The complexity of decision problems about equilibria in two-player Boolean games

Egor Ianovski[a], Luke Ong[a]

[a] *Department of Computer Science, Wolfson Building, Parks Rd, Oxford OX1 3QD, United Kingdom*

Abstract

Boolean games allow us to succinctly represent strategic games with binary payoffs in the case where the players' preferences have a structure readily expressible in propositional logic. Since their introduction, the computational aspects of Boolean games have been of interest to the multiagent community, but so far the focus has been exclusively on pure strategy equilibria. In this paper we consider the complexity of problems involving mixed strategy equilibria, such as the existence of an equilibrium satisfying a given payoff constraint. The results are obtained by the observation that a mixed strategy can hold enough information to encode the computation history of an exponential time Turing machine.

Keywords: Game theory, Boolean games, complexity, propositional logic

1. Introduction

The ideas of game theory find fertile ground in the multiagent framework. Any analysis of a system composed of self-interested agents, whether human or otherwise, cannot ignore the fact that an agent's behaviour is going to be influences by the choices of the agents around him, and in turn his choice of action will necessitate that the other agents respond to him.

Unfortunately games tend to get very large very quickly—the standard normal form representation is of size $O(S^n)$ for a game with n players possessing S strategies each. This has motivated the study of succinct game representations.

The Boolean game (Harrenstein et al., 2001) is a natural and expressive game representation that achieves succinctness by interpreting a player's strategy as an assignment of truth values and his preferences as a formula of propositional logic. Given n players with formulae of length k, this gives a representation of size $O(nk)$. The dependency on the number of players is reduced to linear, and while no a priori upper bound can be given on k, non-trivial games can be represented with formulae of size logarithmic in the number of strategies.

There is a wide literature on complexity considerations for Boolean games and related extensions, but, with the exception of a preliminary version of this

paper (Ianovski and Ong, 2014), these have said next to nothing about the complexity of mixed equilibria.

1.1. Our contribution

We demonstrate that the following problems are NEXP-complete for two-player Boolean games:

- Given a Boolean game G and a vector of payoffs $\boldsymbol{v}$, determine whether G has an equilibrium where Player i gets at least $\boldsymbol{v}[i]$ utility.
- Given a Boolean game G and a formula of propositional logic φ, determine whether G has an equilibrium where φ is satisfied with a probability of 1.

And that the following are coNEXP-complete for two-player Boolean games:

- Given a Boolean game G and a vector of payoffs $\boldsymbol{v}$, determine whether Player i gets at least $\boldsymbol{v}[i]$ utility in every equilibrium of G.
- Given a Boolean game G and a formula of propositional logic φ, determine whether φ is satisfied with a probability of 1 in every equilibrium of G.
- Given a Boolean game G, determine whether G has a unique equilibrium.

1.2. Related work

Boolean games were first introduced by Harrenstein et al. (2001) as two-player, zero-sum games induced by a formula of propositional logic. The multi-player definition used today is due to Bonzon et al. (2006). Algorithmic studies (e.g. Dunne and van der Hoek, 2004; Bonzon et al., 2006; Dunne and Wooldridge, 2012) focused on the complexity of problems involving pure equilibria, where it was shown that determining whether a pure strategy equilibrium exists is Σ_2^p-complete, but the complexity can be lower if the players' goal formulae are restricted to a tractable fragment of propositional logic.

Boolean games can be seen to be a special case of circuit games (Fortnow et al., 2005; Schoenebeck and Vadhan, 2012). In a circuit game a player is equiped with a Boolean circuit, control of the input gates is distributed among the players, and the output of the circuit is an integer representing the player's payoff. A Boolean game, then, is a circuit game where the output of the circuit is limited to 0 or 1 and the circuit has fan-out 1. As a consequence of this, a complexity result for circuit games is an upper bound for the complexity of the same problem for Boolean games, and a result for Boolean games is a lower bound for circuit games. In particular, this means the result of Schoenebeck and Vadhan (2012) that ∃GuaranteeNash is NEXP-complete for circuit games does not imply the result in this paper.

2. Preliminaries

2.1. Strategic and Boolean games

We start with standard notions of game theory:

Definition 2.1. A *strategic game* is a triple $(N, \{ S_1, \ldots, S_n \}, \{ u_1, \ldots, u_n \})$. N is a finite set of *players*, of cardinality n. S_i is a finite set of Player i's *pure strategies*. An n-tuple of pure strategies, i.e. a member of $\mathcal{S} = S_1 \times \cdots \times S_n$ is called a *pure-strategy profile*. The function $u_i : \mathcal{S} \to \mathbb{R}$ is Player i's *utility function*.

A strategic game is called *zero-sum* just if there exists a $c \in \mathbb{R}$ such that for every $\boldsymbol{s} \in \mathcal{S}$:

$$\sum_{i \in N} u_i(\boldsymbol{s}) = c. \qquad \blacksquare$$

Definition 2.2. A pure-strategy profile $\boldsymbol{s}$ is called a *pure-strategy equilibrium* just if for each $i \in N$, for all $s' \in S_i$:

$$u_i(\boldsymbol{s}) \geq u_i(\boldsymbol{s_{-i}}(s')),$$

where $\boldsymbol{s_{-i}}(s')$ denotes the profile obtained by replacing Player i's strategy in $\boldsymbol{s}$ with s'. $\blacksquare$

Definition 2.3. Let $\mathcal{P}(S_i)$ denote the space of probability distributions over S_i. A *mixed strategy* for Player i is a member of $\mathcal{P}(S_i)$. The weight assigned to a pure strategy s by a mixed strategy σ, or $P(s \mid \sigma)$, is called the *strategy weight* of s.

An n-tuple of mixed strategies, $\boldsymbol{\sigma} \in \mathcal{P}(\mathcal{S})$, is called a *mixed-strategy profile*. We extend Player i's utility function to the space of mixed-strategy profiles on the principle of expected utility. That is:

$$u_i(\boldsymbol{\sigma}) = \sum_{\boldsymbol{s} \in \mathcal{S}} u_i(\boldsymbol{s}) P(\boldsymbol{s} \mid \boldsymbol{\sigma}).$$

A mixed-strategy profile is called a *mixed-strategy equilibrium* just if for all $s' \in S_i$:

$$u_i(\boldsymbol{\sigma}) \geq u_i(\boldsymbol{\sigma_{-i}}(s')). \qquad \blacksquare$$

We have need of two classical results about games.

Theorem 2.4 (von Neumann (1928))**.** *In every two-player zero-sum game there exists a v such that Player One gets a utility of v in every equilibrium. This v is called the* value *of the game.*

Theorem 2.5 (Nash (1951))**.** *Every strategic game has an equilibrium in mixed strategies.*

A Boolean game is a succinct way of representing a strategic game.

Definition 2.6 (Harrenstein et al. (2001); Bonzon et al. (2006))**.** A *Boolean game* is a triple $(N, \{\Phi_1, \ldots, \Phi_n\}, \{\gamma_1, \ldots, \gamma_n\})$. The Φ_i are mutually disjoint sets of propositional variables and each γ_i is a formula of propositional logic defined over $\biguplus_{i \in N} \Phi_i$. These components form a representation of a strategic game—Player i's set of pure strategies is the truth assignments to Φ_i, i.e. $S_i = 2^{\Phi_i}$, and Player i's utility function is:

$$u_i(\boldsymbol{\nu}) = \begin{cases} 1 \text{ if } \boldsymbol{\nu} \vDash \gamma_i, \\ 0 \text{ otherwise.} \end{cases}$$

■

If a strategic game is represented explicitly, i.e. by listing the elements of S_i and the graph of u_i, then such a representation will need $O(n|\mathcal{S}|)$ space, and $|\mathcal{S}| = |S_1 \times \cdots \times S_n|$ is exponential in the number of players. The Boolean representation needs $O(\sum_{i=1}^{n} |\gamma_i|)$ space. The dependency on the number of players is linear, and $|\gamma_i|$ could be as small as $\log |S_i|$, seeing how only $\log |S_i|$ variables are needed to represent $2^{\log |S_i|}$ strategies. There is no upper bound on $|\gamma_i|$, as any formula of propositional logic is logically equivalent to infinitely many other formulae.

Since a Boolean game is simply a representation of a strategic game, every result about strategic games carries over to the Boolean case. In particular, this includes Theorem 2.4 and Theorem 2.5.

Figure 2.1 shows the algorithmic problems studied in this paper. The problems ∃GuaranteeNash, ∀GuaranteeNash and UniqueNash are well defined for all representations of strategic games and are known to be NP-complete, coNP-complete, and coNP-complete for two-player games in normal form (Gilboa and Zemel, 1989). ∀NashSat and ∃NashSat are defined for Boolean games only.

2.2. Encoding integers in logic

The goal of this section is to introduce shorthand and notation that will allow us to discuss integers and arithmetic in the language of propositional logic. This will ultimately allow us to interpret a player's strategy as a choice of integers, and a goal formula as an arithmetical constraint.

Our constructions will often involve sequences of propositional variables, so we will use the notation $\overline{p_m}$ to mean $p_1, \ldots, p_m$. Sequences differ from sets in that they have an order, which allows us to interpret a truth assignment to a sequence as an integer. We do this in the standard way—a truth assignment that sets p_i to *true* defines a binary integer where the ith most significant bit is 1.

Definition 2.7. Let $\overline{p_m}$ be a sequence of m propositional variables, and ν a truth assignment to $\overline{p_m}$. We use $[\![\overline{p_m}]\!]_\nu$ to denote the numeric value associated

∃GuaranteeNash	
Input:	A representation of a game G and a vector of payoffs $\boldsymbol{v}$.
Output:	YES if there exists an equilibrium $\boldsymbol{\sigma}$ of G such that $u_i(\boldsymbol{\sigma}) \geq \boldsymbol{v}[i]$ for all i, NO otherwise.

∀GuaranteeNash	
Input:	A representation of a game G and a vector of payoffs $\boldsymbol{v}$.
Output:	YES if for all equilibria $\boldsymbol{\sigma}$ of G, $u_i(\boldsymbol{\sigma}) \geq \boldsymbol{v}[i]$ for all i, NO otherwise.

UniqueNash	
Input:	A representation of a game G.
Output:	YES if G has a unique equilibrium, NO otherwise.

∃NashSat	
Input:	A Boolean game G and a propositional formula φ.
Output:	YES if there exists an equilibrium $\boldsymbol{\sigma}$ of G in which φ holds with probability 1, NO otherwise.

∀NashSat	
Input:	A Boolean game G and a propositional formula φ.
Output:	YES if in all equilibria $\boldsymbol{\sigma}$ of G, φ holds with probability 1, NO otherwise.

Figure 2.1: Algorithmic problems studied.

with ν via its assignment to $\overline{p_m}$. That is:[1]

$$[\![\overline{p_m}]\!]_\nu = \sum_{i=1}^{m} 2^{m-i}(\nu \vDash p_i).$$

If ν is clear from context, we just write $[\![\overline{p_m}]\!]$. ■

This allows us to define formulae that test truth assignments for arithmetic relations.

Lemma 2.8. *Let* $\boldsymbol{Equal}(\overline{p_m}; \overline{q_m})$ *denote a term that is true if and only if* $[\![\overline{p_m}]\!] = [\![\overline{q_m}]\!]$. *I.e.,* $\nu \vDash \boldsymbol{Equal}(\overline{p_m}; \overline{q_m})$ *if and only if* $[\![\overline{p_m}]\!]_\nu = [\![\overline{q_m}]\!]_\nu$.

We can construct $\boldsymbol{Equal}(\overline{p_m}; \overline{q_m})$ *in time linear in* m.

Proof. Two binary integers are equal if and only if they are bitwise equal. This gives us the following:

$$\boldsymbol{Equal}(\overline{p_m}; \overline{q_m}) = \bigwedge_{1 \leq i \leq m} (p_i \leftrightarrow q_i).$$

□

Lemma 2.9. *Let* $\boldsymbol{Succ}(\overline{p_m}; \overline{q_m})$ *denote a term that is true if and only if* $[\![\overline{p_m}]\!] + 1 = [\![\overline{q_m}]\!]$.

We can construct $\boldsymbol{Succ}(\overline{p_m}; \overline{q_m})$ *in time quadratic in* m.

Proof. $2^m - 1$ does not have an m bit successor, so we first require that $\overline{p_m}$ is not all 1s:

$$\boldsymbol{Succ}(\overline{p_m}; \overline{q_m}) = \neg(\bigwedge_{1 \leq i \leq m} p_i) \wedge Succ'.$$

$Succ'$ will do the heavy lifting.

For intuition, recall that adding a 1 to a binary integer x can have one of two outcomes: either x ends in a 0 and this 0 is replaced with a 1, or x ends with a 1 in which case that 1 is replaced with a 0 and a carry operation occurs—which is, of course, simply the act of adding a 1 to a rightshift of x.

In other words, we have a recursive operation that replaces the rightmost consecutive block of 1s in x with 0s, the subsequent 0 with a 1, and leaves the rest of the integer unchanged.

[1] $(\nu \vDash p_i)$ is 1 if $\nu \vDash p_i$, 0 otherwise.

This gives us $Succ'$:

$$\begin{aligned}
&\Big(\neg p_m \to \big(q_m \wedge \boldsymbol{Equal}(\overline{p_{m-1}}; \overline{q_{m-1}})\big)\Big)\\
\wedge&\Big(\big((p_m \wedge \neg p_{m-1}) \to \big(\neg q_m \wedge q_{m-1} \wedge \boldsymbol{Equal}(\overline{p_{m-2}}; \overline{q_{m-2}})\big)\big)\Big)\\
&\vdots\\
\wedge&\Big((\neg p_1 \wedge \bigwedge_{1<i\leq m} p_i) \to (q_1 \wedge \bigwedge_{1<i\leq m} \neg q_i)\Big).
\end{aligned}$$

For the reader who abhors ellipses, we can restate this in one line:

$$Succ' = \bigwedge_{i\leq m} \Big(\big(\neg p_i \wedge \bigwedge_{j>i} p_j\big) \to \big(\boldsymbol{Equal}(\overline{p_{i-1}}; \overline{q_{i-1}}) \wedge q_i \wedge \bigwedge_{j>i} \neg q_j\big)\Big).$$

This is quadratic in the number of variables, giving us the desired result. □

Lemma 2.10. *Let* $\boldsymbol{Less}(\overline{p_m}; \overline{q_m})$ *denote a term that is true under* ν *if and only if* $[\![\overline{p_m}]\!]_\nu < [\![\overline{q_m}]\!]_\nu$.

We can construct $\boldsymbol{Less}(\overline{p_m}; \overline{q_m})$ *in time cubic in* m.

Proof. Let $a[i]$ be the ith most significant bit of a.

Intuitively, if $[\![\overline{p_m}]\!] < [\![\overline{q_m}]\!]$ for two big-endian binary digits then if we read the bits of the two from left to right we will see a 0 in $[\![\overline{p_m}]\!]$ before we see one in $[\![\overline{q_m}]\!]$. That is, there exists a k such that:

$$\begin{gathered}
[\![\overline{p_k}]\!] = [\![\overline{q_k}]\!],\\
\nu \not\models p_{k+1}, \quad \nu \models q_{k+1}.
\end{gathered}$$

It follows that the first bit where the two integers differ is a 1 for $[\![\overline{q_m}]\!]$ and a 0 for $[\![\overline{p_m}]\!]$. This is clearly both necessary and sufficient.

Since there are only m possible values of k, this can be replaced by a cubic size formula that looks as follows:

$$\bigvee_{0\leq k<m} \boldsymbol{Succ}(\overline{p_k}; \overline{q_k}).$$

□

Corollary 2.11. *Let* $\boldsymbol{LessEq}(\overline{p_m}; \overline{q_m})$ *denote a term that is true under* ν *if and only if* $[\![\overline{p_m}]\!]_\nu \leq [\![\overline{q_m}]\!]_\nu$.

We can construct $\boldsymbol{LessEq}(\overline{p_m}; \overline{q_m})$ *in time cubic in* m.

Lemma 2.12. *Let* $\boldsymbol{AddMod}(\overline{p_m}; \overline{q_m}; \overline{r_m})$ *denote a term that is true if and only if* $[\![\overline{p_m}]\!] + [\![\overline{q_m}]\!] = [\![\overline{r_m}]\!] \mod 2^k$. *Let* $\boldsymbol{Add}(\overline{p_m}; \overline{q_m}; \overline{r_m})$ *denote a term that is true if and only if* $[\![\overline{p_m}]\!] + [\![\overline{q_m}]\!] = [\![\overline{r_m}]\!]$.

$\boldsymbol{AddMod}(\overline{p_m}; \overline{q_m}; \overline{r_m})$ *and* $\boldsymbol{Add}(\overline{p_m}; \overline{q_m}; \overline{r_m})$ *can both be replaced by a formula of propositional logic cubic in* m.

Proof. Binary addition is easy: 0 is identity, and $1 + 1$ is 0. In other words if the summands are the same the answer is 0, if they are different the answer is 1. Thus if no carry occurred at position $i+1$, then $r_i \leftrightarrow (\neg(p_i \wedge q_i))$. If a carry has occurred, we add another 1 to this result, which gives us $r_i \leftrightarrow (p_i \wedge q_i)$.

We can determine whether a carry bit reached position i with the following formula:

$$\mathit{Carry}(i) = \bigvee_{j \geq i+1} \Big((p_j \wedge q_j) \wedge \bigwedge_{i+1 \leq k < j} (p_k \vee q_k) \Big).$$

In words, $\mathit{Carry}(i)$ holds just if there exists a $j > i$ such that the bits at the jth position are both 1, and every single position between j and i has at least one of the bits being 1.

This gives us modular addition:

$$\begin{aligned} \boldsymbol{AddMod}(\overline{p_m}; \overline{q_m}; \overline{r_m}) = \bigwedge_{i \leq m} \Big[\Big(\neg \mathit{Carry}(i) \rightarrow \big(r_i \leftrightarrow (\neg(p_i \leftrightarrow q_i)) \big) \Big) \\ \wedge \Big(\mathit{Carry}(i) \rightarrow \big(r_i \leftrightarrow (p_i \leftrightarrow q_i) \big) \Big) \Big]. \end{aligned}$$

For exact addition, we need to make sure that no carry bit spills over past the 0th position:

$$\boldsymbol{Add}(\overline{p_m}; \overline{q_m}; \overline{r_m}) = \boldsymbol{AddMod}(\overline{p_m}; \overline{q_m}; \overline{r_m}) \wedge \neg \Big((p_1 \wedge q_1) \vee \big(\mathit{Carry}(0) \wedge (p_1 \vee q_1) \big) \Big).$$

This is cubic in m, proving the lemma. □

Lemma 2.13. *Let* $\boldsymbol{Sub}(\overline{p_m}; \overline{q_m}; \overline{r_m})$ *denote a term that is true if and only if* $[\![\overline{p_m}]\!] - [\![\overline{q_m}]\!] = [\![\overline{r_m}]\!]$.

$\boldsymbol{Sub}(\overline{p_m}; \overline{q_m}; \overline{r_m})$ *can be replaced by a formula of propositional logic cubic in* m.

Proof. Subtraction follows from the arithmetic identity:

$$a - b = c \iff a = c + b.$$

Hence:

$$\boldsymbol{Sub}(\overline{p_m}; \overline{q_m}; \overline{r_m}) = \boldsymbol{Add}(\overline{r_m}; \overline{q_m}; \overline{p_m}).$$

□

At times in lieu of testing $[\![\overline{p_m}]\!]$ for one of these three relations against $[\![\overline{q_m}]\!]$, we may wish to test, for example, whether $[\![\overline{p_m}]\!] < 3$. In other words, we would like to have access to constants as well as variables, and that is the purpose of the following definition:

Definition 2.14. Let j be a binary integer of length m. We use $\ulcorner \boldsymbol{j} \urcorner$ to denote a sequence of the logical constants $\top$ and $\bot$, the ith element of which is $\top$ if and only if the ith most significant bit of j is 1.

The intended use of $\ulcorner \boldsymbol{j} \urcorner$ is as an argument to the parametrised formulae defined so far. For example, we interpret $\boldsymbol{Less}(\overline{p_m}; \ulcorner \boldsymbol{j} \urcorner)$ in the sense of Lemma 2.10, except every instance of q_i is replaced with $\top$ or $\bot$ depending on whether the ith bit of j is a 1 or 0. ■

Finally, we need a generalised XOR.

Lemma 2.15. *Let $\boldsymbol{OneOf}(\overline{p_m})$ denote a term that is true just if exactly one of the p_i variables is true. $\boldsymbol{OneOf}(\overline{p_m})$ can be constructed in time quadratic in m.*

Proof.

$$\boldsymbol{OneOf}(\overline{p_m}) = \bigvee_{i \leq m} \left(p_i \wedge \left(\bigwedge_{j \neq i} \neg p_j\right)\right).$$

□

2.3. Games of any value

The reader will note that while the payoff for Player One in any *pure* profile in a Boolean game need be 1 or 0, there is no such restriction on *mixed* profiles—after all, the unique mixed equilibrium of Matching Pennies gives either player $1/2$. This will provide us with a convenient family of gadget games to facilitate proofs.

In a win-lose game, $v \in [0,1]_{\mathbb{Q}}$ is the probability of Player One winning the game. Given $v = a/b$ we can uniformly construct a family of games in which Player One chooses an interval (possibly looping around the edges) of length a over $[0, b-1]_{\mathbb{N}}$, and Player Two chooses a single integer from the same range. If a and b are coprime, i.e. the fraction a/b is maximally reduced, we can show that this game has a unique equilibrium—the profile where Player One randomises equally over every interval and Player Two over every integer—the value of which is clearly $a/b = v$. The vocabulary we have developed in Section 2.2 will allow us to express this as a Boolean game.

Lemma 2.16. *For $v \in [0,1]_{\mathbb{Q}}$ we can construct, in time polynomial in $|v|$, a two-player, zero-sum Boolean game with value v and a unique equilibrium.*

We shall refer to this game as $\mathfrak{G}(v)$.

Proof. The boundary cases where $v = 0$ or $v = 1$ are handled by creating a game with no variables for either player, with Player One's goal formula being $\bot$ or $\top$ respectively.

For the non-trivial cases, let $v = a/b$. We insist that a, b are coprime.

Consider the game where Player One selects two numbers $c_1, c_2 \in [0, b-1]_{\mathbb{N}}$ with the property that $c_2 - c_1 \equiv a - 1 \mod b$ (the start and end points of an interval of length a). Player Two selects $d \in [0, b-1]_{\mathbb{N}}$. The game is won by Player One if $c_2 \geq d \geq c_1$ (Player Two's choice is in a non-looping interval),

$d \geq c_1 > c_2$ (Player Two's choice is in a looping interval left of $b-1$) or $c_1 > c_2 \geq d$ (Player Two's choice is in a looping interval right of 0).

To give this game a Boolean rendition we need the following variables:

$$\begin{aligned}\Phi_1 =&\{p_1, \ldots, p_m, q_1, \ldots, q_m, s_1, \ldots, s_m, t_1, \ldots, t_m\},\\ \Phi_2 =&\{r_1, \ldots, r_m\}.\end{aligned}$$

The interpretation is that $[\![\overline{p_m}]\!] = c_1, [\![\overline{q_m}]\!] = c_2$ and $[\![\overline{r_m}]\!] = d$. The s and t variables come in to play if Player One wishes to play a looping interval, in which case $[\![\overline{s_m}]\!]$ is the distance between 0 and c_2, while $[\![\overline{t_m}]\!]$ is the distance between c_1 and $b-1$. These variables are added to give us a way to check that if Player One plays a looping interval, its length is still a.

Player One's goal formula is the following:

$$\begin{aligned}\gamma_1 =&\big(\boldsymbol{Sub}(\overline{q_m}, \overline{p_m}, \ulcorner \boldsymbol{a-1}\urcorner) \wedge \boldsymbol{LessEq}(\overline{q_m}, \ulcorner \boldsymbol{b-1}\urcorner)\\ &\wedge \boldsymbol{LessEq}(\overline{r_m}, \overline{q_m}) \wedge \boldsymbol{LessEq}(\overline{p_m}, \overline{r_m}) \wedge \boldsymbol{Equal}(\overline{s_m}, \ulcorner \boldsymbol{0}\urcorner)\\ &\wedge \boldsymbol{Equal}(\overline{t_m}, \ulcorner \boldsymbol{0}\urcorner)\big)\\ \vee&\Big(\boldsymbol{Add}(\overline{s_m}, \overline{t_m}, \ulcorner \boldsymbol{a-1}\urcorner) \wedge \boldsymbol{Sub}(\overline{q_m}, \ulcorner \boldsymbol{0}\urcorner, \overline{s_m})\\ &\wedge \boldsymbol{Sub}(\ulcorner \boldsymbol{b-1}\urcorner, \overline{p_m}, \overline{t_m}) \wedge \big(\boldsymbol{LessEq}(\overline{r_m}, \overline{q_m}) \vee \boldsymbol{LessEq}(\overline{p_m}, \overline{r_m})\big)\Big)\\ \vee&\boldsymbol{Less}(\ulcorner \boldsymbol{b-1}\urcorner, \overline{r_m}).\end{aligned}$$

The first disjunct handles the case where the interval is non-looping: $[\![\overline{q_m}]\!] - [\![\overline{p_m}]\!] = a-1$ (because the interval is closed) and Player Two's choice falls between them. We require that $[\![\overline{s_m}]\!] = [\![\overline{t_m}]\!] = 0$ to ensure the equilibrium is unique, as otherwise Player One would have been able to assign any value to those variables without affecting his chance of winning. The second disjunct handles the looping case; here $[\![\overline{s_m}]\!]$ and $[\![\overline{t_m}]\!]$ store the length of the interval on either side of zero. The last disjunct of serves to award the game to One should Two name a d outside of $[0, b-1]_{\mathbb{N}}$.

As the game is zero-sum, γ_2 is simply $\neg\gamma_1$.

Now let us verify that the equilibrium is unique. Consider an arbitrary profile $\boldsymbol{\sigma}$. We can without loss of generality assume that every strategy in the support of σ_2 sets $[\![\overline{r_m}]\!] \leq b-1$. Any strategy which does not would yield Player Two a utility of zero in every profile, and as such cannot be part of any equilibrium.

Likewise, recall that we are interpreting a pure strategy by Player One as selecting an interval of length a. Not every strategy available to him is amenable to this interpretation. First, Player One could play a strategy where $[\![\overline{q_m}]\!] - [\![\overline{p_m}]\!] = a-1$ but not $[\![\overline{s_m}]\!] = [\![\overline{t_m}]\!] = 0$. Second, he could play a strategy where $[\![\overline{q_m}]\!] - [\![\overline{p_m}]\!] \neq a-1$ and it is not the case that $[\![\overline{s_m}]\!] + [\![\overline{t_m}]\!] = a-1$, $[\![\overline{q_m}]\!] = [\![\overline{s_m}]\!]$, and $[\![\overline{p_m}]\!] + [\![\overline{t_m}]\!] = b-1$. However, with these strategies, and the assumption that Player Two is only selecting strategies with $[\![\overline{r_m}]\!] \leq b-1$, it is impossible to satisfy γ_1, meaning they would yield Player One a utility of zero in every profile. We can exclude these strategies as well.

As such, every strategy in the support of σ_1 is an interval over $[0, b-1]_{\mathbb{N}}$. We say the *cover weight* of a number i, c_i, is the sum of the weights Player One attaches to every interval containing i. That is,

$$c_i = P[(i \geq [\![\overline{p_m}]\!] \text{ and } i \leq [\![\overline{p_m}]\!] + a - 1) \text{ or } (i \leq [\![\overline{q_m}]\!] \text{ and } i \geq [\![\overline{q_m}]\!] - a + 1)].$$

In other words, the cover weight of i is the probability that Player Two will lose if she plays i with probability 1. If w_j is the weight Player One attaches to the interval starting at j then the cover weight of i, c_i, can be expressed as follows:

$$c_i = \sum_{j \equiv i-a+1 \mod b}^{i} w_j.$$

That is, the first w_j in the sum is the interval with i at its rightmost point, and hence its leftmost point is at $i - (a-1) \mod b$.

Now observe that a best response for Player Two to any strategy of Player One is to play $\arg\min_i c_i$ with probability 1, giving Player One a utility of $\min_i c_i$; Player One's maxmin strategy is thus to maximise the smallest of the cover weights. Furthermore, we can see that the sum of all the cover weights is invariant and equal to a (as the sum of all w_i is equal to 1 and each w_i appears in a distinct intervals). It follows that the only way to maximise the smallest cover weight is to make all c_i equal.

We will demonstrate that $c_i = c_{i+1}$ implies that $w_{i-a+1 \mod b} = w_{i+1}$. As a is a generating element in the additive group of b elements (this is where coprimality comes in), this will establish that all w_i must be the same.

The argument itself is trivial. Suppose $c_i = c_{i+1}$. If we expand this we have:

$$w_{i-a+1} + w_{i-a+2} + \cdots + w_i = w_{i-a+2} + w_{i-a+3} + \cdots + w_{i+1}.$$

By subtracting $(w_{i-a+2} + \cdots + w_i)$ from both sides we have that $w_{i-a+1} = w_{i+1}$.

We have thus shown that:

a) In every equilibrium Player One must make all the cover weights equal.
b) The only way to make all the cover weights equal is to play every interval with equal weight.

This settles the argument for Player One. We turn to Player Two.

This time we define the *breadth* of the interval starting at i, b_i, to be the sum of the weights, v_j, that Player Two attaches to every number in that interval:

$$b_i = \sum_{j=i}^{i+a-1 \mod b} v_j.$$

By an argument parallel to the one above, we see that Player Two seeks to choose a strategy that will make all b_i equal. Assuming $b_i = b_{i+1}$, this leads us to:

$$v_i + v_{i+1} + \cdots + v_{i+a-1} = v_{i+1} + v_{i+2} + \cdots + v_{i+a}.$$

This allows us to invoke the same argument.

In sum, we have shown that every equilibrium involves all c_i and b_i being equal, and the only way to achieve this is by randomising equally over all intervals and integers. This equilibrium is thus unique. □

We also introduce the following shorthand:

Definition 2.17. Let $G = (\Phi_1, \ldots, \Phi_n, \gamma_1, \ldots, \gamma_n)$ be a Boolean game. We use $\gamma_i(G)$ to refer to γ_i and $\mathrm{var}_i(G)$ to refer to Φ_i. This notation will be useful in the presence of multiple games, as it will allow us to distinguish between $\gamma_i(G)$ and $\gamma_i(G')$.

If G is a game or φ is a formula, we will also write $\mathrm{var}(G)$ and $\mathrm{var}(\varphi)$ to refer to the set of all variables present in the game or formula respectively. ■

3. Results

3.1. NEXP*-completeness of* ∃GuaranteeNash

Theorem 3.1. ∃GuaranteeNash *for two-player Boolean games is* NEXP*-complete.*

Proof. For membership in NEXP, expand the Boolean game into its normal form and run the NP algorithm.

For NEXP-hardness, we reduce from the bounded halting problem. That is, we will demonstrate that given a nondeterministic machine M, a binary input string w, and a computation bound K in binary, we can construct in polynomial time a two-player Boolean game G and a vector of payoffs $\boldsymbol{v}$ such that there exists an equilibrium $\boldsymbol{\sigma}$ of G where each player i attains a utility of at least $\boldsymbol{v}[i]$ if and only if M accepts w in at most K steps.

Without loss of generality, we assume two restrictions on M. First, M is augmented with a "do nothing" transition that is only activated from an accepting state. This ensures that every run of M on w is defined for all K computation steps, even if the machine stops computing early. Second, we assume that M has just one accepting state q_a, and if M accepts it accepts with the head over the leftmost cell and a 0 on the tape. This assumption is harmless as we can always extend M into an M' that simulates M until M reaches the accepting state, and then M' enters a routine requiring it to move to the left of the tape, print a 0, and then accept.

Recall that a transition rule of a Turing machine can be expressed as $(a, q) \to (b, D, q')$—"The machine, upon reading a in state q, writes b on the tape, transitions to state q' and moves the head in direction D". The variable D can be one of $\{ \mathit{Left}, \mathit{Right}, \mathit{Stop} \}$.

Let $k = |K|$. Observe that since K is given in binary, $2^k \geq K$. We can thus envisage a run of M on w as being represented by a $2^k \times 2^k$ table. Every row of the table is a machine configuration (that is, the tape contents, machine state and head location). The ith row, jth column of the table tells us the contents of the jth cell at the ith computation step; whether the head is over the cell, to

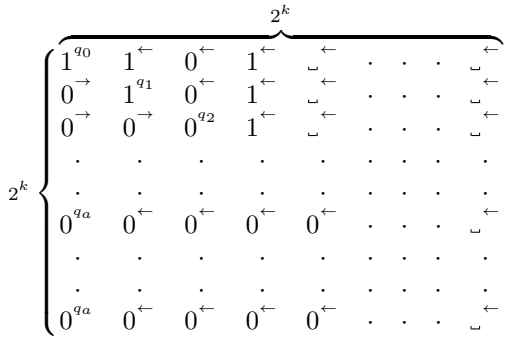

Figure 3.1: What a run of M on $w = 1101$ might look like. The row in the middle represents the step in which the machine accepts, after which it enters into a "do nothing" routine right through to step $2^k - 1$.

the left of it, or to the right; and, if the head is over the cell, the machine state. This is illustrated in Figure 3.1.

We formalise this in a definition.

Definition 3.2. A *history table* is a $2^k \times 2^k$ matrix with entries of the form a^b; $a \in \{ 1, 0, \sqcup \}$ and $b \in \{ \leftarrow, \rightarrow, q_i \in Q \}$.

Row i of T is *induced* by rule $(a, q) \to (b, D, q')$ of M just if there exists a j such that the $(i-1, j)$-entry of the history table is a^q, for $l < j$ the $(i-1, l)$-entry contains $\rightarrow$ in the superscript, for $l > j$ the $(i-1, l)$-entry contains $\leftarrow$ in the superscript, and:

1. If $D = \mathit{Left}$ and $j > 0$, then the (i, j)-entry is $b^{\leftarrow}$; letting c^d denote the $(i-1, j-1)$-entry, the $(i, j-1)$-entry is $c^{q'}$; for $l \neq j, j-1$, the (i, l)-entry is the same as the $(i-1, l)$-entry.
2. If $D = \mathit{Left}$ and $j = 0$, then the (i, j)-entry is $b^{q'}$, and for $l > j$ the (i, l)-entry is the same as the $(i-1, l)$-entry.
3. If $D = \mathit{Right}$, then the (i, j)-entry is $b^{\rightarrow}$; letting c^d denote the $(i-1, j+1)$-entry, the $(i, j+1)$-entry is $c^{q'}$; and for $l \neq j, j+1$, the (i, l)-entry is the same as the $(i-1, l)$-entry.
4. If $D = \mathit{Stop}$, then the (i, j)-entry is $b^{q'}$, and for $l \neq j$ the (i, l)-entry is the same as the $(i-1, l)$-entry.

Using w_i to denote the ith letter of w, a history table is *valid* just if:

1. The $(0, 0)$-entry is $w_0^{q_0}$.
2. For $0 < j < |w|$, the $(0, j)$-entry is $w_j^{\leftarrow}$.
3. For $j \geq |w|$, the $(0, j)$-entry is $\sqcup^{\leftarrow}$.
4. The $(K, 0)$ entry is 0^{q_a}.
5. For $i > 0$, row i of the history table is induced by some rule of the machine.

■

It is clear that M accepts w in at most K steps if and only if there exists a valid history table.

Consider the following variables for Player One:

$$\Phi_1' = \{\mathit{Zero}^1, \mathit{One}^1, \mathit{Left}^1, \mathit{Right}^1\} \cup \{\overline{\mathit{Time}_k^1}\} \cup \{\overline{\mathit{Tape}_k^1}\} \cup \{\overline{\mathit{State}_{|Q|}^1}\}.$$

With these variables, Player One is able to describe a single entry of T—$(\overline{[\![\mathit{Time}_k^1]\!]}, \overline{[\![\mathit{Tape}_k^1]\!]})$ is the index of the entry, the truth of variables Zero^1 and One^1 determines whether 0 or 1 appears in that entry, Left^1 and Right^1 determines whether $\leftarrow$ or $\rightarrow$ appears in the superscript, and $\overline{\mathit{State}_{|Q|}^1}$ the state of the machine. We use $\nu_1(T, i, j)$ to denote the truth assignment to Φ_1' that sets the truth of the variables in agreement with the (i, j)-entry of T. Not every assignment to Φ_1' is of the form $\nu_1(T, i, j)$ (e.g. the assignment that sets both One_1 and Zero_1 to *true*), but $\nu_1(T, i, j)$ is well defined for any choice of T and (i, j).

Let $\sigma_1(T)$ denote the probability distribution that realises $\nu_1(T, i, j)$ with probability $1/2^{2k}$ for every choice of (i, j).

A key observation, that we will show in Lemma 3.4, is that validity is a local property—if a history table is invalid, there is a 2×2 submatrix in the history table testifying to that fact.

Definition 3.3. A *local square* is a 2×2 matrix with entries of the form a^b; $a \in \{1, 0, \textvisiblespace\}$ and $b \in \{\leftarrow, \rightarrow, q_i \in Q\}$.

Given a history table T, we say that a local square S has (i, j) as its *upper left* just if S is the submatrix obtained by deleting all rows of T except $i, i \oplus 1$ and all columns except $j, j \oplus 1$.[2] Note that a consequence of this definition is that $(2^k, 2^k)$ is the upper left of the local square consisting of the $(2^k, 2^k), (2^k, 0), (0, 2^k), (0, 0)$-entries of T. ■

In our construction, Player Two's strategy will involve naming an index (i, j) and describing the local square with (i, j) as its upper left. To this end, we equip Player Two with the following variables:

$$\begin{aligned}\Phi_2' = \{&\mathit{Zero}^2, \mathit{One}^2, \mathit{sZero}^2, \mathit{sOne}^2, \mathit{nZero}^2, \mathit{nOne}^2, \mathit{nsZero}^2, \mathit{nsOne}^2,\\ &\mathit{Left}^2, \mathit{Right}^2, \mathit{sLeft}^2, \mathit{sRight}^2, \mathit{nLeft}^2, \mathit{nRight}^2, \mathit{nsLeft}^2, \mathit{nsRight}^2\}\\ \cup \{&\overline{\mathit{Time}_k^2}\} \cup \{\overline{\mathit{Tape}_k^2}\} \cup \{\overline{\mathit{State}_{|Q|}^2}\} \cup \{\overline{\mathit{sState}_{|Q|}^2}\} \cup \{\overline{\mathit{nState}_{|Q|}^2}\} \cup \{\overline{\mathit{nsState}_{|Q|}^2}\}.\end{aligned}$$

$(\overline{[\![\mathit{Time}_k^2]\!]}, \overline{[\![\mathit{Tape}_k^2]\!]})$ is the upper left of the local square. The truth of variables Zero^2 and One^2 determines whether 0 or 1 appears in entry $(\overline{[\![\mathit{Time}_k^2]\!]}, \overline{[\![\mathit{Tape}_k^2]\!]})$, Left^2 and Right^2 whether $\leftarrow$ or $\rightarrow$ appears in the superscript, and $\overline{\mathit{State}_{|Q|}^2}$ the state of the machine. The variables prefixed with s apply likewise to the $(\overline{[\![\mathit{Time}_k^2]\!]}, \overline{[\![\mathit{Tape}_k^2]\!]} \oplus 1)$-entry, with n the $(\overline{[\![\mathit{Time}_k^2]\!]} \oplus 1, \overline{[\![\mathit{Tape}_k^2]\!]})$-entry, and with

[2] Modular addition in this section is mod 2^k.

ns the $(\llbracket \overline{Time_k^2} \rrbracket \oplus 1, \llbracket \overline{Tape_k^2} \rrbracket \oplus 1)$-entry. Given a history table T, we write $\nu_2(T,i,j)$ to refer to the truth assignment that sets $\llbracket \overline{Time_k^2} \rrbracket = i$, $\llbracket \overline{Tape_k^2} \rrbracket = j$, and the other variables to the encoding of the local square with upper left (i,j). $\sigma_2(T)$ is the probability distribution that realises $\nu_2(T,i,j)$ with probability $1/2^{2k}$.

Lemma 3.4. *There exists a formula Valid*(Φ_2')*, of size polynomial in* $|M|$*, such that:*

1. *A history table T is invalid if and only if there exists an index (i,j) such that $\nu(T,i,j) \not\models Valid$.*
2. *For every ν that does not encode a local square, $\nu \not\models Valid$.*

Proof. We make two observations. First, the total number of possible local squares is polynomial in the size of M. This can be seen because an entry in the square is of the form a^b; $a \in \{1, 0, \llcorner\}$ and $b \in \{\leftarrow, \rightarrow, q_i \in Q\}$. There are 3 choices for a and $|Q|+2$ choices for b, giving $3(|Q|+2)$ possibilities. A local square has four entries, so the total number of possible squares is $(3|Q|+6)^4$. Accordingly, any subset of local squares is of polynomial size as well.

Second, given a fixed local square S and an index (i,j), we can construct a polynomial-size formula ψ_S such that $\nu(T,i,j) \vDash \psi_S$ if and only if the local square with upper left (i,j) is equal to S. This is done in the following manner:

$$\begin{pmatrix} 1^{\rightarrow} & \llcorner^{q_x} \\ 1^{q_y} & 0^{\leftarrow} \end{pmatrix}$$

$$\Downarrow$$

$$\begin{aligned} &One^2 \wedge \neg Zero^2 \wedge Right^2 \wedge \neg Left^2 \wedge \bigwedge_{q_l \in Q} \neg State_l^2 \\ \wedge &\neg sOne^2 \wedge \neg sZero^2 \wedge \neg sRight^2 \wedge \neg sLeft^2 \wedge sState_x^2 \wedge \bigwedge_{q_l \neq q_x} \neg sState_l^2 \\ \wedge &nOne^2 \wedge \neg nZero^2 \wedge \neg nRight^2 \wedge \neg nLeft^2 \wedge nState_y^2 \wedge \bigwedge_{q_l \neq q_y} \neg nState_l^2 \\ \wedge &\neg nsOne^2 \wedge nsZero^2 \wedge \neg nsRight^2 \wedge nsLeft^2 \wedge \bigwedge_{q_l \in Q} \neg nsState_l^2. \end{aligned}$$

Let X be the set of all local squares. Observe that $\nu \vDash \bigvee_{S \in X} \psi_S$ if and only if ν encodes a local square.

There are five requirements for validity in Definition 3.2, so we will show the existence of formulae V_1, V_2, V_3, V_4, V_5 with the property that the lth requirement is violated if and only if there exists an index (i,j) such that $\nu(T,i,j) \not\models V_l$. We then set $Valid = V_1 \wedge V_2 \wedge V_3 \wedge V_4 \wedge V_5 \wedge (\bigvee_{S \in X} \psi_S)$.

Requirement 1 is satisfied if and only if the local square with upper left $(0,0)$ contains $w_0^{q_0}$ in the upper left. Let X_0 be the set of all local squares satisfying this property. The required formula is:

$$V_1 = \big(\boldsymbol{Equal}(\overline{Time_k^2}; \ulcorner \mathbf{0} \urcorner) \wedge \boldsymbol{Equal}(\overline{Tape_k^2}; \ulcorner \mathbf{0} \urcorner)\big) \rightarrow (\bigvee_{S \in X_0} \psi_S).$$

Requirement 2 is satisfied if and only if for $0 < j < |w|$, the local square with upper left $(0, j)$ contains $w_j^{\leftarrow}$ in the upper left. Let X_j be the set of all local squares with this property. The required formula is:

$$V_2 = \bigvee_{0<j<|w|} \Big(\big(\boldsymbol{Equal}(\overline{Time_k^2}; \ulcorner \mathbf{0} \urcorner) \wedge \boldsymbol{Equal}(\overline{Tape_k^2}; \ulcorner \boldsymbol{j} \urcorner) \big) \rightarrow (\bigvee_{S \in X_j} \psi_S) \Big).$$

Requirement 3 is satisfied if and only if for $j \geq |w|$, the local square with upper left $(0, j)$ contains $\llcorner^{\leftarrow}$ in the upper left. Let X_e be the set of all local squares with this property. The required formula is:

$$V_3 = \big(\boldsymbol{LessEq}(\ulcorner |\boldsymbol{w}| \urcorner, \overline{Tape_k^2}) \wedge \boldsymbol{Equal}(\overline{Time_k^2}; \ulcorner \mathbf{0} \urcorner) \big) \rightarrow (\bigvee_{S \in X_e} \psi_S).$$

Requirement 4 is satisfied if and only if the local square with upper left $(K, 0)$ contains 0^{q_a} in the upper left. Let X_a be the set of all local squares satisfying this property. The required formula is:

$$V_4 = \big(\boldsymbol{Equal}(\overline{Time_k^2}; \ulcorner \boldsymbol{K} \urcorner) \wedge \boldsymbol{Equal}(\overline{Tape_k^2}; \ulcorner \mathbf{0} \urcorner) \big) \rightarrow (\bigvee_{S \in X_a} \psi_S).$$

Requirement 5 is satisfied if and only if for $i > 0$, row i of the history table is induced by some rule of the machine. Given an arbitrary rule $R = (a, q) \rightarrow (b, D, q')$, we will show the existence of a polynomial-size χ_R with the property that the ith row fails to be induced by rule R if and only if there exists an index j such that $\nu(T, i-1, j) \not\models \chi_R$. These formulae will have the further property that if i fails to be induced by *any* rule, then there exists an index j such that for every choice of R, $\nu(T, i-1, j) \not\models \chi_R$. The required formula will then be:

$$V_5 = \boldsymbol{Less}(\overline{Time_k^2}; \ulcorner \mathbf{2^k - 1} \urcorner) \rightarrow (\bigvee_{R \in M} \chi_R).$$

Fix an $R = (a, q) \rightarrow (b, D, q')$. We first address the requirement that there exist an l such that the $(i-1, l)$-entry of the history table is a^q, the entries to the left contain a $\rightarrow$ in the superscript, and the entries to the right a $\leftarrow$. Let X_l be the set of all local squares without a $\leftarrow$ in the upper left, X_r without a $\rightarrow$ in the upper left. Let X_m be the set of all local squares that contain either $(\rightarrow, \rightarrow)$. $(\rightarrow, q)$, $(q, \leftarrow)$ or $(\leftarrow, \leftarrow)$ in the superscripts of the upper row. Consider the formula:

$$\begin{aligned} \mathit{Prior} = & \big(\neg \boldsymbol{Equal}(\ulcorner \mathbf{2^k - 1} \urcorner; \overline{Tape_2^k}) \rightarrow (\bigvee_{S \in X_m} \psi_S) \big) \\ & \wedge \big(\boldsymbol{Equal}(\overline{Tape_k^2}; \ulcorner \mathbf{0} \urcorner) \rightarrow (\bigvee_{S \in X_l} \psi_S) \big) \\ & \wedge \big(\boldsymbol{Equal}(\overline{Tape_k^2}; \ulcorner \mathbf{2^k - 1} \urcorner) \rightarrow (\bigvee_{S \in X_r} \psi_S) \big). \end{aligned}$$

Suppose $\nu(T, i-1, j) \vDash \mathit{Prior}$ for all choices of j. This can only be the case if the $(i-1, 0)$-entry contains either q or $\rightarrow$ in the superscript, the $(i-1, 2^k-1)$-entry contains either q or $\leftarrow$ in the superscript, and all intervening entries contain arrows pointing in the direction of the entry containing q; such an entry must exist because X_m does not contain any squares with $(\rightarrow, \leftarrow)$ in the upper row. As such this can only be the case if row $i-1$ indeed contains the head in exactly one position, and the arrows point in the direction of the head.

Suppose $D = \mathit{Left}$. If the $(i-1, j)$-entry of T contains a^q, and $j > 0$, then the (i, j)-entry must contain $b^{\leftarrow}$; if the $(i-1, j-1)$-entry contains c^d, then the $(i, j-1)$ entry must contain $c^{q'}$; and for every $l \neq j, j-1$ the (i, l)-entry must be the same as the $(i-1, l)$-entry. Let X_m be the set of all local squares conforming to one of the following patterns:

$$\begin{pmatrix} c^d & a^q \\ c^{q'} & b^{\leftarrow} \end{pmatrix} \quad \begin{pmatrix} a^q & c^d \\ b^{\leftarrow} & c^d \end{pmatrix} \quad \begin{pmatrix} c^d & c'^{d'} \\ c^d & c'^{q'} \end{pmatrix} \quad \begin{pmatrix} c^d & c'^{d'} \\ c^d & c'^{d'} \end{pmatrix}$$

If the $(i-1, 0)$-entry of T contains a^q, then the $(i, 0)$-entry must contain $b^{q'}$. Let X_b be the set of all local squares conforming to one of the following patterns:

$$\begin{pmatrix} c^d & a^q \\ c^{q'} & b^{\leftarrow} \end{pmatrix} \quad \begin{pmatrix} a^q & c^d \\ b^{q'} & c^d \end{pmatrix} \quad \begin{pmatrix} c^d & c'^{d'} \\ c^d & c'^{q'} \end{pmatrix} \quad \begin{pmatrix} c^d & c'^{d'} \\ c^d & c'^{d'} \end{pmatrix}$$

The required formula is:

$$\mathit{Post} = \big(\boldsymbol{Equal}(\overline{\mathit{Tape}^2_k}; \ulcorner \mathbf{0} \urcorner) \rightarrow (\bigvee_{S \in X_b} \psi_S)\big)$$
$$\Big(\big((\neg \boldsymbol{Equal}(\overline{\mathit{Tape}^2_k}; \ulcorner \mathbf{0} \urcorner) \wedge \neg \boldsymbol{Equal}(\overline{\mathit{Tape}^2_k}; \ulcorner \mathbf{2^k - 1} \urcorner)\big) \rightarrow (\bigvee_{S \in X_m} \psi_S)\Big).$$

Observe that if $\nu(T, i-1, j) \vDash \mathit{Prior}$ for all choices of j, then row i of T must be the same as row $i-1$, except for the entries in vicinity to the head. Combined with Post, this ensures that row i is induced by R. This gives us:

$$\chi_R = \mathit{Prior} \wedge \mathit{Post}.$$

The cases $D \in \{ \mathit{Right}, \mathit{Stop} \}$ are handled analogously.

It remains to show that if i fails to be induced by any R, then there exists a j such that $\nu(T, i-1, j) \nvDash \chi_R$ for all choices of R.

Observe that if row $i-1$ does not have a unique head with arrows pointing to it, the Prior formula of every χ_R would be falsified, likewise if the head were in a state for which no transition rules exist. We can thus assume that row $i-1$ has the head in state q at index j. If none of the entries $(i, j-1), (i, j), (i, j+1)$ contain the head then the Post formula of every χ_R would be falsified. If both the $(i, j-1)$ and (i, j)-entry contain the head, then the local square with upper left $(i-1, j-1)$ would falsify every χ_R; likewise if both the (i, j) and $(i, j+1)$-entry contain the head, then the local square with upper left $(i-1, j)$ would

falsify every χ_R. If both the $(i, j-1)$ and $(i, j+1)$-entry contain the head, then the arrow in (i, j) must point either left or right, and the local square with upper left $(i-1, j)$ or $(i-1, j-1)$ would falsify every χ_R. Suppose the head is in exactly one of these entries, without loss of generality $(i, j+1)$. Let a^q be the $(i-1, j)$-entry and $b^{q'}$ the $(i, j+1)$-entry. If $R' = (a, q) \to (b, q', \mathit{Right})$ is not a rule of the machine, then the local square with upper left $(i-1, j)$ would falsify every χ_R. If R' is a rule, but there exists a $\nu(T, i-1, j) \not\models \chi_{R'}$, then that must mean the local square with upper left $(i-1, j)$ fails to correctly copy the contents of row $i-1$ to row i, which would falsify every χ_R. □

The game will consist of Player One describing a history table and Player Two verifying that the table is valid. To ensure that the two players are describing the same history table, we introduce the following formula, which is true if Player One names an index within Player Two's local square, and the descriptions of that entry by both players agree:

$$
\begin{aligned}
\mathit{Agree} = \Big(& \big(\boldsymbol{Equal}(\overline{\mathit{Tape}^1_k}; \overline{\mathit{Tape}^2_k}) \wedge \boldsymbol{Equal}(\overline{\mathit{Time}^1_k}; \overline{\mathit{Time}^2_k}) \big) \to \\
& \big((\mathit{Zero}^1 \leftrightarrow \mathit{Zero}^2) \wedge (\mathit{One}^1 \leftrightarrow \mathit{One}^2) \wedge (\mathit{Left}^1 \leftrightarrow \mathit{Left}^2) \\
& \wedge (\mathit{Right}^1 \leftrightarrow \mathit{Right}^2) \wedge \bigwedge_{1 \le i \le |Q|} (\mathit{State}^1_i \leftrightarrow \mathit{State}^2_i) \big) \Big) \\
\wedge \Big(& \big(\boldsymbol{AddMod}(\overline{\mathit{Tape}^2_k}; \ulcorner \mathbf{1} \urcorner; \overline{\mathit{Tape}^1_k}) \wedge \boldsymbol{Equal}(\overline{\mathit{Time}^1_k}; \overline{\mathit{Time}^2_k}) \big) \to \\
& \big((\mathit{Zero}^1 \leftrightarrow \mathit{sZero}^2) \wedge (\mathit{One}^1 \leftrightarrow \mathit{sOne}^2) \wedge (\mathit{Left}^1 \leftrightarrow \mathit{sLeft}^2) \\
& \wedge (\mathit{Right}^1 \leftrightarrow \mathit{sRight}^2) \wedge \bigwedge_{1 \le i \le |Q|} (\mathit{State}^1_i \leftrightarrow \mathit{sState}^2_i) \big) \Big) \\
\wedge \Big(& \big(\boldsymbol{Equal}(\overline{\mathit{Tape}^1_k}; \overline{\mathit{Tape}^2_k}) \wedge \boldsymbol{AddMod}(\overline{\mathit{Time}^2_k}; \ulcorner \mathbf{1} \urcorner; \overline{\mathit{Time}^1_k}) \big) \to \\
& \big((\mathit{Zero}^1 \leftrightarrow \mathit{nZero}^2) \wedge (\mathit{One}^1 \leftrightarrow \mathit{nOne}^2) \wedge (\mathit{Left}^1 \leftrightarrow \mathit{nLeft}^2) \\
& \wedge (\mathit{Right}^1 \leftrightarrow \mathit{nRight}^2) \wedge \bigwedge_{1 \le i \le |Q|} (\mathit{State}^1_i \leftrightarrow \mathit{nState}^2_i) \big) \Big) \\
\wedge \Big(& \big(\boldsymbol{AddMod}(\overline{\mathit{Tape}^2_k}; \ulcorner \mathbf{1} \urcorner; \overline{\mathit{Tape}^1_k}) \wedge \boldsymbol{AddMod}(\overline{\mathit{Time}^2_k}; \ulcorner \mathbf{1} \urcorner; \overline{\mathit{Time}^1_k}) \big) \to \\
& \big((\mathit{Zero}^1 \leftrightarrow \mathit{nsZero}^2) \wedge (\mathit{One}^1 \leftrightarrow \mathit{nsOne}^2) \wedge (\mathit{Left}^1 \leftrightarrow \mathit{nsLeft}^2) \\
& \wedge (\mathit{Right}^1 \leftrightarrow \mathit{nsRight}^2) \wedge \bigwedge_{1 \le i \le |Q|} (\mathit{State}^1_i \leftrightarrow \mathit{nsState}^2_i) \big) \Big).
\end{aligned}
$$

Observe that under $(\sigma_1(T), \sigma_2(T))$ the formula $\mathit{Agree} \wedge \mathit{Valid}$ is satisfied with probability 1 if and only if T is valid. However, as of yet the players have no incentive of playing $(\sigma_1(T), \sigma_2(T))$. We define the following formulae, which are true if Player One names an index outside of the local square named by Player

Two:

$$\begin{aligned}
\mathit{AvoidC} =& \neg\big(\boldsymbol{Equal}(\overline{\mathit{Tape}_k^1}; \overline{\mathit{Tape}_k^2}) \wedge \boldsymbol{Equal}(\overline{\mathit{Time}_k^1}; \overline{\mathit{Time}_k^2})\big),\\
\mathit{Avoid} =& \neg\big(\boldsymbol{AddMod}(\overline{\mathit{Tape}_k^2}; \ulcorner \mathbf{1} \urcorner; \overline{\mathit{Tape}_k^1}) \wedge \boldsymbol{AddMod}(\overline{\mathit{Time}_k^2}; \ulcorner \mathbf{1} \urcorner; \overline{\mathit{Time}_k^1})\big)\\
& \wedge \neg\big(\boldsymbol{AddMod}(\overline{\mathit{Tape}_k^2}; \ulcorner \mathbf{1} \urcorner; \overline{\mathit{Tape}_k^1}) \wedge \boldsymbol{Equal}(\overline{\mathit{Time}_k^1}; \overline{\mathit{Time}_k^2})\big)\\
& \wedge \neg\big(\boldsymbol{Equal}(\overline{\mathit{Tape}_k^1}; \overline{\mathit{Tape}_k^2}) \wedge \boldsymbol{AddMod}(\overline{\mathit{Time}_k^2}; \ulcorner \mathbf{1} \urcorner; \overline{\mathit{Time}_k^1})\big).
\end{aligned}$$

This gives us the complete description of the game:

$$\begin{aligned}
\Phi_1 =& \Phi_1' \cup \mathrm{var}_1 \mathfrak{G}(3/4),\\
\Phi_2 =& \Phi_2' \cup \mathrm{var}_2 \mathfrak{G}(3/4),\\
\gamma_1 =& \mathit{AvoidC} \wedge (\mathit{Avoid} \vee \gamma_1(\mathfrak{G}(3/4)),\\
\gamma_2 =& \Big(\neg \mathit{AvoidC} \vee \big(\neg \mathit{Avoid} \wedge \gamma_2(\mathfrak{G}(3/4))\big)\Big) \wedge \mathit{Agree} \wedge \mathit{Valid}.
\end{aligned}$$

We now claim that the existence of an accepting run of M on w in at most K steps is equivalent to the existence of an equilibrium in the game described where Player Two is guaranteed utility:

$$\boldsymbol{v}[2] = \frac{1}{2^{2k}} + \frac{3}{2^{2k} \cdot 4}.$$

Suppose an accepting run exists. Let T be a valid history table. Let $\sigma_i(\mathfrak{G}(3/4))$ represent Player i's equilibrium strategy in $\mathfrak{G}(3/4)$. Consider the profile where Player One plays $\sigma_1(T)\sigma_1(\mathfrak{G}(3/4))$, and Player Two plays $\sigma_2(T)\sigma_2(\mathfrak{G}(3/4))$. Call this profile $\boldsymbol{\sigma}$.

There is a $1/2^{2k}$ chance of the players playing picking the same index, i.e. the truth assignments $(\nu_1(T,i,j), \nu_2(T,i,j))$. In this case $\neg \mathit{AvoidC}$, *Agree*, and *Valid* are satisfied. This would satisfy γ_2.

On top of this, there is a $3/2^{2k}$ chance of the players picking one of the following: $\{(\nu_1(T, i \oplus 1, j), \nu_2(T,i,j)), (\nu_1(T,i,j \oplus 1), \nu_2(T,i,j), (\nu_1(T, i \oplus 1, j \oplus 1), \nu_2(T,i,j))\}$. In this case, $\neg \mathit{Avoid}$, *Agree* and *Valid* are satisfied, but $\gamma_2(\mathfrak{G}(3/4))$ has only a $1/4$ chance of being satisfied. For any other $(\nu_1(T,i,j), \nu_2(T,i',j'))$, neither $\neg \mathit{Avoid}$ nor $\neg \mathit{AvoidC}$ is satisfied. Altogether, gives Player Two a $1/2^{2k}$ chance of satisfying γ_2 via $\neg \mathit{AvoidC}$, and $3 \cdot \frac{1}{2^{2k} \cdot 4}$ via $\neg \mathit{Avoid}$. It remains to see that this is an equilibrium.

Suppose Player Two attempts to deviate with a pure strategy. Note that since $\gamma_2 = \Big(\neg \mathit{AvoidC} \vee \big(\neg \mathit{Avoid} \wedge \gamma_2(\mathfrak{G}(3/4))\big)\Big) \wedge \gamma_2'$, the probability of satisfying $\neg \mathit{AvoidC} \vee \big(\neg \mathit{Avoid} \wedge \gamma_2(\mathfrak{G}(3/4))\big)$ is an upper bound on her utility. Given that Player One is playing $\sigma_1(T)\sigma_1(\mathfrak{G}(3/4)$, whatever her assignment to $\overline{\mathit{Time}_k^2}$, $\overline{\mathit{Time}_k^2}$, she will have a $1/2^{2k}$ chance of satisfying $\neg \mathit{AvoidC}$, and a $3/2^{2k} \cdot 4$ chance of satisfying $\neg \mathit{Avoid} \wedge \gamma_2(\mathfrak{G}(3/4))$. Thus she can never obtain a utility superior to $\boldsymbol{v}[2]$.

In $\boldsymbol{\sigma}$ Player One earns $1-\boldsymbol{v}[2]$ utility as he satisfies γ_1 precisely when Player Two fails to satisfy $\neg \mathit{AvoidC} \vee \big(\neg \mathit{Avoid} \wedge \gamma_2(\mathfrak{G}(3/4))\big)$. Suppose he deviates with a pure strategy Whatever his assignment to $\overline{\mathit{Time}_k^1}, \overline{\mathit{Time}_k^1}$, he will have a $1/2^{2k}$ chance of satisfying AvoidC and a $3/2^{2k}$ chance of Avoid, in which case he has a $3/4$ chance of winning $\mathfrak{G}(3/4)$. His payoff is thus still $1-\boldsymbol{v}[2]$.

Now, suppose that no accepting run exists. We have already observed that Player Two's utility is bounded above by the probability of satisfying $\neg \mathit{AvoidC} \vee \big(\neg \mathit{Avoid} \wedge \gamma_2(\mathfrak{G}(3/4))\big)$. We will show that there is exactly one way to satisfy this with probability at least $\boldsymbol{v}[2]$, which will allow us to reduce the number of equilibria we need to consider—any equilibrium that does not offer a $\boldsymbol{v}[2]$ chance of satisfying $\neg \mathit{AvoidC} \vee \big(\neg \mathit{Avoid} \wedge \gamma_2(\mathfrak{G}(3/4))\big)$ would yield Player Two less than $\boldsymbol{v}[2]$ utility.

Let $x_{i,j}$ be the probability with which Player Two plays $(\llbracket \overline{\mathit{Time}_2^k} \rrbracket = i$ and $\llbracket \overline{\mathit{Tape}_2^k} \rrbracket = j$. Note that Player One's payoff for playing $(\llbracket \overline{\mathit{Time}_1^k} \rrbracket = i, \llbracket \overline{\mathit{Tape}_1^k} \rrbracket = j)$ with probability 1 is $1 - c_{i,j}$, defining $c_{i,j}$ to be:

$$c_{i,j} = x_{i,j} + \frac{x_{i\ominus 1,j}}{4} + \frac{x_{i,j\ominus 1}}{4} + \frac{x_{i\ominus 1,j\ominus 1}}{4}.$$

As such, a necessary condition for equilibrium is that Player Two chooses a strategy where all $c_{i,j}$ are equal, else Player One could play $\arg\min_{i,j} c_{i,j}$ with probability 1. To maximise her utility, Player Two must maximise the value of $c_{i,j}$. As can be seen from the sum below, this involves setting $c_{i,j} = \boldsymbol{v}[2]$:

$$\sum_{i,j<2^k} c_{i,j} = 1 + \frac{3}{4} = \frac{2^{2k}}{2^{2k}} + \frac{2^{2k}\cdot 3}{2^{2k}\cdot 4} = 2^{2k}\boldsymbol{v}[2].$$

Thus we only need to consider equilibria where the probabilities $x_{i,j}$ form a solution to the linear system $c_{i,j} = \boldsymbol{v}[2]$.

Let A be a $2^{2k} \times 2^{2k}$ matrix of coefficients with row $i + 2^k j$ corresponding to the equation $c_{i,j}$ and column $i + 2^k j$ to the variable $x_{i,j}$. Observe that every row in this matrix contains a 1 on the diagonal and 3 $1/4$'s elsewhere in the row. Define a disc for each row with centre $A[i,i]$ and radius $\sum_{j\neq i} |A[i,j]|$. By Gerschgorin's circle theorem (Gerschgorin, 1931), every eigenvalue of the matrix lies in at least one such disc. Since none of the discs cover 0, the matrix is non-singular, and the system $c_{i,j} = \boldsymbol{v}[2]$ has a unique solution—namely, the solution that involves Player Two playing $x_{i,j} = 1/2^{2k}$. We can thus restrict ourselves to equilibria of this form, as no other equilibria can give Player Two a utility of at least $\boldsymbol{v}[2]$.

If Player One is playing all (i,j) with non-zero probability, then by Lemma 3.4 there must exist some (i,j) that does not satisfy *Valid*, and hence Player Two cannot attain $\boldsymbol{v}[2]$ utility. Suppose Player One attaches zero weight to some entries, but all the other entries satisfy *Valid*. Let (i,j) be an entry to which Player One attaches zero weight, and (i',j') the entry to which Player One attaches the highest weight. As Player Two is randomising equally, she must attach the same weight to the local square with upper left (i,j), as to the local

square with upper left corner (i', j'). This profile cannot be in equilibrium: if x is the probability Player One attaches to (i', j') then by transferring the weight Player Two currently attaches to the local square with upper left (i, j) to the local square with upper left (i', j') she will lose at most $x \cdot \frac{3}{2^{2k} \cdot 4}$ utility, and gain at least $x \cdot \frac{1}{2^{2k}}$ utility—which would be a profitable deviation. □

3.2. Other decision problems

Having established that ∃GuaranteeNash is NEXP-complete, we can return to more familiar game-theoretic arguments for the other problems. The difficulty lies in expressing the constructions in the restricted vocabulary of Boolean games, which requires a bit of coding.

Theorem 3.5. UniqueNash *and* ∀NashSat *are* coNEXP*-complete for two-player Boolean games .* ∃NashSat *is* NEXP*-complete.*

Proof. For ∃NashSat, we reduce from ∃GuaranteeNash, while for UniqueNash and ∀NashSat we reduce from the complement of ∃GuaranteeNash. The idea is, given an instance $(G, \boldsymbol{v})$ of ∃GuaranteeNash, we construct a game where the players can either play in G or can unilaterally deviate to a game where they get a utility of $\boldsymbol{v}[1](1-\boldsymbol{v}[2])$ or $\boldsymbol{v}[2](1-\boldsymbol{v}[1])$ in a unique equilibrium. The player who does not deviate gets nothing, so there is no equilibrium where one player chooses G and the other to switch. Hence if ∃GuaranteeNash does not have an equilibrium guaranteeing the players at least $\boldsymbol{v}$ utility, there will be a unique equilibrium where both players deviate. To extend the proof to ∀NashSat, we will show that there exists a formula φ that is true in the unique equilibrium with probability one, and if the equilibrium is not unique then there exists an equilibrium in which φ is false with non-zero probability. Likewise, for ∃NashSat, we show there exists a ψ that is false in the unique equilibrium, but if the equilibrium is not unique then there is some equilibrium under which ψ is true with a probability of one.

Let G be a Boolean game and $\boldsymbol{v} \in [0,1]^2_{\mathbb{Q}}$. Let $u = \boldsymbol{v}[1]$ and $w = 1 - \boldsymbol{v}[2]$. Construct $\mathfrak{G}(u)$ and $\mathfrak{G}(w)$ over fresh variables. Recall, from Lemma 2.16, that $\mathfrak{G}(u)$ and $\mathfrak{G}(w)$ each have a unique equilibrium. Introduce new *Play* and *Dummy* variables for both players.

We claim that G' with the following parameters has a unique Nash equilibrium if and only if $(G, \boldsymbol{v}) \notin$ ∃GuaranteeNash:

$$
\begin{aligned}
\Phi_1 =& \mathrm{var}_1(G) \cup \mathrm{var}_1(\mathfrak{G}(u)) \cup \mathrm{var}_1(\mathfrak{G}(w)) \cup \{Play^1, Dummy^1\},\\
\Phi_2 =& \mathrm{var}_2(G) \cup \mathrm{var}_2(\mathfrak{G}(u)) \cup \mathrm{var}_2(\mathfrak{G}(w)) \cup \{Play^2, Dummy^2\},\\
\gamma_1 =& \gamma_1(\mathfrak{G}(w)) \wedge \Big((\neg Play^1 \wedge \neg Play^2 \wedge \gamma_1(G))\\
&\quad \vee (Play^1 \wedge \neg Dummy^1 \wedge (\bigwedge_{p \in \mathrm{var}_1(G)} \neg p) \wedge \gamma_1(\mathfrak{G}(u)))\Big),\\
\gamma_2 =& \gamma_2(\mathfrak{G}(u)) \wedge \Big((\neg Play^1 \wedge \neg Play^2 \wedge \gamma_2(G))\\
&\quad \vee (Play^2 \wedge \neg Dummy^2 \wedge (\bigwedge_{p \in \mathrm{var}_2(G)} \neg p) \wedge \gamma_2(\mathfrak{G}(w)))\Big).
\end{aligned}
$$

The formulae φ and ψ are the following:

$$
\begin{aligned}
\varphi &= Play_1 \wedge Play_2,\\
\psi &= \neg Play_1 \wedge \neg Play_.
\end{aligned}
$$

First, suppose that $(G, \boldsymbol{v}) \in \exists$GuaranteeNash, and $\boldsymbol{\sigma}$ is an equilibrium that satisfies the payoff criterion. Consider any profile $\boldsymbol{\sigma'}$ in G' in which:

1. $\boldsymbol{\sigma'}|G = \boldsymbol{\sigma}$.
2. $P(Play^1 = true) = 0$, $P(Play^2 = true) = 0$.
3. $\boldsymbol{\sigma'}|\mathfrak{G}(u)$ and $\boldsymbol{\sigma'}|\mathfrak{G}(w)$ are the unique equilibria of $\mathfrak{G}(u)$ and $\mathfrak{G}(w)$ respectively.

There are uncountably many such profiles, as the distribution over the dummy variables is not specified. We will show that every profile of this form is an equilibrium, and hence the equilibrium is not unique.

Observe that the utility of Player One under $\boldsymbol{\sigma'}$ is $w\boldsymbol{v}[1]$—a probability of $\boldsymbol{v}[1]$ of satisfying $\gamma_1(G)$ and a probability of w of satisfying $\gamma_1(\mathfrak{G}(w))$. Likewise, the utility of Player Two is $(1-u)\boldsymbol{v}[2]$. We will show that no deviation can improve a player's utility.

In order for Player One to satisfy γ_1, he must satisfy $\gamma_1(\mathfrak{G}(w))$ and one of $(\neg Play^1 \wedge \neg Play^2 \wedge \gamma_1(G))$ or $(Play^1 \wedge \neg Dummy^1 \wedge (\bigwedge_{p \in \mathrm{var}_1(G)} \neg p) \wedge \gamma_1(\mathfrak{G}(u)))$. These two disjuncts are mutually exclusive, so we only need to consider Player One's probability of satisfying one or the other. Given that $\boldsymbol{\sigma'_2}|\mathfrak{G}(w)$ is the equilibrium strategy, Player One's probability of satisfying $\gamma_1(\mathfrak{G}(w))$ is always w. The probability of satisfying $(\neg Play^1 \wedge \neg Play^2 \wedge \gamma_1(G))$ is at most $\boldsymbol{v}[1]$, else Player One would have a profitable deviation from $\boldsymbol{\sigma}$ in G, which we assumed to be an equilibrium. The probability of satisfying $(Play^1 \wedge \neg Dummy^1 \wedge (\bigwedge_{p \in \mathrm{var}_1(G)} \neg p) \wedge \gamma_1(\mathfrak{G}(u)))$ is the probability of satisfying $\gamma_1(\mathfrak{G}(u))$ which, given $\boldsymbol{\sigma'_2}|\mathfrak{G}(u)$ being the equilibrium strategy, is u. As $u = \boldsymbol{v}[1]$, Player One's utility in either case is bounded above by $w\boldsymbol{v}[1]$. Mutatis mutandis, for Player Two.

This establishes that every $\boldsymbol{\sigma'}$ is an equilibrium. This means that the equilibrium is not unique, so $G \notin$ UniqueNash, and since $P(Play^1 = true) = 0$

and $P(Play^2 = true) = 0$, φ is always false under $\boldsymbol{\sigma'}$ and hence $G \notin \forall$NashSat. Conversely, since ψ is true with probability one under $\boldsymbol{\sigma'}$, $G \notin \exists$NashSat. Now suppose that $(G, \boldsymbol{v}) \notin \exists$GuaranteeNash. We claim that the only equilibrium of G' involves setting the variables in G as well as $Dummy^1, Dummy^2$ to *false*, $Play^1, Play^2$ to *true* and playing the unique equilibria of $\mathfrak{G}(u)$ and $\mathfrak{G}(w)$ over the remaining variables.

First note that this is indeed an equilibrium—Player One's probability of satisfying $\gamma_1(\mathfrak{G}(w))$ is always w, his probability of satisfying $\gamma_1(\mathfrak{G}(u))$ is always u, and given that Player Two's strategy realises $Play^2 = true$ with probability one, Player One's probability of satisfying $(\neg Play^1 \wedge \neg Play^2 \wedge \gamma_1(G))$ is zero.

Now consider any other profile $\boldsymbol{\sigma'}$. Note that if the profile realises $\neg Play^1 \wedge \neg Play^2$ with non-zero probability, then $\boldsymbol{\sigma'}|G$ must be an equilibrium of G, else one of the players could increase their probability of satisfying $(\neg Play^1 \wedge \neg Play^2 \wedge \gamma_1(G))$ or $(\neg Play^1 \wedge \neg Play^2 \wedge \gamma_2(G))$ respectively. However this leads directly to a contradiction. Since $(G, \boldsymbol{v}) \notin \exists$GuaranteeNash, at least one of the players will satisfy $\gamma_i(G)$ under $\boldsymbol{\sigma'}|G$ with probability strictly less than $\boldsymbol{v}[i]$. Without loss of generality, let it be Player One. In this case Player One has a deviation available to him: by setting $Play^1$ to *true*, $Dummy^1$ and all variables in G to *false* he can transfer the weight he is currently assigning to $(\neg Play^1 \wedge \neg Play^2 \wedge \gamma_1(G))$, which yields him less than $\boldsymbol{v}[1]$ utility, to $(Play^1 \wedge \neg Dummy^1 \wedge (\bigwedge_{p \in \mathrm{var}_1(G)} \neg p) \wedge \gamma_1(\mathfrak{G}(u)))$, where he can guarantee himself at least $u = \boldsymbol{v}[1]$ by playing the equilibrium play of $\mathfrak{G}(u)$. This does not affect the probability of him satisfying $\gamma_1(\mathfrak{G}(w))$ and hence increases the probability of satisfying γ_1.

Next suppose that $\neg Play^1 \wedge \neg Play^2$ is realised with probability 0. We can safely assume that the players set $Play^1, Play^2$ to *true* with probability 1 and $\mathrm{var}_1(G), \mathrm{var}_2(G), Dummy^1, Dummy^2$ with probability 0, as anything else would lose them the game. The remaining variables are $\mathrm{var}_1(\mathfrak{G}(u)), \mathrm{var}_2(\mathfrak{G}(u))$ and $\mathrm{var}_1(\mathfrak{G}(w)), \mathrm{var}_2(\mathfrak{G}(w))$, and there is a unique way to play those in equilibrium. But that would means $\boldsymbol{\sigma'}$ is precisely the profile we started with initially, and hence the equilibrium must be unique. Since the equilibrium involves $Play_1$ and $Play_2$ being realised with probability one, φ is satisfied in the unique, and hence in every, equilibrium of G. On the contrary, ψ is false in the unique equilibrium, and hence there exists an equilibrium under which ψ is not true with a probability of one. □

4. Conclusion

In this paper we have capitalised on the advances made into the study of mixed equilibria in Boolean games in Ianovski and Ong (2014) by strengthening, expanding and, hopefully, simplifying the proof techniques used. While this does not exactly allow us to obtain results "for free", it does provide us with a tool kit and a strategy, whereas before we had but wishful thinking. In an upcoming paper we demonstrate how these techniques can be used to obtain highly non-trivial results with apparent ease, and to translate proofs from other frameworks into the language of Boolean games.

What this paper does not provide is any way to deal with function and search problems—notably FindNash, the problem of actually computing an equilibrium. While it is possible that the coding machinery introduced here will be helpful in addressing this challenge, it certainly does not offer any trivial way to translate the argument of Daskalakis and Papadimitriou (2005) or others into our setting. This question should be of interest even to researchers outside of the Boolean games community as it will concern function problems in super-polynomial time—a class rarely studied and little understood. This is clearly an avenue that bears further enquiry.

Acknowledgements

Egor Ianovski is supported by a scholarship from the Oxford-Man Institute of Quantitative Finance.

References

Bonzon, E., Lagasquie-Schiex, M.-C., Lang, J., Zanuttini, B., 2006. Boolean games revisited. In: Proceedings of ECAI 2006. pp. 265–269.

Daskalakis, C., Papadimitriou, C. H., 2005. Three-player games are hard. Tech. Rep. TR05-139, ECCC.

Dunne, P. E., van der Hoek, W., 2004. Representation and complexity in Boolean games. In: Alferes, J. J., Leite, J. a. (Eds.), Logics in Artificial Intelligence. Vol. 3229 of Lecture Notes in Computer Science. Springer-Verlag, pp. 347–359.

Dunne, P. E., Wooldridge, M., 2012. Towards tractable Boolean games. In: Proceedings of AAMAS '12. pp. 939–946.

Fortnow, L., Kabanets, V., Impagliazzo, R., Umans, C., 2005. On the complexity of succinct zero-sum games. In: IEEE Conference on Computational Complexity. pp. 323–332.

Gerschgorin, S., 1931. Über die Abgrenzung der Eigenwerte einer Matrix. Bulletin de l'Académie des Sciences de l'URSS. Classe des sciences mathématiques et naturelles (6), 749–754.

Gilboa, I., Zemel, E., 1989. Nash and correlated equilibria: Some complexity considerations. Games and Economic Behavior 1 (1), 80–93.

Harrenstein, P., van der Hoek, W., Meyer, J.-J., Witteveen, C., 2001. Boolean games. In: Proceedings of TARK 2001. pp. 287–298.

Ianovski, E., Ong, L., 2014. EGuaranteeNash for Boolean games is NEXP-hard. In: Principles of Knowledge Representation and Reasoning: Proceedings of the Fourteenth International Conference, KR 2014, Vienna, Austria, July 20-24, 2014.
URL `http://www.aaai.org/ocs/index.php/KR/KR14/paper/view/8002`

Nash, J., 1951. Non-cooperative games. Annals of Mathematics 54 (2), 286–295.

Schoenebeck, G. R., Vadhan, S., 2012. The computational complexity of Nash equilibria in concisely represented games. ACM Trans. Comput. Theory 4 (2), 4:1–4:50.

von Neumann, J., 1928. Zur Theorie der Gesellschaftsspiele. Mathematische Annalen 100 (1), 295–320.