Proceedings of the 19th International Conference on Principles of Knowledge Representation and Reasoning
Main Track

# Verification and Realizability in Finite-Horizon Multiagent Systems

**Senthil Rajasekaran and Moshe Y. Vardi**
Rice University
{sr79, vardi}@rice.edu

**Abstract**

The problems of *verification* and *realizability* are two central themes in the analysis of reactive systems. When multiagent systems are considered, these problems have natural analogues of existence (nonemptiness) of pure-strategy Nash equilibria and verification of pure-strategy Nash equilibria. Recently, this body of work has begun to include finite-horizon temporal goals. With finite-horizon temporal goals, there is a natural hierarchy of goal representation, ranging from deterministic finite automata (DFA), to nondeterministic finite automata (NFA), and to alternating finite automata (AFA), with a worst-case exponential gap between each successive representation. Previous works showed that the realizability problem with DFA goals was PSPACE-complete, while the realizability problem with temporal logic goals is in 2EXPTIME. In this work, we study both the realizability and the verification problems with respect to various goal representations. We first show that the realizability problem with NFA goals is EXPTIME-complete and with AFA goals is 2EXPTIME-complete, thus establishing strict complexity gaps between realizability with respect to DFA, NFA, and AFA goals. We then contrast these complexity gaps with the complexity of the verification problem, where we show that verification with respect to DFAs, NFA, and AFA goals is PSPACE-complete.

## 1 Introduction

*Verification* (Clarke, Emerson, and Sistla 1986) and *Realizability* (Pnueli and Rosner 1989) are two major decision problems in the study of reactive systems. When the goals of these systems are specified through linear temporal logics, game theory has provided a powerful modeling framework for both problems through a two-agent game in which one agent takes on the role of a system that tries to realize a property and the other takes on the role of the environment that tries to falsify the property. The verification problem corresponds to checking whether an input strategy is winning for the system agent in the relevant game (Kupferman and Vardi 1996), and the realizability (also called *nonemptiness*) problem corresponds to determining whether a winning strategy for the system agent exists (Pnueli and Rosner 1989).

When the number of autonomous agents increases, the games become *concurrent multiagent games*, suitable for analyzing concurrent multiagent systems (Shoham and Leyton-Brown 2009). In this setting, the notion of a winning strategy no longer corresponds to a meaningful solution concept, as there are no longer only two agents with a purely adversarial relationship. In these types of systems, the concept of a *pure-strategy Nash equilibria* (henceforth, Nash equilibria) has come to be a widely used solution concept (Bouyer et al. 2015). Informally, a Nash equilibria is a profile of strategies such that for each agent in the system deviating from the profile is never more profitable than not . In this sense, Nash equilibria represent a stable point that games naturally tend towards over repeated play (Nash 1950; Gutierrez, Harrenstein, and Wooldridge 2015a).

Concurrent multiagent games represent an extremely broad class of games. *Iterated Boolean Games* (Gutierrez, Harrenstein, and Wooldridge 2015b) are a restriction of concurrent multiagent games that naturally mirror the games that model the two-agent realizability and verification problems (Kupferman and Vardi 1996; Pnueli and Rosner 1989). In an iterated boolean game each agent has a temporal goal and at each time step assigns a setting to a unique collection of boolean variables under its control. Thus, when all agents' assignments are considered we are given a complete valuation of the boolean variables at each time step. This infinite sequence of valuations is then used to determine which temporal goals are satisfied and which are not. Finding Nash equilibria in such games corresponds to a useful method of analysis of the systems that these games model; as such, there is a very large of body finding Nash equilibria when agents' goals are given by an infinite-horizon logic such as *Linear Time Temporal Logic* (LTL) (Wooldridge 2009; Gutierrez et al. 2020; Mogavero et al. 2014; Grädel, Thomas, and Wilke 2002; Abate et al. 2021).

Some systems, however, are naturally modeled by agents with finite-horizon goals, such as when notions like 'completion' are considered. The concept of a finite-horizon temporal logic remains a relatively recent development in the study of temporal logics (Giacomo and Vardi 2013). While agents still create an infinite trace by setting their variables at every time step, satisfaction is considered over finite prefixes. The analogous problem of finding Nash equilibria in iterated boolean games in which each agent has been given a finite-horizon temporal goal has recently begun to receive attention (Rajasekaran and Vardi 2021; Gutierrez, Perelli, and Wooldridge 2017).

Our modelling of this problem is done from the viewpoint of a system designer. Specifically, when given a system in which multiple agents have finite-horizon temporal logic goals, we query a subset $W$ of "good" agents to see if there is Nash equilibrium in which precisely the agents in $W$ are able to satisfy their goals. By the definition of the Nash equilibrium, this means that agents not within $W$, which we consider as "bad" agents, are unable to unilaterally change their strategy and satisfy their own "bad" goal. In doing so we can naturally incorporate malicious agents with goals contrary to the designer's intent by specifying a set $W$ that not contain such agents. This study of teams of cooperating agents has clear parallels in earlier work in rational synthesis (Fisman, Kupferman, and Lustig 2010; Kupferman, Perelli, and Vardi 2016).

Here we consider *Linear Time Temporal Logic on Finite Traces* (LTLf) (Giacomo and Vardi 2013) as our standard finite-horizon temporal logic, but by using an automata-based approach we are able to prove more general results that are independent of a specific logic by considering the size of the automata that represents the specification. Since finite-horizon temporal logics describe languages of finite words, they admit a variety of equivalent representations, ranging from deterministic finite automata (DFAs) to nondeterministic finite automata (NFAs) to alternating finite automata (DFAs) (Giacomo and Vardi 2013; Giacomo and Vardi 2015; Giacomo and Vardi 2016). While alternating finite automata are polynomial in the size with respect to their corresponding $LTL_f$ formula, nondeterministic automata are exponential and deterministic finite automata are doubly exponential (Giacomo and Vardi 2013). By reasoning about different types of input automata, we are able to reason broadly about finite-horizon temporal goals with different goal succinctness from a complexity-theoretic viewpoint. It is then natural to consider how the succinctness of the representation influences the complexities of the realizability and verification problems.

Our investigation sheds new light on the computational complexity of temporal Nash Equilibria. Note that, in prior work, the verification problem is usually proven to be easier than the realizability problem from a complexity-theoretic viewpoint. This corresponds to our intuition, since the verification problem checks a single input candidate strategy, while the realizability problem tries to find some solution strategy. Here, we observe the same phenomena of realizability being more difficult than verification. The succinctness of the representation does not, however, seem to effect the complexity of the verification problem in this setting. No matter which representation we use, we get a PSPACE-complete complexity result. In contrast, if we consider the realizability problem then we get a strict hierarchy. For DFA goals, the problem is PSPACE-complete (Rajasekaran and Vardi 2021) (the same as verification, an exception to the intuition that verification is easier than realizability). From NFAs goals, the problem is EXPTIME-complete. Finally, for AFA goals, the problem is 2EXPTIME-complete. This analysis extends the state of the art to include a complete set of results for both problems with varying representations.

Our approach follows the approach outlined in (Rajasekaran and Vardi 2021) in that we consider the Nash equilibria as two separate conditions - one that corresponds to correct behavior under deviation (the $j$-Deviant Trace Condition) and one that corresponds to correct behavior when no deviations are observed (the Primary-Trace condition). Using this characterization we are able to prove a suite of new results and prove novel variants of a few older results that appeared in previous works under this unified framework. Taken together, they represent a complete characterization of the complexity of both problems for the three main types of automata-theoretic representations common to the literature on finite-horizon temporal logic.

## 2 Background

The background presented here largely follows (Rajasekaran and Vardi 2021). We assume familiarity with automata theory, as in (Sipser 2006; Vardi 1996).

### 2.1 Games

In this section we provide some definitions related to two-agent games to provide a standard notation throughout this paper. The two agents are denoted agent 0 and agent 1.

**Definition 2.1** (Arena)**.** An *arena* is a four tuple $A = (V, V_0, V_1, E)$ where $V$ is a finite set of vertices, $V_0$ and $V_1$ are disjoint subsets of $V$ with $V_0 \cup V_1 = V$ that represent the vertices that belong to agent 0 and agent 1 respectively, and $E \subseteq V \times V$ is a set of directed edges, i.e. $(v, v') \mid \in E$ if there is an edge from $v$ to $v'$.

Intuitively, the agent that owns a node decides which outgoing edge to follow. Since $V = V_0 \cup V_1$, we omit $V$ and write $A = (V_0, V_1, E)$.

**Definition 2.2** (Play)**.** A *play* in an arena $A$ is an infinite sequence $\rho = \rho_0\rho_1\rho_2 \ldots \in V^\omega$ such that $(\rho_n, \rho_{n+1}) \in E$ holds for all $n \in \mathbb{N}$. We say that $\rho$ starts at $\rho_0$

**Definition 2.3** (Game)**.** A *game* $G = (A, Win)$ consists of an arena $A$ with vertex set $V$ and a set of winning plays $Win \subseteq V^\omega$. A play $\rho$ is winning for agent 0 if $\rho \in Win$, otherwise it is winning for agent 1.

Note that in this formulation of a game, reaching a state $v \in V$ with no outgoing transitions is always losing for agent 0, as agent 0 is the one that must ensure that $\rho$ is infinite ( a member of $V^\omega$).

A game is thus defined by its set of winning plays, often called the *winning condition*. One such widely used winning condition is the *safety condition*.

**Definition 2.4** (Safety Games)**.** Let $A = (V, V_0, V_1, E)$ be an arena and $S \subseteq V$ be a subset of $A$'s vertices. Then, the *safety condition* $Safety(S)$ is defined as $Safety(S) = \{\rho \in V^\omega \mid Occ(\rho) \subseteq S\}$, where $Occ(\rho)$ denotes the subset of vertices that occur at least once in $\rho$. A game with the safety condition for a subset $S$ is a *safety game* with the set $S$ of *safe vertices*. Information about solving safety games, including notions of *winning strategies* and *winning sets* can be found here (McNaughton 1993).

## 2.2 Concurrent Games and iBGs

A *concurrent game structure* (CGS) is an 8-tuple

$$(Prop, \Omega, (ACT_i)_{i\in\Omega}, S, \lambda, \tau, s_0 \in S, (A^i)_{i\in\Omega})$$

where $Prop$ is a finite set of *propositions*, $\Omega = \{0, \dots k-1\}$ is a finite set of *agents*, $ACT_i$ is a set of *actions*, where each $ACT_i$ is associated with Agent $i$, the set of *decisions* is $D = ACT_0 \times ACT_1 \dots ACT_{k-1}$, $S$ is a set of *states*, $\lambda : S \to 2^{Prop}$ is a *labeling function* that associates each state with a set of propositions that are interpreted as true in that state, $\tau : S \times D \to S$ is a deterministic *transition function* that takes a state and a decision as input and returns another state, $s_0 \in S$ is the *initial state*, and $A^i$ is a *goal* specification for Agent $i$ given in the form of an deterministic finite automaton (DFA), nondeterministic finite automaton (NFA), or alternating finite automaton (AFA). [1] We say that a finite word automaton accepts an infinite word $\omega$ if it accepts a finite prefix of $\omega$. In a CGS, Agent $i$ prefers plays in the game that satisfy $A^i$, that is, a play such that some finite prefix of the play is accepted by $A^i$. For a goal automaton we use the notation $A^i = \langle Q^i, q_0^i, \Sigma, \delta^i, F^i \rangle$, where $Q^i$ is the state space, $q_0^i \in Q^i$ is the initial state, $\Sigma$ is the alphabet, $\delta^i$ is the transition function, and $F^i$ is the set of final states. In this paper we refer to arbitrary infinite sequences of states in a CGS as *traces*.

We now define *iterated boolean games* (iBG), a restriction on the CGS formalism (Gutierrez, Harrenstein, and Wooldridge 2015b). We follow the formulation of (Rajasekaran and Vardi 2021), as we take the set of actions to be a finite alphabet rather than a set of truth assignments. An iBG is defined by applying the following restrictions to the CGS formalism. Agent $i$ 'owns" alphabet $\Sigma_i$. These $\Sigma_i$ are disjoint and each $\Sigma_i$ serves as the set of actions for Agent $i$-an action for agent $i$ consists of choosing a letter in $\Sigma_i$. The set of decisions is then $\Sigma = \bigtimes_{i=0}^{k-1} \Sigma_i$. The set of states is also $\Sigma$, and the labeling function is the identity function, i.e., $\lambda(s) = s$. Our use of the iBG model is motivated by presentation, as iBGs offer a simple model in which agent actions influence global states - considering general CGS models would not influence the forthcoming complexity-theoretic results. Finally, the transition function $\tau$ is simply the right projection $\tau(s, d) = d$.

We now introduce the notion of a *strategy* for Agent $i$ in the general CGS formalism.

**Definition 2.5** (Strategy for Agent $i$)**.** A strategy for Agent $i$ is a function $\pi_i : S^* \to ACT_i$. Intuitively, this is a function that, given the observed history of the game (represented by an element of $S^*$), returns an action $act_i \in ACT_i$.

Recalling that $\Omega = \{0, 1 \dots k-1\}$ represents the set of agents, we now introduce the notion of a *strategy profile*.

**Definition 2.6** (Strategy Profile)**.** Let $\Pi_i$ represent the set of strategies for agent $i$. We define the set of strategy profiles $\Pi = \bigtimes_{i\in\Omega} \Pi_i$ and denote a single strategy profile as $\pi$.

A strategy profile can be naturally thought of as a function of type $\Sigma^* \to \Sigma$, and we will call any function with such type a *global strategy*. Since a global strategy is deterministic, it yields a unique element of $S^\omega$, which we call a *primary trace*.

**Definition 2.7** (Primary Trace resulting from a Global Strategy)**.** Given a global strategy $\pi : \Sigma^* \to \Sigma$, the primary trace of $\pi$ is the unique trace $t$ that satisfies

1. $t[0] = \pi(\epsilon)$
2. $t[i] = \pi(t[0], \dots t[i-1])$

We denote this trace as $t_\pi$.

Given a trace $t \in S^\omega$, define the *winning set* $W_t = \{i \in \Omega : t \models A^i\}$ to be the set of agents whose DFA goals are satisfied by a finite prefix of the trace $t$. The *losing set* is then defined as $\Omega / W_t$.

A common solution concept in game theory is the *Nash equilibrium*, which we adapt to our iBG framework. In our framework, a Nash equilibrium is a strategy profile $\pi$ such that for each Agent $i$, if $A^i$ is not satisfied on $t_\pi$, then a unilateral strategy deviation for Agent $i$ does not result in a trace that satisfies $A^i$.

**Definition 2.8** (Nash Equilibrium)**.** (Gutierrez, Harrenstein, and Wooldridge 2015b) Let $G$ be an iBG and $\pi = \langle \pi_0, \pi_1 \dots \pi_{k-1} \rangle$ be a strategy profile. We denote $W_\pi = W_{t_\pi}$. The profile $\pi$ is a *Nash equilibrium* if for every $i \in \Omega / W_t$ we have that for each strategy profile of the form $\pi' = \langle \pi_0, \pi_1 \dots \pi_i' \dots \pi_{k-1} \rangle$, with $\pi_i' \in \Pi_i$, it is the case that $i \in \Omega / W_{\pi'}$.

This definition provides an analogy for the Nash Equilibrium defined in (Nash 1950) by capturing the same property - no agent can unilaterally deviate to improve its own payoff (moving from an unsatisfied goal to a satisfied goal). Agents in the set $W_\pi$ cannot have their payoff improved further, so we do not check their deviations. We say that $\pi$ is a $W$-NE iff $\pi$ is a Nash Equilibrium with $W_\pi = W$.

We have already defined the primary trace, which corresponds to the trace that results from no deviations to a profile $\pi$. Since we consider unilateral deviations from a single agent in our analysis, we define these traces as well.

**Definition 2.9** ($j$-Deviant-Trace from a Strategy Profile)**.** Given a strategy profile $\pi$, a $j$-Deviant-Trace (w.r.t $\pi$) is defined as follows. For $\alpha \in \Sigma$, we introduce the notation $\alpha[-j]$ to refer to $\alpha|_{\Sigma \setminus \Sigma_j}$ (that is, $\alpha$ with $\Sigma_j$ projected out). A trace $t = y_0, y_1, \dots$ is $j$-deviant if

1. $y_0 = \varepsilon$
2. $y_{i+1} = y_0, \dots, y_i, \alpha$, where $\alpha \in \Sigma$ and $\alpha[-j] = \pi(y_i)[-j]$
3. $t$ is not the primary trace

Our characterization of the Nash equilibria is based on (Rajasekaran and Vardi 2021), in which the Nash Equilibrium condition was decomposed into the *Primary-Trace* and *j-Deviant-Trace* Condition, which we reintroduce here. For a fixed strategy profile $\pi$ we have:

1. Primary-Trace Condition: The primary infinite trace $t_\pi$ defined by $\pi$ satisfies the goals $A^i$ for precisely $i \in W$.

[1] For automata-theoretic background, see (Vardi 1996)

2. $j$-Deviant-Trace Condition: Each *$j$-deviant* trace $t = y_0, y_1, \ldots$ (w.r.t $\pi$) for $j \notin W$, does not satisfy the goal $A^j$.

A strategy profile $\pi : \Sigma^* \rightarrow \Sigma$ is then a $W$-NE iff it satisfies both the Primary-Trace Condition and the $j$-Deviant Trace Condition w.r.t $W$. In this definition we work with a general notion of the goal $A^j$ accepting a prefix, therefore this universally applies to DFA, NFA, and AFA goals.

The main insight of (Rajasekaran and Vardi 2021) was to show that these two conditions could be reasoned about through automata-theoretic means *separately* when considering DFA goals. In order to analyze the $j$-Deviant-Trace Condition for a single agent $j$ with goal $A^j$, a safety game $G_j$ with $V_0 = Q^j$ and $V_1 = \{Q^j \times \Sigma\}$ and 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 all $\alpha \in \Sigma$.
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 $\alpha[-j] = \beta[-j]$.

The winning set of agent 0 in this game is denoted $Win_0(G_j)$; the winning set for agent 1 is $Win_1(G_j)$.

In order to reason about the Primary Trace, we construct the deterministic Büchi automaton (Vardi 1996) $A_W = \langle Q, q_0, \Sigma, \delta, F \rangle$ with

1. $Q = (\times_{j \in \Omega} Q^j) \times 2^\Omega$
2. $q_0 = \langle q_0^1, \ldots, q_0^n, W \rangle$
3. $F = (\times_{j \in \Omega} Q^j) \times \{\emptyset\}$
4. $\delta(\langle q_1, \ldots q_n, U \rangle, \alpha) = \langle q_1', \ldots q_n', V \rangle$ if $q_j' = \delta^i(q_j, \alpha)$, where $q_j' \notin F^j$ for $j \notin W$, and $V = U - \{i : q_i' \in F^i\}$.

The intuition is that a word in $\Sigma^\omega$ is accepted by $A_W$ iff only the goals in $W$ are satisfied on it.

The winning sets for agent 0 in the games $G_j$ are now used to refine the state space and transition function of $A_W$ to create the deterministic Büchi automaton $A'_W = (Q', q_0, \Sigma, \delta', F \cap Q')$, with $Q' = \times_{i \in W} Q^i \times \times_{j \in \Omega \setminus W} \{Win_0(G_j) \cap Q^j\} \times 2^\Omega$, and $\delta'$ defined as follows: $\delta'(q, \alpha) = \delta(q, \alpha)$ if, for all $j \notin W$, we have that $(q[j], \alpha) \in Win_0(G_j)$; otherwise, $\delta'(q, \alpha)$ is undefined. It was then shown that

**Theorem 1.** (Rajasekaran and Vardi 2021) *For a given iBG $G$, a $W$-NE strategy exists in $G$ iff $A'_W$ is nonempty.*

Since the goal automata being considered have an input alphabet that is the cross product of $k$ other alphabets ($\Sigma = \Sigma_0 \times \Sigma_1 \ldots \Sigma_{k-1}$) , they can be seen as exponential constructions themselves. For this reason, we introduce *bounded-channel automata*.

**Definition 2.10** (Bounded-Channel Automaton)**.** Let $\Sigma = \Sigma_0 \times \Sigma_1 \ldots \Sigma_{k-1}$ and let $I \subset \{0 \ldots k-1\}$ be a strict subset of agents. A *bounded-channel automaton* is an automaton with a transition function $\rho$ that satisfies the property that for all $\alpha, \beta \in \Sigma$ and states $q$ in the automaton, if $\alpha_I = \beta_I$, i.e $\alpha$ and $\beta$ agree on $\Sigma_i$ for every $i \in I$, then $\rho(q, \alpha) = \rho(q, \beta)$, i.e. the transition function only considers $\Sigma_i$ for $i \in I$ on a state $q$ and a symbol $\alpha \in \Sigma$.

Intuitively, a bounded-channel automaton does not consider the actions of every agent through an element of $\Sigma$ but has an input alphabet $\Sigma_B = \times_{i \in I} \Sigma_i$ where $I \subset \Omega$. Note that if $|I|$ is a constant w.r.t $|\Omega| = k$, then such automata have polynomial-sized alphabets. While the use of bounded-channel goal automata does not affect many of the complexity results in this paper, considering bounded channel automata allows us to reason about a more succinct input type that arguably better corresponds to realistic situations.

# 3 Realizability

In this section we study the *realizability problem* (referred to as the *nonemptiness problem* in (Gutierrez, Harrenstein, and Wooldridge 2015b; Rajasekaran and Vardi 2021)) in which we are are given an iBG $G$ *and* a set $W \subseteq \Omega$ of agents and we wish to decide if a Nash equilibria strategy profile exists in which only the agents in $W$ have their goals satisfied. In (Rajasekaran and Vardi 2021), this problem was proven to be PSPACE-complete for DFA goals, using the safety games $G_j$ and Büchi automaton $A'_W$, as described in section 2.2. We now analyze the complexity for NFA and AFA goals.

## 3.1 NFA Goals

Assume the input goal automata are NFAs. By determinizing each goal automaton, we can readily apply the procedure from (Rajasekaran and Vardi 2021). Constructing the DFA goal automaton $A^j = \langle Q^j, q_0^j, \Sigma, \delta^j, F^j \rangle$ from the NFA goal input involves a worst-case exponential blowup in the number of states, with no blow-up in the size of the alphabet $\Sigma$. Therefore, the state space of each safety game $G_j$, given by $Q^j \cup \{Q^j \times \Sigma\}$ is overall exponential in the size of the goal NFAs. Safety games can be solved in time linear in the size of the game (McNaughton 1993), so each relevant $G_j$ can be analyzed in EXPTIME.

The automaton $A'_W$ from (Rajasekaran and Vardi 2021) has a state space $Q'$ that is upper bounded by the size of the cross product of all DFA goals and $2^\Omega$. Each DFA goal has a state space that is exponential in the size of the NFA, so the product of all DFAs is still singly exponential with respect to the NFA goals. Furthermore, $2^\Omega$ is also singly exponential, implying that $A'_W$ is singly exponential in size with respect to the input. Testing a Büchi automaton for nonemptiness can be done in NLOGSPACE (Vardi and Wolper 1994), meaning that $A'_W$ can be checked for nonemptiness in PSPACE. Since the safety games are in EXPTIME in the worst case, the overall complexity of this method is in EXPTIME. These results still hold when considering bounded-channel goals, as they are still polynomial in the size of the input.

The upper bound for the realizability problem with NFA goals is analyzed above. For lack of space, we do not include here the lower-bound proof, which holds even for two-agent iBGs (and therefore holds for the bounded-channel case as well). The proof is included in detail in our corresponding technical report (Rajasekaran and Vardi 2022) [2]

[2] https://arxiv.org/abs/2205.01029

**Theorem 2.** *The realizability problem with NFA goals is EXPTIME-complete.*

### 3.2 AFA Goals

We now analyze the case of the realizability problem when the agents' goals are specified by AFAs. Since AFAs are linear (in number of states) in the size of equivalent finite-trace temporal specifications such as $LTL_f$ or $LDL_f$ (Giacomo and Vardi 2013), we note that a similar problem of deciding whether any $W$-NE exists was given a 2EXPTIME upper bound, but no lower bound, in (Gutierrez, Perelli, and Wooldridge 2017). Here we focus on the a single agent set $W$ and provide a tight 2EXPTIME bound.

Constructing the DFA goal automata $A^j$ from the AFA goals involves a doubly exponential worst-case blowup in the number of states (Giacomo and Vardi 2013), with no blow-up in the size of the alphabet $\Sigma$ - this holds for bounded-channel automata as well. The analysis now largely follows the NFA-goals case. The size of the safety game $G_j$ is now doubly exponential in the size of the input due to the presence of $Q^j$, and so each $G_j$ can be solved in 2EXPTIME. Meanwhile $A'_W$ consists of the cross product of the doubly exponential $Q^j$s and the singly exponential $2^\Omega$, so it is doubly exponential overall. Following the same logic as before yields a 2EXPTIME upper bound.

**Theorem 3.** *The realizability problem with AFA goals can be solved in 2EXPTIME.*

This result agrees with the result in (Gutierrez, Perelli, and Wooldridge 2017) (where $W$ is not part of the input). We now extend the analysis by providing a matching lower bound, proving the problem to be 2EXPTIME-hard by reducing from the 2EXPTIME-complete problem of $LTL_f$ realizability, noting that there is a linear-time conversion from $LTL_f$ formulas to equivalent AFA for a fixed-size alphabet (Giacomo and Vardi 2013). We note that the 2EXPTIME lower bound for $LTL_f$ realizability holds already for fixed alphabet goals, as in (Rosner 1992).

The $LTL_f$ *realizability problem* (Giacomo and Vardi 2015) takes as input an $LTL_f$ formula $\phi$ along with a partition of the variables $V$ in $\phi$ into two sets $X$ and $Y$. The problem asks whether an agent (Agent 0) that takes control of the variables in $X$ can always ensure a trace satisfying $\phi$ with an antagonistic agent (Agent 1) setting the variables in $Y$. At each time step the agents set their variables, thus producing an infinite trace over $2^{X\cup Y}$. As before, $\phi$ is satisfied by an infinite trace if it is satisfied by some finite prefix of that infinite trace. The interaction is naturally modeled as a game between the two agents, which is called the *$\phi$-realizability game*. There are several variations of these games, we consider the variation where Agent 0 and Agent 1 move concurrently, assigning values to the $X$ and $Y$ variables, respectively (Giacomo and Vardi 2015).

Given an instance of the $LTL_f$ realizability problem with goal $\phi$, we construct an iBG $G_\phi$ with two agents. Agent 0 is given the goal $\phi$ expressed as AFA and Agent 1 is given an empty goal, i.e., an AFA that accepts the empty language. Let $\Sigma_0 = 2^X$, $\Sigma_1 = 2^Y$, and the set $W$ be the empty set. Since we assume that the temporal goal has a bounded alphabet, the translation to AFAs is linear in number of states.

**Theorem 4.** *Given an $LTL_f$ formula $\phi$, Agent 0 wins the $\phi$-realizability game iff no $\emptyset$-NE exists in $G_\phi$.*

*Proof.* ($\rightarrow$) Assume that Agent 0 wins the $\phi$-realizability game. Then, there exists a strategy $\pi'_0 : (2^Y)^* \rightarrow 2^X$ that ensures that the formula $\phi$ is eventually satisfied given an arbitrary Agent 1 strategy $\pi_1 : (2^X)^* \rightarrow 2^Y$. Therefore, there can not be $\emptyset$-NE in $G$. Suppose to the contrary that the profile $\langle \pi_0, \pi_1 \rangle$ is an $\emptyset$-NE, which means that the primary trace does not satisfy $\phi$. Agent 0 can now deviate from this profile and follow the strategy $\pi'_0$, so now Agent 0 and Agent 1 are following the profile $\langle \pi'_0, \pi_1 \rangle$. Since $\pi'_0$ is a winning strategy in the $\phi$-realizability game, Agent 0 is able to force satisfaction of $\phi$, so $\langle \pi_0, \pi_1 \rangle$ is not an $\emptyset$-NE.

($\leftarrow$) Assume to the contrary that there is an $\emptyset$-NE in $G_\phi$. We show that it implies that Agent 1 wins the $\phi$-realizability game. Let $\langle \pi_0, \pi_1 \rangle$ be the strategy profile for the $\emptyset$-NE in $G_\phi$. This means that $\phi$ is not satisfied in the primary trace, and, furthermore, for every strategy profile $\langle \pi'_0, \pi_1 \rangle$ the primary trace does not satisfy $\phi$. This means that $\pi_1$ is winning strategy for Agent 1 in the $\phi$-realizability game.

□

Since $LTL_f$-realizability is known to be 2EXPTIME-complete (Giacomo and Vardi 2015), we get:

**Theorem 5.** *The realizability problem with AFA goals is 2EXPTIME-complete.*

Since this lower bound was shown for two-agent games, it holds for the bounded-channel case as well.

## 4 Verification

We now address the verification problem in which we are given an iBG $G$, a set $W \subseteq \Omega$ of agents, and a strategy profile $\pi = \langle \pi_0 \dots \pi_{k-1} \rangle$, where $k = |\Omega|$. We are given the strategy profile in terms of the individual strategies: Each $\pi_i = \langle S^i, s^i_0, \Sigma, \Sigma_i, \rho^i, \gamma^i \rangle$ is a Moore machine (Sipser 2006) that represents a function of type $\Sigma^* \rightarrow \Sigma_i$, where $S^i$ is the set of states, $s^i_0 \in S^i$ is the initial state, $\Sigma_i$ is the alphabet controlled by agent $i$, $\rho^i : S^i \times \Sigma \rightarrow S^i$ is the transition function, and $\gamma^i : S^i \rightarrow \Sigma^i$ is the output function. $\Sigma = \times_{i\in\Omega} \Sigma_i$ is the common alphabet of the goals in $G$. The *verification problem* takes as inputs $G$, $W$, and $\pi$ and outputs whether $\pi$ is a $W$-NE in $G$.

We first construct a Moore machine for $\pi$ from $\pi_0 \dots \pi_{k-1}$ using the standard product construction: $\pi = \langle S, s_0, \Sigma, \Sigma, \rho, \gamma \rangle$, where

1. $S = \times_{i\in\Omega} S^i$.
2. $s_0 = \langle s^0_0, \dots, s^{k-1}_0 \rangle$.
3. $\Sigma$ is both the input and output alphabet in $\pi$.
4. The transition function $\rho$ is defined component-wise. For $\bar{t} = \langle t_0, \dots, t_{k-1} \rangle \in S$ and $\alpha \in \Sigma$, we have $\rho(\bar{t}, \alpha) = \langle t'_0, \dots, t'_{k-1} \rangle$, where $t'_i = \rho^i(t_i, \alpha)$, for each $i \in \Omega$.
5. The output function $\gamma$ is defined component-wise. For $\bar{t} = \langle t_0, \dots, t_{k_1} \rangle \in S$, we have $\gamma(\bar{t}) = \langle \sigma_0, \dots, \sigma_{k-1} \rangle$, where $\sigma_i = \gamma^i(t_i)$, for each $i \in \Omega$.

Note that the size of $\pi$ is exponential in the sizes of $\pi_0 \dots \pi_{k-1}$.

We now define the state outcome of running $\pi$ on input words in $\Sigma^*$, inductively:

- $\pi(\varepsilon) = s_0$, and
- $\pi(w\alpha) = \rho(\pi(w), \alpha)$, for $w \in \Sigma^*$ and $\alpha \in \Sigma$

The output of running $\pi$ on a word $w \in \Sigma^*$ is then $\gamma(\pi(w))$. In this way, we get that $\pi$ represents a function of type $\Sigma^* \to \Sigma$ ,i.e. a global strategy (see Definition 2.6), which yields a primary trace (Definition 2.7).

We can define bounded-channel Moore Machines for single-agent strategies in the same way we have defined bounded-channel goal automata. The construction is the same - instead of considering input alphabet $\Sigma$, we consider a restricted version that only considers some subset of agents. This allows us to consider single-agent strategies that have a succinct representation just as we considered goals with succinct representations. We now proceed to analyze the complexity of verification with respect to DFA, NFA, and AFA goals.

## 4.1 DFA Goals

As described in Section 2.2, the existence of a $W$-NE in an iBG $G$ with DFA goals can be analyzed through solving safety games $G_j$ and testing a Büchi word automaton $A'_W$ for nonemptiness. Note, however, that $A'_W$ had a state space that consisted of the cross product of all DFA goals. Since here we have a strategy profile specified explicitly by the Moore machine $\pi = \langle \pi_0 \dots \pi_{k-1} \rangle$, the Primary-Trace and $j$-Deviant-Trace Conditions can be checked separately for each agent, avoiding a cross-product construction. We begin our analysis by considering agents in $W$.

**Checking Agents in $W$** As mentioned before, we no longer need to create an automaton from the cross product of all DFA goals $A^i$ to form $A_{W,\pi}$, which checks the primary trace for *all* agents at once. Since the primary trace of $\pi$ is uniquely determined (Definition 2.7), we can check whether this trace satisfies the goal $A^i$ for each agent $i \in W$.

For each agent $i \in W$, we construct a DFA $A^i \times \pi$ as the product of the goal $A^i = (Q^i, q^i_0, \Sigma, \delta^i, F^i)$ and $\pi$. In detail, $A^i \times \pi = (Q^i \times S, \langle q^i_0, s_0 \rangle, \emptyset, \tau^i, F^i \times S)$. Note that the alphabet of this automaton is empty, so transitions are defined between states. For $q \in Q^i$ and $s \in S$, We have $\tau^i(\langle q, s \rangle) = \langle q', s' \rangle$, where $q' = \delta^i(q, \gamma(s))$ and $s' = \rho(s, \gamma(s))$. Satisfaction of $A^i$ on the primary trace now corresponds to nonemptiness of this product automaton (as the transition function $\tau^i$ simulates the run of $A^i$ on the primary trace of $\pi$), which means that a state $\langle f, s \rangle$ with $f \in F^i$ is reachable from $\langle q^i_0, s_0 \rangle$. This implies that a prefix of the primary trace is accepted by $A^i$. Note that the state space of $A^i \times \pi$ is of exponential size, since the state space of $\pi$ is of exponential size. Nonemptiness in a DFA with an exponential state space can be decided in NPSPACE=PSPACE.

If $A^i \times \pi$ is empty for some $i \in W$. then $\pi$ is not a $W$-NE as the goal goal $A^i$ for an agent $i$ is not satisfied on the primary trace. We refer to these nonemptiness queries of $A^i \times \pi$ as the $i$-queries.

**Safety Game for Deviating Agents** We now move on to the analyzing the Primary Trace and Deviant Trace conditions for agents $j \notin W$. As before, we first construct a safety game to characterize the set of states from which successful deviations are possible. We adapt the safety game $G_j$ to take in account the fact that we wish to check if $\pi$ is a $W$-NE. Formally, we construct the safety game $G_{\pi,j} = (Q^j \times S, Q^j \times S \times \Sigma, E_{\pi,j})$. Agent 0 owns $Q^j \times S$ and agent 1 owns $Q^j \times S \times \Sigma$. The edge relation $E_{\pi,j}$ is defined as follows:

1. $(\langle q, s \rangle, \langle q, s, \alpha \rangle)) \in E_{\pi,j}$ for $q \in Q^j \setminus F^j$, $s \in S$, and $\alpha = \gamma(s)$.
2. $(\langle q, s, \alpha \rangle, \langle q', s' \rangle) \in E_{\pi,j}$ for $q, q' \in Q^j$ and $s, s' \in S$, where $q' = \delta^j(q, \beta)$ and $s' = \rho(s, \beta)$, for some $\beta \in \Sigma$ such that $\alpha[-j] = \beta[-j]$.

As in $G_j$, if $q \in F^j$, then $\langle q, s \rangle$ has no successor node, and agent 0 is stuck and loses the game. Since $G_{\pi,j}$ is a safety game, agent 0's goal is to avoid states in $F^j$ and not get stuck. Unlike in $G_j$, however, agent 0 has no "discretion" in $G_{\pi,j}$; the move in state ($\langle q, s \rangle$ must be to $\langle q, s, \alpha \rangle$), where $\alpha = \gamma(s)$. Intuitively, we check whether agent 0 can win this game while sticking to the strategy profile $\pi$. By keeping track of the state $s \in S$ of $\pi$, agent 0 must move in accordance with $\gamma(s)$. Therefore, solving the safety game $G_{\pi,j}$ amounts to a reachability query; agent 1 wins if she can reach a state $\langle q, s \rangle$, with $q \in F^j$. Because graph reachability is in NLOGSPACE and the size of the game $G_{\pi,j}$ is exponential in the input due to the exponential state space $S$, the game can be solved in NPSPACE=PSPACE. We denote the set of winning states for agent 0 as $Win_0(G_{\pi,j})$; $Win_1(G_{\pi,j})$ is the set of winning states for agent 1. Note that, in particular, $F^j \times S \subseteq Win_1(G_{\pi,j})$.

**Checking the Agents in $\Omega \setminus W$**

For an agent $j \notin W$, we construct a DFA that checks that the goal $A^j$ is not satisfied on the primary trace or on a deviant trace. In detail, $A^j \times \pi = (Q^j \times S, \langle q^j_0, s_0 \rangle, \emptyset, \tau^j, Win_1(G_{\pi,j}))$ is defined in exactly the same way as the automaton $A^i \times \pi$ for $i \in W$ with the exception of the set of final states. We show below that if $A^j \times \pi$ is nonempty, then either the Primary Trace Condition or the Deviant Trace conditions is violated for Agent $j$.

So, we must make sure that no state in $Win_1(G_{\pi,j})$ is reachable in $A^j \times \pi$ from $\langle q^j_0, s_0 \rangle$. This is equivalent to the automaton being empty, so we have another nonemptiness query but now one that should fail. A path from $\langle q^j_0, s_0 \rangle$ to a state in $Win_1(G_{\pi,j})$ corresponds to either acceptance of $A^j$ on the primary trace or a violation of the $j$-Deviant-Trace Condition , both of which contradicts $\pi$ being a $W$-NE. We refer to these nonemptiness queries as the $j$-queries (a reference to $j \notin W$), and we note that they can be decided in NPSPACE=PSPACE by the exact same logic as $i$-queries. We now prove the correctness of checking the $i$-queries for the agents $i \in W$ and the $j$-queries for the agents $j \notin W$.

**Theorem 6.** *Given an iBG $G$ with DFA goal inputs, a strategy profile $\pi$ is a $W$-NE iff the $i$-queries succeed and the*

*$j$-queries fail.*

*Proof.* ($\rightarrow$). Assume that $\pi$ is a $W$-NE. Therefore, it satisfies both the Primary-Trace Condition and the $j$-Deviant-Trace Condition. Since $\pi$ satisfies the Primary-Trace Condition, there is a path from $\langle q_0^i, s_0\rangle$ to $F^i \times S$ in $A^i \times \pi$, so we have that the $i$-queries are successful.

Suppose now, for contradiction, that some $j$-query succeeds. This means that there is a run of $A^j$ on the primary trace that enters a state $\langle q^j, s\rangle$ in $Win_1(G_{\pi,j})$. There are now two cases:

1. If $q^j \in F^j$, then $A^j$ has just accepted on the primary trace, contradicting the assumption that $\pi$ was a $W$-NE.
2. Otherwise, Agent $j$ now has a winning strategy in $G_{\pi,j}$ from state $\langle q^j, s\rangle$. By following this strategy, the agent is able to reach a state in $F^j \times S$, which means that Agent $j$ constructed a $j$-Deviant-Trace that satisfies $A^j$, violating the $j$-Deviant-Trace Condition of $\pi$ and contradicting the assumption that $\pi$ was a $W$-NE.

Therefore, we have that the $i$-queries must succeed and the $j$-queries must fail.

($\leftarrow$) Assume now that the $i$-queries succeed and the $j$-queries fail. We show that $\pi$ satisfies the Primary-Trace Condition and the $j$-Deviant-Trace Condition.

For the Primary-Trace Condition, note that goals $A^i$, for $i \in W$, accept on the primary trace of $\pi$, since the $i$-queries succeeded. Furthermore, the goals $A^j$, for $j \notin W$, cannot accept on the primary trace of $\pi$, as this would correspond to a path in $A^j \times \pi$ from $\langle q_0^j, s_0\rangle$ to $F^j \times S \subseteq Win_1(G_{\pi,j})$ in $A^j \times \pi$. No such path exists since the $j$-queries failed.

For the $j$-Deviant-Trace Condition, we only need to study the $j$-queries, $j \notin W$. Note that $A^j \times \pi$ cannot enter a state in $Win_1(G_{\pi,j})$, since the $j$-queries failed. Thus, $A^j \times \pi$ stays in $Win_0(G_{\pi,j})$. A $j$-Deviant-Trace must separate from the primary trace at some time step $k \geq 0$, since deviant traces cannot be the primary trace, so at that point $A^j \times \pi$ is in some state $\langle q, s\rangle \in Win_0(G_{\pi,j})$. For $A^j$ to accept on a deviant trace means that agent 1 can force reaching, in the games $G_{\pi,j}$, from $\langle q, s\rangle$ to some $\langle q', s'\rangle$ for $q \in F^j$ and $s \in S$. But that is not possible, since it would mean that $\langle q, s\rangle \in Win_1(G_{\pi,j})$. It follows that $A^j$ cannot accept on a $j$-deviant trace. □

**Complexity** As noted before, each safety game and reachability query can be solved in PSPACE. Therefore, the entire algorithm has a PSPACE upper bound. The same holds for the bounded-channel case (for both goal automata and Moore machines), as the safety games and reachability queries would still be solved in PSPACE since $S$ would still be exponential in the size of the input.

**Theorem 7.** *The verification problem with DFA goals can be solved in PSPACE.*

## 4.2 NFA and AFA Goals

**NFA Goals** The algorithm for NFA goals follows from the algorithm for DFA goals with some adaptation. Since we are dealing with nondeterministic automata now, we denote the transition function $\delta^i$ of the goal automaton $A^i$ as a set of triples with $\langle q, \alpha, q'\rangle$ belonging to $\delta^i$ if it possible to transition from state $q$ to state $q'$ upon reading $\alpha \in \Sigma$. As before, we start by considering the agents in $W$.

**Checking Agents in $W$** Given a goal automaton $A^i = (Q^i, q_0^i, \Sigma, \delta^i, F^i)$ we use essentially the same construction of $A^i \times \pi = (Q^i \times S, \langle q_0^i, s_0\rangle, \emptyset, \tau^i, F^i \times S)$, which is now a nondeterministic finite automaton. The transition function $\tau^i$ is modified slightly to accommodate nondeterministic transitions. As before, the alphabet of this automaton is empty so transitions are defined between states; therefore $\tau^i$ is represented a set of pairs. For $q \in Q^i$ and $s \in S$, We have $\langle\langle q, s\rangle, \langle q', s'\rangle\rangle \in \tau^i$ if $\langle q, \gamma(s), q'\rangle \in \delta^i$ and $s' = \rho(s, \gamma(s))$. Once again we test these automata for nonemptiness, noting that a word accepted by $A^i \times \pi$ corresponds to $A^i$ accepting a finite prefix of the primary trace of $\pi$. As before, we denote these nonemptiness queries as the i-queries. They can once again be tested for nonemptiness in NPSPACE=PSPACE, as they are once again equivalent to reachability testing in an exponentially large graph (caused by the exponential state space $S$).

As before, we proceed with the construction of safety games to analyze the set of states from which deviation is possible for an agent $j \notin W$.

**Safety Game for Deviating Agents** We construct the safety game $G_{\pi,j} = (Q^j \times S, Q^j \times S \times \Sigma, E_{\pi,j})$. agent 0 owns $Q^j \times S$, and agent 1 owns $Q^j \times S \times \Sigma$. The edge relation $E_{\pi,j}$ is defined as follows:

1. $(\langle q, s\rangle, \langle q, s, \alpha\rangle)) \in E_{\pi,j}$ for $q \in Q^j \setminus F^j$, $s \in S$, and $\alpha = \gamma(s)$.
2. $(\langle q, s, \alpha\rangle, \langle q', s'\rangle) \in E_{\pi,j}$ for $q, q' \in Q^j$ and $s, s' \in S$, if $\langle q, \beta, q'\rangle \in \delta^j$ and $s' = \rho(s, \beta)$, for some $\beta \in \Sigma$ such that $\alpha[-j] = \beta[-j]$.

This is a slight modification from the previous construction that takes into account that there are now multiple transitions possible for a state $q$ and a letter $\beta \in \Sigma$ in $A^j$, so the fundamental structure of the game is unchanged. It still amounts to a reachability query, as agent 0 still has no choice in moves. As before, these safety games are exponential in the size of the input due to the presence of $S$, and therefore they can be solved in NPSPACE=PSPACE. We retain the notation that the set of winning states for agent 0 is given by $Win_0(G_{\pi,j})$ with $Win_1(G_{\pi,j})$ defined analogously.

**Checking the Agents in $\Omega \setminus W$** Once again, the same argument from before applies. We create the NFA $A^j \times \pi = (Q^j \times S, \langle q_0^j, s_0\rangle, \emptyset, \tau^j, Win_1(G_{\pi,j}))$ from the goal NFA $A^j$, which differs from the previous construction of the NFA $A^i \times \pi$ in only the set of final states. As before, in the DFA case, we denote nonemptiness queries of $A^j \times \pi$ as the j-queries and they can once again be conducted in NPSPACE=PSPACE. It is once again integral to $\pi$ being a $W$-NE that the j-queries fail. We state an equivalent theorem to Theorem 6 for NFA inputs.

**Theorem 8.** *Given an iBG $G$ with NFA goal inputs, a strategy profile $\pi$ is a $W$-NE iff the $i$-reachability queries succeed and the $j$-reachability queries fail.*

*Proof.* The proof of this theorem closely follows the proof of Theorem 6 and is therefore omitted. □

**Complexity** Each safety game and reachability query was conducted in PSPACE. Therefore, the entire algorithm has a PSPACE upper bound. Once again, $S$ is exponential in the size of the input for even the bounded-channel case, so the result holds for the bounded-channel case as well.

**Theorem 9.** *The verification problem with NFA goals can be solved in PSPACE.*

We note that we can achieve the same upper bound by simply determinizing each goal automaton and then applying the DFA verification procedure. While both a DFA-based approach and the approach outlined in this section lie in PSPACE, the latter has better complexity in practice since it avoids a second exponential blowup. The approach in this section also prepares us to handle AFA goals.

**AFA Goals** A similar version of this problem in which a game $G$ with $LDL_f$ goal specifications was queried to see if some $W$-NE existed was presented in (Gutierrez, Perelli, and Wooldridge 2017) and was proven PSPACE-complete. In this section, we show that this upper bound also holds when $W$ is specified and we are given AFA goals.

With AFA goals, we have a choice of converting to NFAs, incurring an exponential blowup, or DFAs, incurring a doubly exponential blowup. By converting to NFAs, we can avoid a second exponential blow up and show that this problem lies in PSPACE. Therefore given goal AFAs, we create equivalent NFAs $A^i$ from the input and proceed as before.

The safety games constructed for NFA goals had a state space of $\{Q^j \times S\} \cup \{Q^j \times S \times \Sigma\}$. Since we converted from an AFA to NFA to obtain $Q^j$, $Q^j$ is now exponential in the size of the input. Since $S$ remains exponential and $\Sigma$ was a part of the input, this game remains exponential in the size of the input. Therefore, these safety games can still be solved in PSPACE.

With respect to the automata constructed for the reachability queries, each vertex space $Q^i \times S$ still remains exponential in the size of the input even when $Q^i$ is exponential in the size of the input. Therefore, these reachability queries can also be solved in PSPACE.

**Theorem 10.** *The verification problem with AFA goals can solved in PSPACE.*

Note that by employing an approach in which we check each agent individually, we have also avoided a situation in which we must take the cross product of exponentially large automata. If we are given $k$ AFAs and wish to convert them in $k$ NFAs, this represents an exponential blowup. At this point, we could take the cross product of all NFAs and still remain exponential in the size of the input. Note, however, that this would involve a quadratic blowup in the exponent - while $2^n$ and $2^{kn}$ are both exponential in $n$, there is an exponential (in $k$) gap between $2^n$ and $2^{kn}$.

## 4.3 Lower Bound

We now prove PSPACE-hardness for the verification problem with DFA goals, which also serves as a lower bound for the verification problems with NFA and AFA goals.

We use the succinct representations of bounded-channel automata to show that the verification problem for DFA goals is PSPACE-hard through a polynomial time reduction from the following canonical PSPACE-complete problem : given a deterministic Turing machine $M$ and a natural number $n$ in unary, does $M$ accept the empty tape using at most $n$ space (Sipser 2006)? We further assume that $M$ has a unique accepting configuration in which the tape consists solely of a special unused character $*$ with the head on the rightmost of the $n$ cells. This standard assumption does not influence the complexity of the problem.

The Turing Machine $M$ has a state set denoted by $R$ and an alphabet denoted by $\Delta$. Our reduction relies on the notion of an instantaneous description (ID) of a Turing Machine, which is a string that represents the content of the tape at a discrete time step in the run time of $M$. Such an ID includes

- The complete contents of the tape from left to right.
- The position and state of the head of $M$. As a matter of notation, if the head is on cell $i$ then the character corresponding to the content of cell $i$ is a pair consisting of the the element of $\Delta$ on the tape *and* an element of $R$ representing the state of the machine.

As an example, an ID could be of the form $121\langle 0, q\rangle 31$. In this case, the content of the tape is $121031$, while the pair $\langle 0, q\rangle$ denotes that the machine is currently reading the cell with symbol $0$ while in state $q$. Since the machine is deterministic, a sequence of IDs corresponding to the computation run of $M$ on the empty tape is uniquely given by the initial state and position of the head of $M$, which we will call $ID_0$. The machine then accepts if there is a sequence of IDs $ID_0 \dots ID_m$ such that $ID_m$ is the unique accepting configuration of $M$ and $ID_{i+1}$ follows from $ID_i$ according to the transition function of $M$. Our reduction strategy to the verification problem is to use a set of bounded-channel Moore Machines to simulate the transitions from one $ID_i$ to $ID_{i+1}$, and a set bounded channel DFA goals of the agents to verify that the sequence is correct - that it starts at $ID_0$ and eventually reaches $ID_m$.

We now sketch a construction of a game $G_{M,n}$ and a strategy profile $\pi_{M,n} = \langle \pi_0 \dots \pi_{n-1} \rangle$ such that the Turing machine $M$ accepts the empty tape in at most $n$ steps iff $\pi$ is an $\Omega$-NE in $G$, i.e. a strategy profile that satisfies every agent's goal on its primary trace. The number of agents in this game is given by $n$, the same as the length of the unary input to the Turing Machine $M$.

We first consider the $\Sigma_i$ assigned to each agent in $G$. This alphabet is $R \cup \{\Delta \times R\}$ for every agent. Intuitively, each $\Sigma_i$ represents a single character of the ID, with $\Sigma_i$ specifically corresponding to the $i$-th cell of the ID. Therefore taking all $n$ alphabets together as $\Sigma$ corresponds to an entire ID.

The strategy $\pi_i$ of an agent $i$ outputs the next configuration of cell $i$ based on the previous configurations of the cell to the right, the cell to the left, and the cell itself. Thus, each strategy only needs to read at most three symbols, from

$\Sigma_{i-1}, \Sigma_i$, and $\Sigma_{i+1}$–since transitions in a Turing machine are determined locally–and output a symbol in $\Sigma_i$ according to the transition function of $M$. (Agents 0 and $n-1$ need only consider two of the $\Sigma_i$ since there are no cells to the left and right of these agents, respectively). If the computation moves out of bounds, then the Moore machine for the agent that moved it out of bounds (either 0 or $n-1$) immediately moves to a sink state that continuously outputs a special character that does not appear in the unique accepting configuration. Since each strategy only considers at most three $\Sigma_i$'s as input and outputs a single $\Sigma_i$, these strategies can be represented by bounded-channel Moore machines. The state space of each machine has a size upper bound of $|R \cup \{\Delta \times R\}|^3$, meaning that each machine is polynomial in the size of the input.

We now consider the goals for the agents. The purpose of the goals is to check that the initial ID represent the empty tape, and the final ID is an accepting one. The goal for each agent except Agents 0 and $n-1$ is to see the symbol $*$ some time after reading the empty symbol $\perp$ as the first symbol corresponding to the empty tape input. The goal for Agent 0 is to eventually read the symbol $*$, after reading the first symbol $\langle \perp, q_0 \rangle$, where $q_0 \in R$ is the initial state of $M$. The goal for Agent $n-1$ is to read the pair $\langle *, q_F \rangle$ some time after seeing an empty cell as the first character, where $q_F \in R$ is the unique state corresponding to the accepting configuration $ID_m$ since the head is moved all the way to the right in $ID_m$. The DFA representations of these goals are very simple, as they solely consist of eventually reading a single character after verifying an initial character. Therefore, all goal DFAs are bounded-channel automata and are therefore polynomial in the size of the input. Overall, we have a polynomial number of agents, each with a polynomial-sized Moore machine and a polynomial-sized goal DFA. Therefore, the game $G$ can be constructed in polynomial time. We now prove the correctness of the reduction.

**Theorem 11.** *The strategy profile $\pi_{M,n} = \langle \pi_0 \dots \pi_{n-1} \rangle$ is an $\Omega$-NE in $G_{M,n}$ iff $M$ accepts the empty tape using at most space $n$.*

*Proof.* ($\rightarrow$) Assume that $\pi_{M,n}$ is an $\Omega$-NE. By construction, the primary trace of $\pi_{M,n}$ simulates the sequence of IDs of $M$ running on the empty tape with a built in check to ensure that the computation uses no more than $n$ cells. Therefore, for the DFA goals to accept on this trace it means that in the final ID all cells except the last are filled with the special $*$ character and the last has the pair $\langle *, q_F \rangle$ and that each cell started empty. This means that there is a valid sequence of IDs generated by $M$ upon reading the empty tape, meaning that $M$ accepted the empty tape while staying in bounds.

($\leftarrow$)Assume that $M$ accepts the empty tape using no more than $n$ space. Then, it generates a unique valid sequence of IDs that eventually end at the unique accepting configuration. By construction, the primary trace of $\pi$ consists of this same sequence of IDs, and $ID_m$ must consist of the unique configuration consisting of $*$ on every cell but the rightmost with the pair $\langle *, q_F \rangle$. Therefore, all DFA goals will accept on the primary trace of $\pi_{M,n}$, so $\pi_{M,n}$ is an $\Omega$-NE. □

We are able to construct the game $G_{M,n}$ and the profile $\pi_{M,n}$ in polynomial time due to the succinct representation of bounded-channel automata. Therefore, we have exhibited a polynomial time reduction from a known PSPACE-complete problem for the verification problem with DFA goal inputs. Therefore, we have:

**Theorem 12.** *The verification problems with DFA goals is PSPACE-complete.*

Since DFAs are a special case of both NFAs and AFAs, we get lower bounds for both corresponding verification problems as well.

**Corollary 12.1.** *The verification problems with NFA or AFA goals are PSPACE-complete.*

## 5 Concluding Remarks

In this work we provided complexity results for both the realizability and verification problems in the finite-horizon multiagent setting with different types of goal specifications, significantly extending previous works (Gutierrez, Perelli, and Wooldridge 2017; Rajasekaran and Vardi 2021). One of the key points of interest from this analysis is the complexity gap observed between the complexities of the realizability and verification problems. While realizability with DFA goals was proven to be PSPACE-complete in (Rajasekaran and Vardi 2021), here we have shown that realizability with NFA goals is EXPTIME-complete and realizability with AFA goals is 2EXPTIME-complete. Therefore, with respect to the realizability problem, we have shown that the succincness of goal specification greatly influences the complexity of the realizability problem. With respect to the verification problem, however, this distinction does not exist, as the verification problems with DFA,NFA, and AFA goals are all PSPACE-complete. Thus, the complexity for DFA goals is the same for both the realizability and the verification problems, but as the automata get more succinct the realizability problem grows in complexity while the verification problem remains PSPACE-complete. This complexity picture is similar to what is known in temporal reasoning in two-agent systems (system and environment), where the complexity of realizability rises from PTIME for DFA goals to 2EXPTIME for $LTL_f$ goals, while verification, i.e., model checking, is PSPACE-complete for different types of goals, with the system *state-explosion problem* being the primary source of PSPACE-hardness, cf. (Vardi 1996)

Finally, by reasoning about the Primary-Trace Condition and the $j$-Deviant-Trace Conditions separately, as in (Rajasekaran and Vardi 2021), we were able to get algorithms that are easy to understand and optimal. This method of separation was made specifically to reason about Nash equilibria in qualitative games, by leveraging properties of both the Nash equilibria as a solution concept and qualitative goals themselves. By analyzing the $j$-Deviant-Trace Condition separately through the use of safety games we are able to get much better complexity bounds than if we dealt with the entire Nash equilibria at once. We believe that this principle of separation provides a powerful framework to reason about other qualitative multi-agent systems and perhaps even other solution concepts.

## Acknowledgements

Work supported in part by NSF grants IIS-1527668, CCF-1704883, IIS-1830549, CNS-2016656, DoD MURI grant N00014-20-1-2787, and an award from the Maryland Procurement Office.

## References

Abate, A.; Gutierrez, J.; Hammond, L.; Harrenstein, P.; Kwiatkowska, M.; Najib, M.; Perelli, G.; Steeples, T.; and Wooldridge, M. J. 2021. Rational verification: game-theoretic verification of multi-agent systems. *Appl. Intell.* 51(9):6569–6584.

Bouyer, P.; Brenguier, R.; Markey, N.; and Ummels, M. 2015. Pure Nash equilibria in concurrent deterministic games. *Log. Methods Comput. Sci.* 11(2).

Clarke, E.; Emerson, E.; and Sistla, A. 1986. Automatic verification of finite-state concurrent systems using temporal logic specifications. *ACM Transactions on Programming Languages and Systems* 8(2):244–263.

Fisman, D.; Kupferman, O.; and Lustig, Y. 2010. Rational synthesis. In *Proc. 16th Int'l Conf. on Tools and Algorithms for the Construction and Analysis of Systems*, volume 6015 of *Lecture Notes in Computer Science*, 190–204. Springer.

Giacomo, G. D., and Vardi, M. Y. 2013. Linear temporal logic and linear dynamic logic on finite traces. In *Proc. 23rd Int'l Joint Conf. on Artificial Intelligence*, 854–860. IJCAI/AAAI.

Giacomo, G. D., and Vardi, M. Y. 2015. Synthesis for LTL and LDL on finite traces. In Yang, Q., and Wooldridge, M. J., eds., *Proc. 24th Int'l Joint Conf. on Artificial Intelligence*, 1558–1564. AAAI Press.

Giacomo, G. D., and Vardi, M. Y. 2016. $\text{LTL}_\text{f}$ and $\text{LDL}_\text{f}$ synthesis under partial observability. In *Proc. 25th Int'l Joint Conf. on Artificial Intelligence*, 1044–1050. IJCAI/AAAI Press.

Grädel, E.; Thomas, W.; and Wilke, T. 2002. *Automata, Logics, and Infinite Games: A Guide to Current Research*. Lecture Notes in Computer Science 2500. Springer.

Gutierrez, J.; Najib, M.; Perelli, G.; and Wooldridge, M. J. 2020. Automated temporal equilibrium analysis: Verification and synthesis of multi-player games. *Artif. Intell.* 287:103353.

Gutierrez, J.; Harrenstein, P.; and Wooldridge, M. J. 2015a. Expresiveness and complexity results for strategic reasoning. In *Proc. 26th Int'l Conf. on Concurrency Theory*, volume 42 of *LIPIcs*, 268–282. Schloss Dagstuhl - Leibniz-Zentrum für Informatik.

Gutierrez, J.; Harrenstein, P.; and Wooldridge, M. J. 2015b. Iterated Boolean games. *Inf. Comput.* 242:53–79.

Gutierrez, J.; Perelli, G.; and Wooldridge, M. J. 2017. Iterated games with LDL goals over finite traces. In *Proc. 16th Conf. on Autonomous Agents and MultiAgent Systems*, 696–704. ACM.

Kupferman, O., and Vardi, M. 1996. Module checking. In *Proc. 8th Int. Conf. on Computer Aided Verification*, volume 1102 of *Lecture Notes in Computer Science*, 75–86. Springer.

Kupferman, O.; Perelli, G.; and Vardi, M. Y. 2016. Synthesis with rational environments. *Ann. Math. Artif. Intell.* 78(1):3–20.

McNaughton, R. 1993. Infinite games played on finite graphs. *Ann. Pure Appl. Logic* 65(2):149–184.

Mogavero, F.; Murano, A.; Perelli, G.; and Vardi, M. Y. 2014. Reasoning about strategies: On the model-checking problem. *ACM Trans. Comput. Log.* 15(4):34:1–34:47.

Nash, J. F. 1950. Equilibrium points in n-person games. *Proceedings of the National Academy of Sciences* 36(1):48–49.

Pnueli, A., and Rosner, R. 1989. On the synthesis of a reactive module. In *Proc. 16th ACM Symp. on Principles of Programming Languages*, 179–190.

Rajasekaran, S., and Vardi, M. Y. 2021. Nash equilibria in finite-horizon multiagent concurrent games. In *Proc. 20th Int'l Conf. on Autonomous Agents and MultiAgent Systems*, 1046–1054. Int'l Found. for Autonomous Agents and Multiagent Systems.

Rajasekaran, S., and Vardi, M. Y. 2022. Verification and realizability in finite-horizon multiagent systems.

Rosner, R. 1992. *Modular Synthesis of Reactive Systems*. Ph.D. Dissertation, Weizmann Institute of Science.

Shoham, Y., and Leyton-Brown, K. 2009. *Multiagent Systems - Algorithmic, Game-Theoretic, and Logical Foundations*. Cambridge University Press.

Sipser, M. 2006. *Introduction to the Theory of Computation*. Course Technology, second edition.

Vardi, M., and Wolper, P. 1994. Reasoning about infinite computations. *Information and Computation* 115(1):1–37.

Vardi, M. 1996. An automata-theoretic approach to linear temporal logic. In *Logics for Concurrency: Structure versus Automata*, volume 1043 of *Lecture Notes in Computer Science*, 238–266. Springer.

Wooldridge, M. J. 2009. *An Introduction to MultiAgent Systems, Second Edition*. Wiley.