.*, 2010])[3]. Since 2EXPTIME is a deterministic complexity class, 2EXPTIME-completeness of A-NASH immediately follows as a corollary.

Proposition 7 ([Fisman *et al.*, 2010]) *Both* A-NASH *and* E-NASH *are 2EXPTIME-complete.*

We can now show that NON-EMPTINESS is also complete for 2EXPTIME. For membership we ask whether $(G, \top)$ is accepted as an instance of E-NASH; for hardness we reduce the *LTL realizability* problem [Pnueli and Rosner, 1989b], which is known to be 2EXPTIME-complete.

Proposition 8 NON-EMPTINESS *is 2EXPTIME-complete.*

[3] Although we consider machine strategies rather than strategies as functions between partial plays, our strategy model is known to be sufficiently powerful for LTL goals [Pnueli and Rosner, 1989a].

5 Folk Theorems

In game theory much of the interest in iterated games derives from the *Nash Folk Theorems* [Osborne and Rubinstein, 1994, p.143]. These theorems tell us that the range of outcomes that can be sustained as equilibria in iterated games is much wider than one might at first suspect from the component game. Their usual form is that the set of *all* feasible and individually rational payoff vectors are achievable as Nash equilibria in iterated games; the definitions of feasibility and individual rationality depend on the precise setting considered.

To take a famous example, the Nash Folk Theorems tell us that cooperation can be sustained in the iterated Prisoner's Dilemma. The standard device for proving Folk Theorems is a *trigger strategy* [Osborne and Rubinstein, 1994, p.143]. A trigger strategy intended to obtain a particular outcome works by punishing any player who deviates from behaviour leading to the intended outcome: no player can benefit from deviation, as this would result in punishment by all other players. The desired outcome is thereby obtained as an equilibrium.

It seems very natural, therefore, to consider Folk Theorems in the context of our iterated Boolean games. Given our interest in using LTL to express properties of equilibria, we can formulate the question of which equilibria can be sustained in an iterated Boolean game in the following way:

Which LTL properties are preserved by the Nash equilibria of a given iterated Boolean game?

This question is closely related to the rational synthesis problem proposed by [Fisman *et al.*, 2010], and here formalised in the E-NASH problem. However, the rational synthesis problem, in effect, pertains to *particular* LTL properties being realised in some Nash equilibrium, while our concern is rather with characterising *the set of LTL formulae that can be satisfied in equilibria of iterated Boolean games*.

In this section, we show that the concepts and techniques used to prove the Nash Folk Theorems can be adapted to our setting. In particular, for games in which players have *safety goals* (of the form $\mathbf{G}\,\varphi$), we use a punishment strategy construction, similar in spirit to that used to prove the Nash Folk Theorems, to characterise precisely the circumstances under which arbitrary LTL formulae are satisfied in some equilibrium of an iterated Boolean game.

To be able to use such a construction, we need to be able to define for our setting some counterpart of the notion of a feasible and individually rational payoff for each player. In game theory, a payoff is individually rational and feasible if it could be enforced by the set of all other players in the game. In our setting, players have goals, rather than payoffs, and so we need to formulate the concept with respect to whether a player's goal can be falsified by the set of all other players.

Punishable players and goals A player i, with goal γ_i, is *punishable* if (at any point of time) i's opponents can jointly find values for the propositional variables under their control that guarantee γ_i to be false no matter which values i chooses for its variables. Nash Folk Theorems may hold only if all players, at all times, are punishable. This is indeed the case for games with certain kinds of goals, which we define next.

We say a goal γ_i is *non-trivial* if both γ_i and $\neg\gamma_i$ are satisfiable (*i.e.*, γ_i is neither a tautology nor a contradiction in LTL). We say γ_i is a *safety goal* when $\gamma_i = \mathbf{G}\,\varphi_i$ for some LTL formula φ_i. When φ_i is a propositional formula we say that γ_i is a *propositional safety goal*. Folk theorems for iterated Boolean games with safety goals are presented next.

Propositional Safety Goals Let *punishable*(i) be a predicate that formalises when player i is punishable, that is if a trigger strategy can be constructed against i. For games with propositional safety goals the predicate *punishable*(i) reduces to: $\bigvee_{v_{-i}\in V_{-i}} \bigwedge_{v_i'\in V_i} \left(\chi_{(v_{-i},v_i')} \rightarrow \neg\varphi_i\right)$ is valid, for a propositional safety goal $\gamma_i = \mathbf{G}\,\varphi_i$. We say that a player i of an iterated game is *punishable* when the predicate *punishable*(i) holds (and it is not punishable otherwise). We then have that:

Lemma 9 *Let $G = (N, \Phi, \Phi_1, \ldots, \Phi_n, \gamma_1, \ldots, \gamma_n)$ be an iterated Boolean game in which player i has a propositional safety goal $\gamma_i = \mathbf{G}\,\varphi_i$. Then, if player i is not punishable, for all $\vec{\sigma} \in NE(G)$ we have that $\rho(\vec{\sigma}) \models \gamma_i$.*

Proof sketch: W.l.o.g. assume that $i = n$ and that player i is not punishable, that is, $\bigwedge_{v_{-i}\in V_{-i}} \bigvee_{v_i'\in V_i} \left(\chi_{(v_{-i},v_i')} \wedge \varphi_i\right)$ is satisfiable. There is a function $f_n\colon V_{-n} \rightarrow V_n$ such that for each valuation $\vec{v}_{-n} \in V_{-n}$ we have $\rho(\vec{v}_{-n}, f_n(\vec{v}_{-n})) \models \varphi_n$. Consider an arbitrary $\vec{\sigma} \in NE(G)$, where $\sigma_i = (Q_i, q_i^0, \delta_i, \tau_i)$ for each player i. If $\rho(\vec{\sigma})$ satisfies $\mathbf{G}\,\varphi_i$ we are done. So assume $\rho(\vec{\sigma}) \not\models \mathbf{G}\,\varphi_i$. It now suffices to specify a strategy $\sigma_n^* = (Q_n^*, q_n^{00}, \delta_n^*, \tau_n^*)$ for player n such that $\rho(\vec{\sigma}_{-n}, \sigma_n^*) \models \mathbf{G}\,\varphi_n$. This is achieved by a product construction, in particular $Q_n^* = Q_1 \times \cdots \times Q_{n-1} \times V$, and the global behaviour of the profile $\vec{\sigma}_{-n}$ against every possible behaviour of n is mimicked in the states of σ_n^*. Thus, for each time point t, the strategy σ_n^* can predict what $\vec{v}_{-i} = \rho[t] \cap (\Phi \setminus \Phi_i)$ will be and output an appropriate valuation $f(\vec{v}_{-n}) = v_i'$ ensuring that $(\vec{v}_{-n}, v_n') \models \varphi_n$. Hence, we have that $\rho(\vec{\sigma}_{-n}, \sigma_n^*) \models \mathbf{G}\,\varphi_n$, meaning that the strategy profile $\vec{\sigma}$ is not a pure Nash equilibrium. ∎

Now we are in a position to prove a Folk Theorem for iterated Boolean games with propositional safety goals.

Theorem 10 *Let $G = (N, \Phi, \Phi_1, \ldots, \Phi_n, \gamma_1, \ldots, \gamma_n)$ be an iterated Boolean game in which each player i has a non-trivial propositional safety goal $\gamma_i = \mathbf{G}\,\varphi_i$. Then, the following two statements are equivalent:*

(*i*) *all players are punishable, and*

(*ii*) *for all satisfiable $\psi \in L$, there is $\vec{\sigma} \in NE(G)\colon \rho(\vec{\sigma}) \models \psi$.*

Proof sketch: For the direction from (*i*) to (*ii*) consider an arbitrary satisfiable formula ψ and assume all players to be punishable, that is, $\bigvee_{v_{-i}\in V_{-i}} \bigwedge_{v_i'\in V_i} (\chi_{(v_{-i},v_i')} \rightarrow \neg\varphi_i)$ is valid for all players i. Then, by Lemma 1, there is some strategy profile $\vec{\sigma}$ such that $\rho(\vec{\sigma}) \models \psi$. Moreover, for every player j, there is some $(v_1^j, \ldots, v_n^j) \in V$ such that for all $v_j' \in V_j$ we have $(v_1^j, \ldots, v_{j-1}^j, v_j', v_{j+1}^j, \ldots, v_n^j) \not\models \varphi_j$.

For every player i we build a machine strategy $\sigma_i^* = (Q_i^*, q_i^0, \delta_i^*, \tau_i^*)$ with $Q_i^* = (Q_1 \times \cdots \times Q_n) \cup \{q_i^1, \ldots, q_i^n\}$, where each q_i^j is an additional state. These strategies are designed so that $\rho(\vec{\sigma}^*) = \rho(\vec{\sigma})$, that is, if all players play the

strategy σ_i^* exhibits the same global behaviour as the equilibrium profile $\vec{\sigma}$. Hence, $\rho(\vec{\sigma}^*) \models \psi$.

However, if (exactly) one player j adopts a strategy σ_j' such that $\rho(\vec{\sigma}_{-j}^*, \sigma_j') \neq \rho(\vec{\sigma}^*)$, and let t be the earliest time point such that $\rho(\vec{\sigma}_{-j}^*, \sigma_j')[t] \neq \rho(\vec{\sigma}^*)[t]$, then $\vec{\sigma}^*$ is designed so that every player i distinct from j moves to state q_i^j and subsequently chooses v_i^j at $t+1$. Thus, at $t+1$ player j is punished in the sense that φ_j is not satisfied at $t+1$ by $\rho(\vec{\sigma}_{-j}^*, \sigma_j')[t+1]$. It follows that $\rho(\vec{\sigma}_{-j}^*, \sigma_j') \not\models \mathbf{G}\,\varphi_j$ and that j cannot achieve his goal by deviating from $\vec{\sigma}^*$. We may conclude that $\vec{\sigma}^* = (\sigma_1^*, \ldots, \sigma_n^*) \in NE(G)$.

For the direction from (*ii*) to (*i*) we prove the contrapositive. Assume that some player i is not punishable. As γ_i was assumed to be non-trivial, $\neg\gamma_i$ is satisfiable. Yet, by Lemma 9, we have that $\rho(\vec{\sigma}) \models \gamma_i$ for all equilibria $\vec{\sigma} \models \gamma_i$. Hence, there is no $\vec{\sigma} \in NE(G)$ with $\rho(\sigma) \models \neg\gamma_i$. ∎

As seen in Lemma 9 non-punishable players invariably achieve their goals in any Nash equilibrium. The next Folk Theorem, which is proved by using virtually the same constructions and arguments as in Theorem 10, refers to situations where non-punishable players have to be considered.

Theorem 11 *Let $G = (N, \Phi, \Phi_1, \ldots, \Phi_n, \gamma_1, \ldots, \gamma_n)$ be an iterated Boolean game in which each player i has a propositional safety goal $\gamma_i \equiv \mathbf{G}\,\varphi_i$. Then, for all $\psi \in L$ such that $\psi \wedge \bigwedge\{\gamma_i : \gamma_i \text{ is not punishable}\}$ is satisfiable, there is a $\vec{\sigma} \in NE(G)$ such that $\rho(\vec{\sigma}) \models \psi$.*

We can now leverage Theorem 10 and Theorem 11 so as to obtain PSPACE-completeness for the E-NASH problem for iterated Boolean games with propositional safety goals.

Proposition 12 *The* E-NASH *and* A-NASH *problems for iterated Boolean games with propositional safety goals are* PSPACE*-complete.*

Proof sketch: For membership in PSPACE, consider an arbitrary game $G = (N, \Phi, \Phi_1, \ldots, \Phi_n, \gamma_1, \ldots, \gamma_n)$ with propositional safety goals and an arbitrary $\psi \in L$. To decide whether ψ is sustained by some pure Nash equilibrium of G we proceed as follows.

```
1.   N_0 := ∅
2.   for i := 1 to n do
3.      if ∀Φ \ Φ_i . ∃Φ_i . φ is true
4.         then N_0 := N_0 ∪ {i}
5.         else N_0 := N_0
6.      end if
7.   end-for
8.   if ψ ∧ ⋀_{i∈N_0} γ_i is satisfiable then
9.      return "yes"
10.     else return "no"
11.  end-if
```

In the for-loop, that is, in lines (1) through (7), this algorithm singles out the players that are not punishable. Checking the quantified Boolean formula $\forall\Phi \setminus \Phi_i\,.\,\exists\Phi_i\,.\,\varphi$ for truth is equivalent to checking $\bigwedge_{v_{-i}\in V_{-i}} \bigvee_{v_i'\in V_i} (\chi_{(v_{-i},v_i')} \wedge \varphi_i)$ for satisfiability and can be achieved in Σ_2^p. Soundness of this step then follows from Lemma 9. In step (8) satisfiability of $\psi \wedge \bigwedge_{i\in N_0} \gamma_i$ is checked, which can be achieved in PSPACE. As PSPACE subsumes Σ_2^p, the algorithm runs in PSPACE.

For soundness and completeness, it now suffices to show that the following two statements are equivalent:

(*i*) $\psi \wedge \bigwedge_{i\in N_0} \gamma_i$ is satisfiable

(*ii*) there is a $\vec{\sigma} \in NE(G)$ with $\rho(\vec{\sigma}) \models \psi$.

For the direction from (*ii*) to (*i*), assume that there is a $\vec{\sigma} \in NE(G)$ with $\rho(\vec{\sigma}) \models \psi$. By Lemma 9, it then follows that $\rho(\vec{\sigma}) \models \gamma_i$ for all $i \in N_0$. Hence, $\psi \wedge \bigwedge_{i\in N_0} \gamma_i$ is satisfiable. The opposite direction is immediate by Theorem 11.

For PSPACE-hardness, we reduce the LTL satisfiability problem (which is known to be PSPACE-complete). Let ψ be an arbitrary LTL formula in $L(\Phi)$. We construct a game with two punishable players, both having propositional safety goals. The result then follows from Theorem 10. ∎

General Safety Goals For general safety goals $\gamma_i = \mathbf{G}\,\varphi_i$ the predicate *punishable*(i) holds iff there is some LTL satisfiable formula $\chi \in L(\Phi \setminus \Phi_i)$ such that $\chi \rightarrow \neg\,\mathbf{G}\,\varphi_i$ is valid. Based on this definition the next Folk Theorem follows:

Theorem 13 *Let $G = (N, \Phi, \Phi_1, \ldots, \Phi_n, \gamma_1, \ldots, \gamma_n)$ be an iterated Boolean game where each player i has a non-trivial safety goal $\gamma_i = \mathbf{G}\,\varphi_i$. Then, if all players are punishable, for all satisfiable $\psi \in L$, there is $\vec{\sigma} \in NE(G)$ with $\rho(\vec{\sigma}) \models \psi$.*

Proof sketch: Assume that all players are punishable, that is, for each player i, there is some satisfiable $\chi_i \in L(\Phi_{-i})$ such that $\chi_i \rightarrow \neg\,\mathbf{G}\,\varphi_i$ is valid. Moreover, let $\psi \in L$ be a satisfiable formula. By Lemma 1 there is are strategy profiles $\vec{\sigma}(\psi), \vec{\sigma}(\chi_1), \ldots, \vec{\sigma}(\chi_n)$, the runs induced by which satisfy ψ and χ_1 through χ_n, respectively. On their basis and by means of a product construction very similar to the one for Theorem 10, we can build for every player i a machine strategy $\sigma_i^* = (Q_i^*, q_i^{00}, \delta_i^*, \tau_i^*)$ that incorporates a punishment strategy against every other player. It can then be shown that $\rho(\vec{\sigma}^*) \models \psi$ and $\vec{\sigma}^* \in NE(G)$. ∎

6 Future work

In the model of iterated Boolean games we studied in this paper the players' goals are represented by LTL formulae. It is also natural to consider players whose behaviour, as motivated by their goals, would be better described by branching-time temporal logics, such as CTL* [Emerson and Halpern, 1986] or the μ-calculus [Kozen, 1983]. Most probably a framework of this kind would lead to systems (games) with different computational complexity properties.

Another important technical component of this work was the formalisation of semantic conditions underpinning the Folk Theorems. Our study covered so-called (propositional) safety goals; other types of goals should be studied as well, for instance, goals with fairness or response properties.

We plan to analyse iterated Boolean games by means of other solution concepts. Punishment strategies arguably involve incredible threats. Subgame-perfect equilibrium seems to be appealing in this respect, as it does not suffer from this (alleged) defect and, moreover, Folk Theorems have been shown to hold for it (*e.g.*, [Fudenberg and Maskin, 1986]).

References

[Bonzon *et al.*, 2009] Elise Bonzon, Marie-Christine Lagasquie-Schiex, and Jérôme Lang. Dependencies between players in boolean games. *Int. J. Approx. Reasoning*, 50(6):899–914, 2009.

[Chatterjee and Henzinger, 2012] Krishnendu Chatterjee and Thomas A. Henzinger. A survey of stochastic ω-regular games. *J. Comput. Syst. Sci.*, 78(2):394–413, 2012.

[Chatterjee *et al.*, 2008] Krishnendu Chatterjee, Thomas A. Henzinger, and Barbara Jobstmann. Environment assumptions for synthesis. In *CONCUR*, volume 5201 of *LNCS*, pages 147–161. Springer, 2008.

[Diekert and Gastin, 2008] Volker Diekert and Paul Gastin. First-order definable languages. In *Logic and Automata*, volume 2 of *Texts in Logic and Games*, pages 261–306. Amsterdam University Press, 2008.

[Dunne *et al.*, 2008] Paul E. Dunne, Wiebe van der Hoek, Sarit Kraus, and Michael Wooldridge. Cooperative boolean games. In *AAMAS (2)*, pages 1015–1022. IFAAMAS, 2008.

[Emerson and Halpern, 1986] E. Allen Emerson and Joseph Y. Halpern. "sometimes" and "not never" revisited: on branching versus linear time temporal logic. *J. ACM*, 33(1):151–178, 1986.

[Emerson, 1990] E. Allen Emerson. Temporal and modal logic. In *Handbook of Theoretical Computer Science, Volume B: Formal Models and Sematics (B)*, pages 995–1072. 1990.

[Fisman *et al.*, 2010] Dana Fisman, Orna Kupferman, and Yoad Lustig. Rational synthesis. In *TACAS*, volume 6015 of *LNCS*, pages 190–204. Springer, 2010.

[Fudenberg and Maskin, 1986] D. Fudenberg and E. Maskin. The folk theorem in repeated games with discounting or with incomplete information. *Econometrica*, pages 533–554, 1986.

[Grädel and Ummels, 2008] Erich Grädel and Michael Ummels. Solution concepts and algorithms for infinite multiplayer games. In *New Perspectives on Games and Interaction*, volume 4 of *Texts in Logic and Games*, pages 151–178. Amsterdam University Press, 2008.

[Grant *et al.*, 2011] John Grant, Sarit Kraus, Michael Wooldridge, and Inon Zuckerman. Manipulating boolean games through communication. In *IJCAI*, pages 210–215. IJCAI/AAAI, 2011.

[Harrenstein *et al.*, 2001] P. Harrenstein, W. van der Hoek, J.-J.Ch. Meyer, and C. Witteveen. Boolean games. In *Theoretical Aspects of Rationality and Knowledge (TARK VIII)*, pages 287–298, 2001.

[Kozen, 1983] Dexter Kozen. Results on the propositional mu-calculus. *Theor. Comput. Sci.*, 27:333–354, 1983.

[Kupferman *et al.*, 2000] Orna Kupferman, P. Madhusudan, P. S. Thiagarajan, and Moshe Y. Vardi. Open systems in reactive environments: Control and synthesis. In *CONCUR*, volume 1877 of *LNCS*, pages 92–107. Springer, 2000.

[Lustig and Vardi, 2009] Yoad Lustig and Moshe Y. Vardi. Synthesis from component libraries. In *FOSSACS*, volume 5504 of *LNCS*, pages 395–409. Springer, 2009.

[Manna and Pnueli, 1992] Zohar Manna and Amir Pnueli. *The temporal logic of reactive and concurrent systems - specification*. Springer, 1992.

[Manna and Pnueli, 1995] Zohar Manna and Amir Pnueli. *Temporal verification of reactive systems - safety*. Springer, 1995.

[Osborne and Rubinstein, 1994] M. J. Osborne and A. Rubinstein. *A Course in Game Theory*. The MIT Press, 1994.

[Pnueli and Rosner, 1988] Amir Pnueli and Roni Rosner. A framework for the synthesis of reactive modules. In *Concurrency*, volume 335 of *LNCS*, pages 4–17. Springer, 1988.

[Pnueli and Rosner, 1989a] Amir Pnueli and Roni Rosner. On the synthesis of a reactive module. In *POPL*, pages 179–190. ACM Press, 1989.

[Pnueli and Rosner, 1989b] Amir Pnueli and Roni Rosner. On the synthesis of an asynchronous reactive module. In *ICALP*, volume 372 of *LNCS*, pages 652–671. Springer, 1989.

[Sistla and Clarke, 1985] A. Prasad Sistla and Edmund M. Clarke. The complexity of propositional linear temporal logics. *J. ACM*, 32(3):733–749, 1985.

[Thomas, 1990] Wolfgang Thomas. Automata on infinite objects. In *Handbook of Theoretical Computer Science, Volume B: Formal Models and Sematics (B)*, pages 133–192. 1990.

[Ummels, 2006] Michael Ummels. Rational behaviour and strategy construction in infinite multiplayer games. In *FSTTCS*, volume 4337 of *LNCS*, pages 212–223. Springer, 2006.

[Vardi, 2008] Moshe Y. Vardi. From verification to synthesis. In *VSTTE*, volume 5295 of *LNCS*, page 2. Springer, 2008.