reading (α_k, β_k). By previous arguments, we know that T_W will not get stuck and reject along p_P. Once T_W enters q_A it stays in q_A, an accepting state. Therefore T_W accepts the path $p = p_P \cdot p'$

(2) The second case is when p can be factored as $p_P \cdot p_j \cdot p'$, with p_P finite but possibly empty and p_j finite and nonempty. For every label-direction pair (α_i, β_i) in p_P we have that $\alpha_i = \beta_i$. For some $j \in \Omega \setminus W$ we have that $\alpha_i[-j] = \beta_i[-j]$ for every label-direction pair (α_i, β_i) in p_j, again noting that only one choice of j is appropriate. Finally, at the first point (α_k, β_k) of p' we have that $\alpha_k[-j] \neq \beta_k[-j]$. By previous arguments, we know that T_W will not get stuck and reject along p_P or p_j. And since T_W transitions to q_A at the beginning of p', we know that it cannot get stuck and reject along p'. Therefore T_W will accept on $p = p_P \cdot p_j \cdot p'$.

□

Corollary 3.5. *Let G be an iBG and $W \subseteq \Omega$ be a set of agents. Then, a W-NE strategy exists in G iff the automaton T_W constructed with respect to G is nonempty.*

4 ALGORITHMIC FRAMEWORK

In the previous section, we constructed an automaton T_W that recognizes the set of Nash equilibrium strategy profiles with winning set W in an iBG G, which we denoted as W-NE strategies. The problem of determining whether a W-NE strategy exists is equivalent to testing T_W for nonemptiness. The standard algorithm for testing nonemptiness of Büchi tree automata involves Büchi games [9]. In this section, we prove that testing T_W for nonemptiness is equivalent to solving safety games and then testing a Büchi word automata for nonemptiness. This gives us a simpler path towards constructing an algorithm to decide our central question.

4.1 Safety Game for Deviating Agents

Note that the Büchi condition on the j-Deviant traces simply consists of avoiding the set of final states in A^j, making it simpler than a general Büchi acceptance condition. In order to characterize this condition precisely, we now construct a 2-player safety game that partitions the states of Q^j (for ab agent $j \notin W$) in T_W for $j \in \Omega \setminus W$ into two sets - states in which T_W started in state $q \in Q^j$ is empty and states in which T_j started in state $q \in Q^j$ is nonempty. We construct the safety game $G_j = (Q^j, Q^j \times \Sigma, E_j)$. The safety set can intuitively be thought of as all the vertices not in F^j, but for our purposes it is more convenient to not define outgoing transitions from these states - thus making them losing for player 0 by violating the infinite play condition. Player 0 owns Q^j and player 1 owns $Q^j \times \Sigma$. Here we retain our α and β notation in so far as they are both elements of Σ. The edge relation E_j is defined as follows:

(1) $(q, \langle q, \alpha \rangle)) \in E_j$ for $q \in Q^j \setminus F^j$ and $\alpha \in \Sigma$.
(2) $(\langle q, \alpha \rangle, q') \in E_j$ for $q \in Q^j$ and $q' \in Q^j$, where $q' = \delta^j(q, \beta)$ for some $\beta \in \Sigma$ such that $\alpha[-j] = \beta[-j]$.

Note as defined above, if $q \in F^j$, then q has no successor node, and player 0 is stuck and loses the game. Since G_j is a safety game, player 0's goal is to avoid states in F^j and not get stuck. Let $Win_0(G_j)$ be the set of winning states for Player 0 in the safety game G_j.

Theorem 4.1. *A state $q \in Q^j \setminus F^j$ belongs to $Win_0(G_j)$ iff T_W is nonempty when started in state q.*

Proof. ($\rightarrow$) Suppose $q \in Q^j \setminus F^j$ and $q \in Win_0(G_j)$. We construct a tree $\pi_q : \Sigma^* \rightarrow \Sigma$ that is accepted by T_W starting in state q. To show that π_q is accepted, we also construct an accepting run $r_q : \Sigma^* \rightarrow (Q^j \setminus F^j) \cup \{q_A\}$. By construction, we have $r_q(x) \in Win_0(G_j)$ for all $x \in \Sigma^*$. We proceed by induction on the length of the run.

For the basis of the induction, we start by defining $\pi_q(\varepsilon)$ and $r_q(\varepsilon)$. First, we let $r_q(\varepsilon) = q$. By the assumption that $q \notin F^j$, the run cannot get stuck and reject here.

For the step case, suppose now that we have constructed $r_q(y) = p \in Win_0(G_j)$ for some $y \in \Sigma^*$. Now, since $p \in Win_0(G_j)$ and cannot get stuck, there must be a node $\langle q, \alpha_y \rangle$ contained in both $Q^j \times \Sigma$ and $Win_0(G_j)$, so we let $\pi_q(y) = \alpha_y$. Recall that the directions of π are Σ. Divide the possible directions $\beta \in \Sigma$ into two types: either $\alpha_y[-j] = \beta[-j]$ or $\alpha_y[-j] \neq \beta[-j]$. If $\alpha_y[-j] = \beta[-j]$, then this corresponds to a legal move by player 1 in G_j. Since $\langle q, \alpha_y \rangle \in Win_0(G_j)$, moves by player 1 must stay in $Win_0(G_j)$. It follows that $q' = \delta^j(q, \beta) \in Win_0(G_j)$, so $q' \notin F^j$. We let $r_q(y \cdot \beta) = q'$. If, on the other hand, $\alpha_y[-j] \neq \beta[-j]$, we let $r_q(y \cdot \beta) = q_A$. Once we have reached a node $z \in \Sigma^*$ with $r_q(z) = q_A$, we define $r_q(z') = q_A$ for all descendants z' of z and we can define $\pi_q(z')$ arbitrarily. Since we can never get stuck, we never reach a state in F^j, so the run r_q is accepting.

($\leftarrow$) Suppose now that T_W started in state q accepts a tree $\pi_q : \Sigma^* \rightarrow \Sigma$. Since the automaton T_W is deterministic, it accepts with a unique run of T_W on π_q as $r_q : \Sigma^* \rightarrow (Q^j \setminus F^j) \cup \{q_A\}$. We claim that π_q is a winning strategy for player 0 in G_j from the state q.

Consider a play $\pi = p_0, \alpha_0, \beta_0, p_1, \alpha_1, \beta_1, \ldots$, where $p_i \in Q^j$, $p_0 = q$, and $\alpha_i, \beta_i \in \Sigma$. In round $i \geq 0$, player 0 moves from p_i to $\langle p_i, \alpha_i \rangle$, for $\alpha_i = \pi_q(\langle \beta_0, \ldots, \beta_{i-1} \rangle)$, and then player 1 moves from $\langle p_i, \alpha_i \rangle$ to $p_{i+1} = \delta^j(p_i, \beta_i)$, for some β_i such that $\alpha_i[-j] = \beta_i[-j]$. Let $x_i = \langle \beta_0, \ldots, \beta_{i-1} \rangle$, so we have that $\alpha_i = \pi_q(x_i)$. By induction on the length of x_i it follows that $p_i = r_q(x_i)$. Since r_q is an accepting run of T_W on π_q, it follows that $p_i = r_q(x_i) \notin F^j$. Thus, the play π is a winning play for player 0. It follows that π_q is a winning strategy for player 0 in G_j from the state q. □

4.2 A Büchi Automaton for T_W Nonemptiness

Recall that the tree automaton T_W, which recognizes W-NE strategies, emulates the Büchi automaton $A_W = (Q, q_0, \Sigma, \delta, F)$ along the primary trace and the goal automaton A^j along j-deviant traces. We have constructed the above games G_j to capture nonemptiness of T_W from states in Q^j, in terms of the winning sets $Win_0(G_j)$. We now modify A_W to take these safety games into account. Let $A'_W = (Q', q_0, \Sigma, \delta', F \cap Q')$ be obtained from A_W by restricting states to $Q' \subseteq Q$, where $Q' = \bigtimes_{i \in W} Q^i \times \bigtimes_{j \in \Omega \setminus W} \{Win_0(G_j) \cap Q^j\} \times 2^{\Omega}$. In other words, the j^{th}-component q_{i_j} of a state $\overline{q} = \langle q_{i_1}, \ldots, q_{i_n} \rangle \in Q'$ must be in $Win_0(G_j)$ for all $j \in \Omega \setminus W$, otherwise the automaton A'_W gets stuck.

Theorem 4.2. *The Büchi word automaton A'_W is nonempty iff the tree automaton T_W is nonempty.*

Proof. ($\rightarrow$) Assume A'_W is nonempty. Then, it accepts an infinite word $w = w_0w_1\ldots \in \Sigma^\omega$ with a run $r = q_0, q_1, \ldots \in Q'^\omega$. We use w and r to create a tree $\pi : \Sigma^* \rightarrow \Sigma$ with an accepting run $r_\pi : \Sigma^* \rightarrow Q \cup \{q_A\}$ with respect to T_W.

Let $x_0 = \varepsilon$. We start by setting $\pi(x_0) = w_0$ and $r_\pi(x_0) = q_0$. Suppose now that we have just defined $\pi(x_i) = \alpha$ and $r_\pi(x_i) = q$, and, by construction, x_i is on the primary trace. Consider now the node $x_i \cdot \beta$. There are three cases to consider:

(1) If $\pi(x_i) = \beta$, then we set $x_{i+1} = x_i \cdot \beta$, $\pi(x_{i+1}) = w_{i+1}$ and $r_\pi(x_{i+1}) = q_{i+1}$. Note that x_{i+1} is, by construction, the successor of x_i on the primary trace. Thus, the projection of r_π on the primary trace of π is precisely r, so r_π is accepting along the primary path.
(2) If $\pi(x_i)[-j] = \beta[-j]$ and $\pi(x_i) \neq \beta$ for some $j \in \Omega \setminus W$, then we set $r_\pi(x_i \cdot \beta) = q'_j = \delta^j(q_j, \beta)$, where q_j is the j-th component of q. Since $q_j \in Win_0(G_j)$, we have that $q'_j \in Win_0(G_j)$. By Theorem 4.1, T_W is nonempty when started in state q'_j. That is, there is a tree $\pi_{q'_j}$ and an accepting run $r_{q'_j}$ of T_W on $\pi_{q'_j}$, starting from q'_j. So we take the subtree of π rooted at the node $x_i \cdot \beta$ to be $\pi_{q'_j}$, and the run of T_W from $x_i \cdot \beta$ is $r_{q'_j}$. So all paths of r_π that go through $x_i \cdot \beta$ are accepting.
(3) Finally, if $\pi(x_i)[-j] \neq \beta[-j]$ for all $j \in \Omega \setminus W$, then $x_i \cdot \beta$ is neither on the primary trace nor on a j-deviant trace for some $j \in \Omega \setminus W$. So we set $r_\pi(x_i \cdot \beta) = q_A$ as well as $r_\pi(y) = q_A$ for all descendants y of $x_i \cdot \beta$. The labels of $x_i \cdot \beta$ and it descendants can be set arbitrarily. So all paths of r_π that go through $x_i \cdot \beta$ are accepting.

($\leftarrow$) Assume T_W is nonempty. Then, we know that it accepts at least one tree $\pi : \Sigma^* \rightarrow \Sigma$. In particular, since T_W accepts on all branches of π it accepts on the primary trace, denoted as π_p.

Since T_W accepts on π_p, we can consider the run of T_W on π which we denote $r : \Sigma^* \rightarrow Q$. Let the image of $r(\pi_p)$ be $Q^* \subseteq Q$. We claim that $Q^* \subseteq Q'$.

Assume otherwise, that for some finite prefix of the primary trace of π denoted p we have that $r(p) \notin Q'$. Since $r(p)$ clearly is inside Q, it must be the case that for some $j \in \Omega \setminus W$ $r(p)[j] \notin Win_0(G_j)$. Since $r(p)[j]$ is not in $Win_0(G_j)$, it must be in $Win_1(G_j)$. This means that, upon observing p, a direction β exists that transitions T_W into a state $q' \in Win_1(G_j)$. From here player 1 has a winning strategy in G_j. Following one of the paths created by player 1 playing directions according to this winning strategy and player 0 playing anything in response, we get that player 1 will eventually win the game, forcing T_W to attempt a transition into F^j and getting stuck. Therefore T_W does not actually accept π, a contradiction.

Since the image of $r(\pi_p)$ is contained within Q', we claim that A'_W accepts the word formed by the labels along π_p, which we denote by $\alpha(\pi_p)$. Since T_W accepts along π_p and the run $r(\pi_p)$ never leaves Q', we have that there are infinitely many members of the set $F \cap Q'$ in the run $r(\pi_p)$, satisfying the Büchi condition of A'_W. And since any states in which some Q^j for $j \in \Omega \setminus W$ reaches a final state are excluded from Q', A'_W will never get stuck reading $\alpha(\pi_p)$. Therefore, A'_W accepts $\alpha(\pi_p)$ and is therefore nonempty. □

Corollary 4.3. *Let G be an iBG and $W \subseteq \Omega$ be a set of agents. Then, a W-NE strategy exists in G iff the automaton A'_W constructed with respect to G is nonempty.*

5 COMPLEXITY AND ALGORITHMS

5.1 Complexity

The algorithm outlined by our previous constructions consists of two main part. First, we construct and solve a safety game for each agent. Second, for $W \subseteq \Omega$, we check the automaton A'_W for nonemptiness. The input to this algorithm consists of k goal DFAs with alphabet Σ and a set of k alphabets Σ_i corresponding to the actions available to each agent. Therefore, the size of the input is the sum of the sizes of these k goal DFAs.

In the first step, we construct a safety game for each of the agents. The size of the state space of the safety game for agent j is $|Q^j|(|\Sigma| + 1)$. The size of the edge set for the safety game can be bounded by $(|Q^j| * |\Sigma|) + (|Q^j|^2 * |\Sigma|)$, where $|Q^j| * |\Sigma|$ represents the $|\Sigma|$ outgoing transitions from each state in Q^j owned by player 0 and $|Q^j|^2 * |\Sigma|$ is an upper bound assuming that each of the states in $Q^j \times \Sigma$ owned by player 1 can transition to each of the states in Q^j owned by player 0. Since safety games can be solved in linear time with respect to the number of the edges [2], each safety game is solved in polynomial time. We solve one such safety game for each agent which represents a linear blow up. Therefore, solving the safety games for all agents can be done in polynomial time.

For a given $W \subseteq \Omega$, querying the automaton A'_W for nonemptiness can be done in PSPACE, as the state space of A'_W consists of tuples from the product of input DFAs. We can then test A'_W on the fly by guessing the prefix of the lasso and then guessing the cycle, which can be done in polynomial space [30].

Theorem 5.1. *The problem of deciding whether there exists a W-NE strategy profile for an iBG G and a set $W \subseteq \Omega$ of agents is in PSPACE.*

5.2 PSPACE Lower Bound

In this section we show that the problem of determining whether a W-NE exists in an iBG is PSPACE-hard by providing a reduction from the PSPACE-complete problem of DFA Intersection Emptiness (DFAIE). The DFAIE problem is as follows: Given k DFAs $A^0 \ldots A^{k-1}$ with a common alphabet Σ, decide whether $\bigcap_{0 \le i \le k-1} A^i \neq \emptyset$ [15].

Given a DFA $A^i = \langle Q^i, q^i_0, \Sigma, \delta^i, F^i \rangle$, we define the goal DFA $\hat{A}^i = \langle \hat{Q}^i, q^i_0, \hat{\Sigma}, \hat{\delta}^i, \hat{F}^i \rangle$ as follows:

(1) $\hat{\Sigma} = \Sigma \cup \{K\}$, where K is a new symbol, i.e. $K \notin \Sigma$
(2) $\hat{Q}^i = Q^i \cup \{\text{accept}, \text{reject}\}$,
(3)
$$\begin{cases} \hat{\delta}^i(q,a) = q \text{ for } q \in \{\text{accept}, \text{reject}\} \text{ and } a \in \hat{\Sigma} \\ \hat{\delta}^i(q,a) = \delta^i(q,a) \text{ for } q \in Q^i \text{ and } a \in \Sigma \\ \hat{\delta}^i(q,K) = \text{accept for } q \in F^i \\ \hat{\delta}^i(q,K) = \text{reject for } q \in Q^i \setminus F^i \end{cases}$$
(4) $\hat{F}^i = \{\text{accept}\}$

Intuitively, accept and reject are two new accepting and rejecting states that have no outgoing transitions. The new symbol K takes accepting states to accept and rejecting states to reject. The purpose of K is to synchronize acceptance by all goal automata. We call the process of modifying A^i into $\hat{A}^i$ transformation.

The transformation from A^i to $\hat{A}^i$ can be done in linear time with respect to the size of A^i, as the process only involves adding two new states. Furthermore, if A^i is a DFA then $\hat{A}^i$ is also a DFA.

Given an instance of DFAIE, i.e., k DFAs $A^0 \dots A^{k-1}$, we create an iBG G, defined in the following manner.

(1) $\Omega = \{0, 1 \dots k-1\}$
(2) The goal for agent i is $\hat{A}^i$
(3) $\Sigma_0 = \Sigma \cup \{K\}$
(4) $\Sigma_i = \{*\}$ for $i \neq 0$. Here $*$ represents a fresh symbol, i.e., $* \notin \Sigma$ and $* \neq K$.

Clearly, the blow-up of the construction is linear. Since each agent except 0 is given control over a set consisting solely of $*$, the common alphabet of the $\hat{A}^i$ is technically $\hat{\Sigma} \times \{*\}^{k-1}$. This alphabet is isomorphic to $\hat{\Sigma}$, so by a slight abuse of notation we keep considering the alphabet of the $\hat{A}^i$ to be $\hat{\Sigma}$.

Before stating and proving the correctness of the reduction, we make two observations. We are interested here in Nash equilibria in which *every* agent is included in W. This implies the following:

(1) The existence of an Ω-NE is defined solely by the Primary-Trace Condition. Since there are no agents in $\Omega \setminus W$, there is no concept of a j-Deviant-Trace. If we are given an infinite word that satisfies the Primary-Trace Condition, we can extend it to a full Ω-NE strategy tree by labeling the nodes that do not occur on the primary trace arbitrarily.
(2) Since there are no j-Deviant-Traces in this specific instance of the Ω-NE Nonemptiness problem, we can relax our assumption that $|\Sigma_j| \geq 2$ for all $j \in \Omega$, since there is no meaningful concept of deviation in an Ω-NE. Recall that this assumption was made only for simplicity of presentation regarding j-Deviant Traces.

Theorem 5.2. *Let $A^0 \dots A^{k-1}$ be k DFAs with alphabet Σ. Then, $\bigcap_{0 \leq i \leq k-1} L(A^i) \neq \emptyset$ iff there exists an Ω-NE in the iBG G constructed from $A^0 \dots A^{k-1}$.*

Proof. In this proof, we introduce the notation S to denote an infinite suffix, which is an arbitrarily chosen element of $\{\Sigma \cup K\}^{\omega}$.

($\rightarrow$) Assume that $\bigcap_{0 \leq i \leq k-1} L(A^i) \neq \emptyset$. Then, there is a word $w \in \Sigma^*$ that is accepted by each of $A^0 \dots A^{k-1}$. We now show that $w \cdot K \cdot S$ satisfies all goals $\hat{A}^0 \dots \hat{A}^{k-1}$. Since each of $A^0 \dots A^{k-1}$ accepts w, each of $\hat{A}^0 \dots \hat{A}^{k-1}$ reaches a final state of $A^0 \dots A^{k-1}$, respectively, after reading w. Then, after reading K, $\hat{A}^0 \dots \hat{A}^{k-1}$ all simultaneously transition to accept. Therefore all goals $\hat{A}^i$ are satisfied on $w \cdot K \cdot S$ and $w \cdot K \cdot S$ satisfies the Primary-Trace Condition. Since we are considering an Ω-NE, there is no need to check deviant traces and $w \cdot K \cdot S$ can be arbitrarily extended to a full Ω-NE strategy profile tree.

($\leftarrow$) Assume that the iBG G with goals $\hat{A}^0 \dots \hat{A}^{k-1}$ admits an Ω-NE. We claim that its primary trace must be of the form $w \cdot K \cdot S$, where $w \in \Sigma^*$ does not contain K. This is equivalent to saying that a satisfying primary trace must have at least one K. This is easy to see, as the character K is the only way to transition into an accepting state for each $\hat{A}^i$, therefore it must occur at least once if all $\hat{A}^i$ are satisfied on this trace.

We now claim that each of $A^0 \dots A^{k-1}$ accept w. Assume this is not the case, and some A^i does not accept w. Then, while reading w, $\hat{A}^i$ never reaches accept, as w does not contain K. Furthermore, upon seeing the first K, $\hat{A}^i$ transitions to reject, since A^i is not in a final state in F^i after reading w. Thus, $\hat{A}^i$ can never reach accept, contradicting the assumption that $w \cdot K \cdot S$ was an Ω-NE. Therefore all A^i must accept w, and $\bigcap_{0 \leq i \leq k-1} L(A^i) \neq \emptyset$.

□

This establishes a polynomial time reduction from DFAIE to W-NE Nonemptiness; therefore W-NE Nonemptiness is PSPACE-hard. In fact this reduction has shown that checking the Primary-Trace Condition is itself PSPACE-hard. Combining this with our PSPACE decision algorithm yields PSPACE-completeness.

Theorem 5.3. *The problem of deciding whether there exists a W-NE strategy profile for an iBG G and a set $W \subseteq \Omega$ of agents is PSPACE-complete.*

6 CONCLUDING REMARKS

The main contribution of this work is Theorem 5.3, which characterizes the complexity of deciding whether a W-NE strategy profile exists for an iBG G and $W \subseteq \Omega$ is PSPACE-complete.

Separation of Strategic and Temporal Reasoning. : The main objectives of this work is to analyze equilibria in finite-horizon multi-agent concurrent games, focusing on the strategic-reasoning aspect of the problem, separately from temporal reasoning. In order to accomplish this, we used DFA goals instead of goals expressed in some finite-horizon temporal logic. For these finite-horizon temporal logics, previous analysis [12] consisted of two steps. First, the logical goals are translated into a DFA, which involves a doubly exponential blow up [6, 17]. The second step was to perform the strategic reasoning, i.e., finding the Nash equilibria with the DFA from the first step as input. In terms of computational complexity, the first step completely dominated the second step, in which the strategic reasoning was conducted with respect to the DFAs. Here we eliminated the doubly exponential-blow up from consideration by starting with DFA goals and provided a PSPACE-completeness result for the second step.

Future Work. : Our immediate next goals are to analyze problems such as verification (deciding whether a given strategy profile is a W-NE) and strategy extraction (i.e., construction a finite-state controller that implements the W-NEs found) within the context of our DFA based iBGs. Furthermore, we are interested in implementation, i.e. a tool based on the theory developed in this paper. Further points of interest can be motivated from a game-theory lens, such as introducing imperfect information. Earlier work has already introduced imperfect information to problems in synthesis and verification - see [3, 8, 28]. Finally, the work can be extended to both the general CGS formalism (as opposed to iBGs) and to querying other properties/equilibrium concepts outside of the Nash equilibria. *Strategy Logic* [21] has been introduced as a way to query general game theoretic properties on concurrent game structures, and a version of strategy logic with finite goals would be a promising place to start for these extensions.

REFERENCES

[1] Rajeev Alur, Thomas A Henzinger, and Orna Kupferman. 2002. Alternating-time temporal logic. *J. ACM* 49, 5 (2002), 672–713.

[2] Julien Bernet, David Janin, and Igor Walukiewicz. 2002. Permissive strategies: from parity games to safety games. *RAIRO-Theoretical Informatics and Applications-Informatique Théorique et Applications* 36, 3 (2002), 261–275.

[3] Raphaël Berthon, Bastien Maubert, Aniello Murano, Sasha Rubin, and Moshe Y. Vardi. 2018. Strategy Logic with Imperfect Information. *CoRR* abs/1805.12592 (2018). arXiv:1805.12592 http://arxiv.org/abs/1805.12592

[4] J. Elgaard, N. Klarlund, and A. Möller. 1998. Mona 1.x: new techniques for WS1S and WS2S. In *Proc. 10th Int'l Conf. on Computer Aided Verification (Lecture Notes in Computer Science, Vol. 1427)*. Springer, 516–520.

[5] Dana Fisman, Orna Kupferman, and Yoad Lustig. 2010. Rational Synthesis. In *Tools and Algorithms for the Construction and Analysis of Systems, 16th International Conference, TACAS 2010, Held as Part of the Joint European Conferences on Theory and Practice of Software, ETAPS 2010, Paphos, Cyprus, March 20-28, 2010. Proceedings (Lecture Notes in Computer Science, Vol. 6015)*, Javier Esparza and Rupak Majumdar (Eds.). Springer, 190–204. https://doi.org/10.1007/978-3-642-12002-2_16

[6] Giuseppe De Giacomo and Moshe Y. Vardi. 2013. Linear Temporal Logic and Linear Dynamic Logic on Finite Traces. In *IJCAI 2013, Proceedings of the 23rd International Joint Conference on Artificial Intelligence, Beijing, China, August 3-9, 2013*, Francesca Rossi (Ed.). IJCAI/AAAI, 854–860. http://www.aaai.org/ocs/index.php/IJCAI/IJCAI13/paper/view/6997

[7] Giuseppe De Giacomo and Moshe Y. Vardi. 2015. Synthesis for LTL and LDL on Finite Traces. In *Proceedings of the Twenty-Fourth International Joint Conference on Artificial Intelligence, IJCAI 2015, Buenos Aires, Argentina, July 25-31, 2015*, Qiang Yang and Michael J. Wooldridge (Eds.). AAAI Press, 1558–1564. http://ijcai.org/Abstract/15/223

[8] Giuseppe De Giacomo and Moshe Y. Vardi. 2016. LTL_f and LDL_f Synthesis under Partial Observability. In *Proceedings of the Twenty-Fifth International Joint Conference on Artificial Intelligence, IJCAI 2016, New York, NY, USA, 9-15 July 2016*, Subbarao Kambhampati (Ed.). IJCAI/AAAI Press, 1044–1050. http://www.ijcai.org/Abstract/16/152

[9] E. Grädel, W. Thomas, and T. Wilke. 2002. *Automata, Logics, and Infinite Games: A Guide to Current Research.* Springer.

[10] Julian Gutierrez, Paul Harrenstein, and Michael J. Wooldridge. 2015. Iterated Boolean games. *Inf. Comput.* 242 (2015), 53–79. https://doi.org/10.1016/j.ic.2015.03.011

[11] Julian Gutierrez, Muhammad Najib, Giuseppe Perelli, and Michael J. Wooldridge. 2020. Automated temporal equilibrium analysis: Verification and synthesis of multi-player games. *Artif. Intell.* 287 (2020), 103353. https://doi.org/10.1016/j.artint.2020.103353

[12] Julian Gutierrez, Giuseppe Perelli, and Michael J. Wooldridge. 2017. Iterated Games with LDL Goals over Finite Traces. In *Proceedings of the 16th Conference on Autonomous Agents and MultiAgent Systems, AAMAS 2017, São Paulo, Brazil, May 8-12, 2017*, Kate Larson, Michael Winikoff, Sanmay Das, and Edmund H. Durfee (Eds.). ACM, 696–704. http://dl.acm.org/citation.cfm?id=3091225

[13] Mohammadhosein Hasanbeig, Natasha Yogananda Jeppu, Alessandro Abate, Tom Melham, and Daniel Kroening. 2019. DeepSynth: Program Synthesis for Automatic Task Segmentation in Deep Reinforcement Learning. *CoRR* abs/1911.10244 (2019). arXiv:1911.10244 http://arxiv.org/abs/1911.10244

[14] Thomas A. Henzinger. 2005. Games in system design and verification. In *Proceedings of the 10th Conference on Theoretical Aspects of Rationality and Knowledge (TARK-2005), Singapore, June 10-12, 2005*, Ron van der Meyden (Ed.). National University of Singapore, 1–4. https://dl.acm.org/citation.cfm?id=1089935

[15] Dexter Kozen. 1977. Lower Bounds for Natural Proof Systems. In *18th Annual Symposium on Foundations of Computer Science, Providence, Rhode Island, USA, 31 October - 1 November 1977*. IEEE Computer Society, 254–266. https://doi.org/10.1109/SFCS.1977.16

[16] Orna Kupferman, Giuseppe Perelli, and Moshe Y. Vardi. 2016. Synthesis with rational environments. *Ann. Math. Artif. Intell.* 78, 1 (2016), 3–20.

[17] O. Kupferman and M.Y. Vardi. 2001. Model checking of safety properties. *Formal Methods in System Design* 19, 3 (2001), 291–314.

[18] Robert McNaughton. 1993. Infinite Games Played on Finite Graphs. *Ann. Pure Appl. Logic* 65, 2 (1993), 149–184. https://doi.org/10.1016/0168-0072(93)90036-D

[19] Joshua J. Michalenko, Ameesh Shah, Abhinav Verma, Richard G. Baraniuk, Swarat Chaudhuri, and Ankit B. Patel. 2019. Representing Formal Languages: A Comparison Between Finite Automata and Recurrent Neural Networks. In *7th International Conference on Learning Representations, ICLR 2019, New Orleans, LA, USA, May 6-9, 2019*. OpenReview.net. https://openreview.net/forum?id=H1zeHnA9KX

[20] Fabio Mogavero, Aniello Murano, Giuseppe Perelli, and Moshe Y Vardi. 2014. Reasoning about strategies: On the model-checking problem. *ACM Trans. on Computational Logic* 15, 4 (2014), 1–47.

[21] Fabio Mogavero, Aniello Murano, Giuseppe Perelli, and Moshe Y. Vardi. 2014. Reasoning About Strategies: On the Model-Checking Problem. *ACM Trans. Comput. Log.* 15, 4 (2014), 34:1–34:47. https://doi.org/10.1145/2631917

[22] John F. Nash. 1950. Equilibrium points in n-person games. *Proceedings of the National Academy of Sciences* 36, 1 (1950), 48–49. https://doi.org/10.1073/pnas.36.1.48 arXiv:https://www.pnas.org/content/36/1/48.full.pdf

[23] Amir Pnueli. 1977. The Temporal Logic of Programs. In *18th Annual Symposium on Foundations of Computer Science, Providence, Rhode Island, USA, 31 October - 1 November 1977*. IEEE Computer Society, 46–57. https://doi.org/10.1109/SFCS.1977.32

[24] Amir Pnueli and Roni Rosner. 1989. On the Synthesis of a Reactive Module. In *Conference Record of the Sixteenth Annual ACM Symposium on Principles of Programming Languages, Austin, Texas, USA, January 11-13, 1989*. ACM Press, 179–190. https://doi.org/10.1145/75277.75293

[25] Senthil Rajasekaran and Moshe Y. Vardi. 2021. Nash Equilibria in Finite-Horizon Multiagent Concurrent Games. arXiv:2101.00716 [cs.GT]

[26] Yoav Shoham and Kevin Leyton-Brown. 2009. *Multiagent Systems - Algorithmic, Game-Theoretic, and Logical Foundations*. Cambridge University Press.

[27] Michael Sipser. 2006. *Introduction to the Theory of Computation* (second ed.). Course Technology.

[28] Lucas M. Tabajara and Moshe Y. Vardi. 2020. LTLf Synthesis under Partial Observability: From Theory to Practice. *CoRR* abs/2009.10875 (2020). arXiv:2009.10875

[29] Johan van Benthem. 2011. Logic Games: From Tools to Models of Interaction. In *Proof, Computation and Agency - Logic at the Crossroads*, Johan van Benthem, Amitabha Gupta, and Rohit Parikh (Eds.). Synthese library, Vol. 352. Springer, 183–216. https://doi.org/10.1007/978-94-007-0080-2_11

[30] M.Y. Vardi and P. Wolper. 1994. Reasoning about Infinite Computations. *Information and Computation* 115, 1 (1994), 1–37.

[31] Michael J. Wooldridge. 2009. *An Introduction to MultiAgent Systems, Second Edition.* Wiley.

[32] Eran Yahav. 2018. From Programs to Interpretable Deep Models and Back. In *Computer Aided Verification - 30th International Conference, CAV 2018, Held as Part of the Federated Logic Conference, FloC 2018, Oxford, UK, July 14-17, 2018, Proceedings, Part I (Lecture Notes in Computer Science, Vol. 10981)*, Hana Chockler and Georg Weissenbacher (Eds.). Springer, 27–37. https://doi.org/10.1007/978-3-319-96145-3_2