Assume-Guarantee Verification of Strategic Ability

Łukasz Mikulski[1,2], Wojciech Jamroga[2,3], and Damian Kurpiewski[2,1]

[1] Faculty of Mathematics and Computer Science, Nicolaus Copernicus University, Toruń, Poland
[2] Institute of Computer Science, Polish Academy of Sciences, Warsaw, Poland
[3] Interdisciplinary Centre for Security, Reliability and Trust, SnT, University of Luxembourg, Luxembourg

Abstract. Model checking of strategic abilities is a notoriously hard problem, even more so in the realistic case of agents with imperfect information. Assume-guarantee reasoning can be of great help here, providing a way to decompose the complex problem into a small set of exponentially easier subproblems. In this paper, we propose two schemes for assume-guarantee verification of alternating-time temporal logic with imperfect information. We prove the soundness of both schemes, and discuss their completeness. We illustrate the method by examples based on known benchmarks, and show experimental results that demonstrate the practical benefits of the approach.

Keywords: model checking, assume-guarantee reasoning, strategic ability

1 Introduction

Multi-agent systems involve a complex network of social and technological components. Such components often exhibit self-interested, goal-directed behavior, which makes it harder to predict and analyze the dynamics of the system. In consequence, formal specification and automated verification can be of significant help.

Verification of strategic ability. Many important properties of multi-agent systems refer to *strategic abilities* of agents and their groups. *Alternating-time temporal logic* $\mathbf{ATL}^*$ [2,37] and *Strategy Logic* $\mathbf{SL}$ [34] provide powerful tools to reason about such aspects of MAS. For example, the $\mathbf{ATL}^*$ formula $\langle\langle taxi\rangle\rangle \mathrm{G}\, \neg\mathsf{fatality}$ expresses that the autonomous cab can drive in such a way that no one gets ever killed. Similarly, $\langle\langle taxi, passg\rangle\rangle \mathrm{F}\, \mathsf{destination}$ says that the cab and the passenger have a joint strategy to arrive at the destination, no matter what the other agents do. Specifications in agent logics can be used as input to algorithms and tools for *model checking*, that have been in constant development for over 20 years [3,6,7,20,27,30].

Model checking of strategic abilities is hard, both theoretically and in practice. First, it suffers from the well-known state/transition-space explosion. Moreover, the space of possible strategies is at least exponential *on top of the state-space explosion*, and incremental synthesis of strategies is not possible in general – especially in the realistic case of agents with partial observability. Even for the more restricted (and computation-friendly) logic $\mathbf{ATL}$, model checking of its imperfect information variants is $\mathbf{\Delta_2^P}$- to $\mathbf{PSPACE}$-complete for agents playing memoryless strategies [5,37] and **EXPTIME**-complete to undecidable for agents with perfect recall [12,16]. The theoretical results concur with outcomes of empirical studies on benchmarks [6,22,30], as well as recent attempts at verification of real-life multi-agent scenarios [21,26].

Contribution. In this paper, we make the first step towards compositional model checking of strategic properties in asynchronous multi-agent systems with imperfect information. The idea of *assume-guarantee reasoning* [10,36] is to "factorize" the verification task into subtasks where components are verified against a suitable abstraction of the rest of the system. Thus, instead of searching through the states (and, in our case, strategies) of the huge product of all components, most of the search is performed locally.

To achieve this, we adapt and extend the assume-guarantee framework of [31,32]. We redefine the concepts of modules and their composition, follow the idea of expressing assumptions as Büchi automata, and accordingly redefine their interaction with the computations of the coalition. Then, we propose two alternative assume-guarantee schemes for $\mathbf{ATL}^*$ with imperfect information. The first, simpler one is shown to be sound but incomplete. The more complex one turns out to be both sound and complete. We illustrate the properties of the schemes on a variant of the Trains, Gate and Controller scenario [4], and evaluate the practical gains through verification experiments on models of logistic robots, inspired by [26].

Note that our formal treatment of temporal properties, together with strategic properties of curtailment,[4] substantially extends the applicability of schemes in [31,32] from temporal liveness properties to strategic properties with arbitrary **LTL** objectives. We also emphasize that our schemes are sound for the model checking of agents with *imperfect* as well as *perfect recall*. In consequence, they can be used to facilitate verification problems with a high degree of hardness, including the undecidable variant for coalitions of agents with memory. In that case, the undecidable problem reduces to multiple instances of the **EXPTIME**-complete verification of individual abilities.

Structure of the paper. In Section 2, we present the model of concurrent MAS that we consider in this paper. In Section 3, we define the syntax and semantics of the logic used in the formulation of agents' strategic properties. In Sections 4 and 5 we introduce the notions of assumption and guarantee, and utilize them to propose two schemes of assume-guarantee reasoning for strategic abilities. Finally, we present preliminary results of experimental verification in Section 6 and conclude the paper in Section 7.

Related Work. Compositional verification (known as *rely-guarantee* in the program verification community) dates back to the early 1970s and the works of Hoare, Owicki, Gries and Jones [19,24,35]. Assume-guarantee reasoning for temporal specifications was introduced a decade later [10,36], and has been in development since that time [11,13,18,29,31,32]. Moreover, automated synthesis of assumptions for temporal reasoning has been studied in [9,15,17,25].

The works that come closest to our proposal are [11,14,31,32]. In [31,32], models and a reasoning scheme are defined for assume-guarantee verification of liveness properties in distributed systems. We build directly on that approach and extend it to the verification of strategic abilities. [11] studies assume-guarantee reasoning for an early version of **ATL**. However, their assume-guarantee rules are designed for perfect infor-

[4] Provided in the supplementary material, available at https://github.com/agrprima22/sup.

mation strategies (whereas we tackle the more complex case of imperfect information), and targeted specifically the verification of aspect-oriented programs. Finally, [14] investigates the compositional synthesis of strategies for **LTL** objectives. The difference to our work is that they focus on finite-memory strategies while we consider the semantics of ability based on memoryless and perfect recall strategies. Another difference lies in our use of repertoire functions that define agents' choices in a flexible way, and make it closer to real applications. The advantage of the solution presented in [14] is the use of contracts, thanks to which it is possible to synthesize individual strategies using the knowledge of the coalition partners' strategies. We also mention [8] that studies the synthesis of Nash equilibrium strategies for 2-player coalitions pursuing ω-regular objectives. The authors call their approach *assume-guarantee strategy synthesis*, but the connection to assume-guarantee verification is rather loose.

A preliminary version of the ideas, presented here, was published in the extended abstract [33]. Our extension of the STV tool [27], used in the experiments, is described in the companion paper [28].

2 Models of Concurrent MAS

Asynchronous MAS have been modeled by variants of reactive modules [1,32] and automata networks [23]. Here, we adapt the variant of reactive modules that was used to define assume-guarantee verification for temporal properties in [32].

2.1 Modules

Let D be the shared domain of values for all the variables in the system. D^X is the set of all valuations for a set of variables X. The *system* consists of a number of *agents*, each represented by its *module* and a *repertoire* of available choices. Every agent uses *state variables* and *input variables*. It can read and modify its state variables at any moment, and their valuation is determined by the current state of the agent. The input variables are not a part of the state, but their values influence transitions that can be executed.

Definition 1 (Module [32]). *A* module *is a tuple $M = (X, I, Q, T, \lambda, q_0)$, where: X is a finite set of state variables; I is a finite set of input variables with $X \cap I = \varnothing$; $Q = \{q_0, q_1, \ldots, q_n\}$ is a finite set of states; $q_0 \in Q$ is an initial state; $\lambda : Q \to D^X$ labels each state with a valuation of the state variables; finally, $T \subseteq Q \times D^I \times Q$ is a transition relation such that (a) for each pair $(q, \alpha) \in Q \times D^I$ there exists $q' \in Q$ with $(q, \alpha, q') \in T$, and (b) $(q, \alpha, q') \in T, q \neq q'$ implies $(q, \alpha, q) \notin T$. In what follows, we omit the self-loops from the presentation.*

Modules M, M' are *asynchronous* if $X \cap X' = \varnothing$. We extend modules by adding *repertoire functions* that define the agents' available choices in a way similar to [23].

Definition 2 (Repertoire). *Let $M = (X, I, Q, T, \lambda, q_0)$ be a module of agent i. The* repertoire *of i is defined as $R : Q \to \mathcal{P}(\mathcal{P}(T))$, i.e., a mapping from local states to* sets of sets of transitions. *Each $R(q) = \{T_1, \ldots, T_m\}$ must be nonempty and consist of nonempty sets T_i of transitions starting in q. If the agent chooses $T_i \in R(q)$, then only a transition in T_i can be occur at q within the module.*

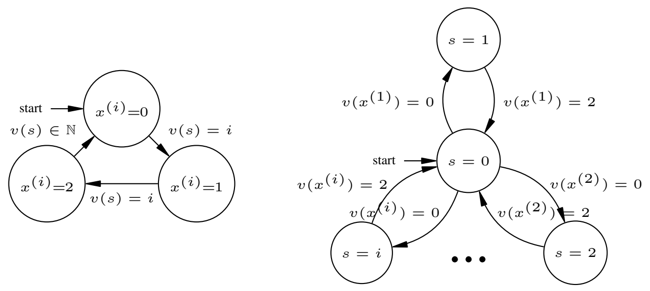


Fig. 1. A variant of TCG: Train synchronizing with a semaphore (left) and the controller (right).

We adapt the Train-Gate-Controller (TGC) benchmark [3] as our running example.

Example 1. The module $M^{(i)}$ of a train is presented in Figure 1 (left). Its local states $Q^{(i)} = \{w^{(i)}, t^{(i)}, a^{(i)}\}$ refer, respectively, to the train **w**aiting at the entrance, riding in the **t**unnel, and cruising **a**way from the tunnel. The sole state variable $x^{(i)}$ labels the state with values 0, 1, and 2, respectively. $I^{(i)} = \{s\}$ consists of a single input variable that takes values from an external multi-valued semaphore. The train can enter and exit the tunnel only if the semaphore allows for that, i.e., if $v(s) = i$. To this end, we define $T^{(i)} = \{(w^{(i)}, i, t^{(i)}), (t^{(i)}, i, a^{(i)}), (a^{(i)}, 0, w^{(i)}), (a^{(i)}, 1, w^{(i)}) \ldots, (a^{(i)}, n, w^{(i)})\} \cup \{(w^{(i)}, j, w^{(i)}), (t^{(i)}, j, t^{(i)}) \mid j \neq i\}$.[5]

The module $M^{(C(n))}$ of a controller that coordinates up to n trains is depicted in Figure 1 (right). Formally, it is defined by:

- $X = \{s\}$ (the semaphore),
- $I = \{x_1, \ldots, x_n\}$ (the positions of trains),
- $Q = \{r, g_1, \ldots, g_n\}$ (red or directed green light),

where a state with subscript 1 represents a tunnel shared with the other trains, $\lambda(g_i)(s) = i$, $\lambda(r)(s) = 0$, and r is the initial state.

The controller can change the light to green when a train is waiting for the permission to enter the tunnel, and back to red after it passed through the tunnel: $T = \{(r, v, g_i) \mid v(x_i) = 0\} \cup \{(g_i, v, r) \mid v(x_i) = 2\}$.

Each agent can freely choose the local transition intended to execute next. Thus, $R^{(i)}(q) = \{\{(q, \alpha, q')\} \mid (q, \alpha, q') \in T^{(i)}\}$, and similarly for $R^{(C(n))}$.

Note that all the modules in TCG are asynchronous.

2.2 Composition of Agents

On the level of the temporal structure, the model of a multi-agent system is given by the asynchronous composition $M = M^{(1)}| \ldots |M^{(n)}$ that combines modules $M^{(i)}$ into a single module. The definition is almost the same as in [32]; we only extend it to handle the repertoire functions that are needed to characterize strategies and strategic abilities.

[5] By a slight abuse of notation, the valuation of a single variable is identified with its value.

We begin with the notion of compatible valuations to adjust local states of one agent with the labels of the actions performed by the other agent. Note that the local states of different asynchronous agents rely on disjoint sets of variables.

Let $Y, Z \subseteq X$ and $\rho_1 \in D^Y$ while $\rho_2 \in D^Z$. We say that ρ_1 is compatible with ρ_2 (denoted by $\rho_1 \sim \rho_2$) if for any $x \in Y \cap Z$ we have $\rho_1(x) = \rho_2(x)$. We can compute the union of ρ_1 with ρ_2 which is compatible with ρ_1 by setting $(\rho_1 \cup \rho_2)(x) = \rho_1(x)$ for $x \in Y$ and $(\rho_1 \cup \rho_2)(x) = \rho_2(x)$ for $x \in Z$.

Definition 3 (Composition of modules [32]). *The composition of asynchronous modules $M^{(1)} = (X^{(1)}, I^{(1)}, Q^{(1)}, T^{(1)}, \lambda^{(1)}, q_0^{(1)})$ and $M^{(2)} = (X^{(2)}, I^{(2)}, Q^{(2)}, T^{(2)}, \lambda^{(2)}, q_0^{(2)})$ (with $X^{(1)} \cap X^{(2)} = \varnothing$) is a composite module $M = (X = X^{(1)} \uplus X^{(2)}, I = (I^{(1)} \cup I^{(2)}) \setminus X, Q^{(1)} \times Q^{(2)}, T, \lambda, q_0 = (q_0^{(1)}, q_0^{(2)}))$, where*

- $\lambda : Q^{(1)} \times Q^{(2)} \to D^X$, $\lambda(q^{(1)}, q^{(2)}) = \lambda^{(1)}(q^{(1)}) \cup \lambda^{(2)}(q^{(2)})$,
- *T is the minimal transition relation derived by the set of rules presented below:*

$$\mathbf{ASYN_L}\quad \frac{q^{(1)} \xrightarrow{\alpha^{(1)}}_{T^{(1)}} q'^{(1)} \quad q^{(2)} \xrightarrow{\alpha^{(2)}}_{T^{(2)}} q'^{(2)} \\ \alpha^{(1)} \sim \alpha^{(2)} \quad \lambda^{(1)}(q^{(1)}) \sim \alpha^{(2)} \quad \lambda^{(2)}(q^{(2)}) \sim \alpha^{(1)}}{(q^{(1)}, q^{(2)}) \xrightarrow{(\alpha^{(1)} \cup \alpha^{(2)}) \setminus X}_T (q'^{(1)}, q^{(2)})}$$

$$\mathbf{ASYN_R}\quad \frac{q^{(1)} \xrightarrow{\alpha^{(1)}}_{T^{(1)}} q'^{(1)} \quad q^{(2)} \xrightarrow{\alpha^{(2)}}_{T^{(2)}} q'^{(2)} \\ \alpha^{(1)} \sim \alpha^{(2)} \quad \lambda^{(1)}(q^{(1)}) \sim \alpha^{(2)} \quad \lambda^{(2)}(q^{(2)}) \sim \alpha^{(1)}}{(q^{(1)}, q^{(2)}) \xrightarrow{(\alpha^{(1)} \cup \alpha^{(2)}) \setminus X}_T (q^{(1)}, q'^{(2)})}$$

$$\mathbf{SYN}\quad \frac{q^{(1)} \xrightarrow{\alpha^{(1)}}_{T^{(1)}} q'^{(1)} \quad q^{(2)} \xrightarrow{\alpha^{(2)}}_{T^{(2)}} q'^{(2)} \\ \alpha^{(1)} \sim \alpha^{(2)} \quad \lambda^{(1)}(q^{(1)}) \sim \alpha^{(2)} \quad \lambda^{(2)}(q^{(2)}) \sim \alpha^{(1)}}{(q^{(1)}, q^{(2)}) \xrightarrow{(\alpha^{(1)} \cup \alpha^{(2)}) \setminus X}_T (q'^{(1)}, q'^{(2)})}$$

pruned in order to avoid disallowed self-loops. We use the notation $M = M^{(1)}|M^{(2)}$.

Note that the operation is defined in [32] for a pair of modules only. It can be easily extended to a larger number of pairwise asynchronous modules. Moreover, the order of the composition does not matter.

Consider agents $(M^{(1)}, R^{(1)}),\ \ldots,\ (M^{(n)}, R^{(n)})$. The *multi-agent system* is defined by $\mathcal{S} = (M^{(1)}|M^{(2)}|\ldots|M^{(n)},\ R^{(1)}, \ldots, R^{(n)})$, i.e., the composition of the underlying modules, together with the agents' repertoires of choices.

Example 2. The composition $M^{(1)}|M^{(2)}|M^{(C(2))}$ of two train modules $M^{(1)}, M^{(2)}$ and controller $M^{(C(2))}$ is presented in Figure 2. The asynchronous transitions are labelled by the agent performing the transitions. All the synchronous transitions performed by both trains are in red, while the synchronous transitions performed by a controller with one of the trains are in blue. There are two synchronous transition performed by all the agents, both in green.

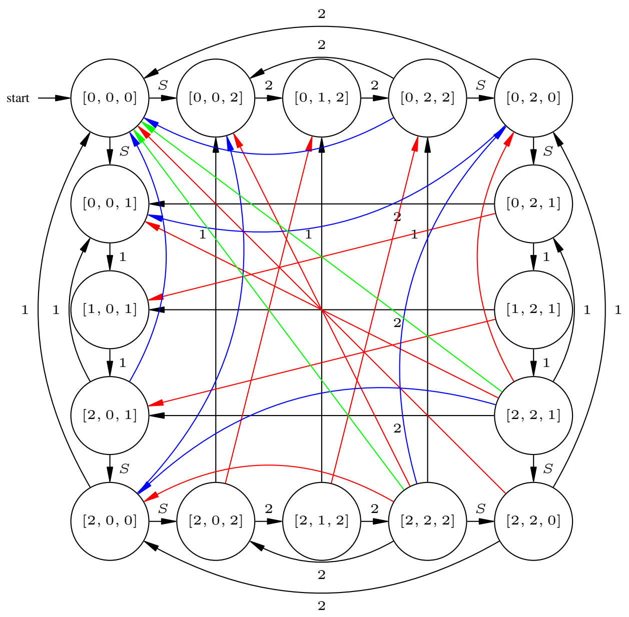


Fig. 2. Composition of modules: two trains $M^{(1)}$, $M^{(2)}$ and controller $M^{(C(2))}$

Traces and Words. A trace of a module M is an infinite sequence of alternating states and transitions $\sigma = q_0\alpha_0q_1\alpha_1\ldots$, where $(q_i, \alpha_i, q_{i+1}) \in T$ for every $i \in \mathbb{N}$ (note that q_0 is the initial state). An infinite word $w = v_0v_1, \ldots \in (D^X)^\omega$ is *derived* by M with trace $\sigma = q_0\alpha_0q_1\alpha_1\ldots$ if $v_i = \lambda(q_i)$ for all $i \in \mathbb{N}$. An infinite word $u = \alpha_0\alpha_1, \ldots \in (D^I)^\omega$ is *admitted* by M with σ if $\sigma = q_0\alpha_0q_1\alpha_1\ldots$. Finally, w (resp. u) is derived (resp. admitted) by M if there exists a trace of M that derives (resp. admits) it.

3 What Agents Can Achieve

Alternating-time temporal logic **ATL*** [2,37] introduces *strategic modalities* $\langle\langle C\rangle\rangle\gamma$, expressing that coalition C can enforce the temporal property γ. We use the semantics based on *imperfect information strategies* with *imperfect recall* (ir) or *perfect recall* (iR) [37]. Moreover, we only consider formulas without the next step operator X due to its questionable interpretation for asynchronous systems, which are based on the notion of local clocks.

Syntax. Formally, the syntax of $\mathbf{ATL}^*_{-\mathbf{X}}$ is as follows:

$$\phi ::= p(Y) \mid \neg\phi \mid \phi \wedge \phi \mid \langle\langle C\rangle\rangle\gamma\,, \qquad\qquad \gamma ::= \phi \mid \neg\gamma \mid \gamma \wedge \gamma \mid \gamma\,\mathrm{U}\,\gamma$$

where $p : Y \to D$ for some subset of domain variables $Y \subseteq X$. That is, each atomic statement refers to the valuation of variables used in the system. U is the "strong until" operator of $\mathbf{LTL}_{-\mathbf{X}}$. The "sometime" and "always" operators can be defined as usual by $\mathrm{F}\,\gamma \equiv \top\,\mathrm{U}\,\gamma$ and $\mathrm{G}\,\gamma \equiv \neg\mathrm{F}\,\neg\gamma$. The set of variables used by the formula γ is denoted by $var(\gamma)$.

In most of the paper, we focus on formulas that consist of a single strategic modality followed by an $\mathbf{LTL}_{-\mathbf{X}}$ formula (i.e., $\langle\!\langle C\rangle\!\rangle\gamma$, where $\gamma \in \mathbf{LTL}_{-\mathbf{X}}$). The corresponding fragment of $\mathbf{ATL}^*_{-\mathbf{X}}$, called $\mathbf{1ATL}^*_{-\mathbf{X}}$, suffices to express many interesting specifications, namely the ones that refer to agents' ability of enforcing trace properties (such as safety or reachability of a winning state). Note that $\mathbf{1ATL}^*_{-\mathbf{X}}$ has strictly higher expressive and distinguishing power than $\mathbf{LTL}_{-\mathbf{X}}$. In fact, model checking $\mathbf{1ATL}^*_{-\mathbf{X}}$ is equivalent to $\mathbf{LTL}_{-\mathbf{X}}$ controller synthesis, i.e., a variant of $\mathbf{LTL}$ realizability.

Nested strategic modalities might be sometimes needed to refer to an agent's ability to endow or deprive another agent with/of ability. We discuss assume-guarantee verification for such specifications in Section 5.4.

Strategies and Their Outcomes. Let $\mathcal{S}$ be a system composed of n agents with asynchronous modules $M^{(i)} = (X^{(i)}, I^{(i)}, Q^{(i)}, T^{(i)}, \lambda^{(i)}, q_0^{(i)})$ and repertoires $R^{(i)}$.

Definition 4 (Strategies). *A* memoryless strategy *for agent i (*ir*-strategy in short) is a function $s_i^{\mathrm{ir}} : Q^{(i)} \to \mathcal{P}(\mathcal{P}(T^{(i)}))$ such that $s_i^{\mathrm{ir}}(q^{(i)}) \in R^{(i)}(q^{(i)})$ for every $q^{(i)} \in Q^{(i)}$. That is, a memoryless strategy assigns a legitimate choice to each local state of i.*

A perfect recall strategy *for i (*iR*-strategy in short) is a function $s_i^{\mathrm{iR}} : (Q^{(i)})^+ \to T^{(i)}$ such that $s_i^{\mathrm{iR}}(q_1^{(i)}, \ldots, q_k^{(i)}) \in R^{(i)}(q_k^{(i)})$, i.e., it assigns choices to finite sequences of local states. We assume that s_i^{iR} is stuttering-invariant, i.e.,*

$$s_i^{\mathrm{iR}}(q_1^{(i)}, \ldots, q_j^{(i)}, q_j^{(i)}, \ldots, q_k^{(i)}) = s_i^{\mathrm{iR}}(q_1^{(i)}, \ldots, q_j^{(i)}, \ldots, q_k^{(i)}).$$

Note that the agent's choices in a strategy depend only on its local *states, thus being uniform by construction.*

Let $\sigma = q_0\alpha_0q_1\alpha_1\ldots$ be a trace, where $q_j = (q_j^{(1)}, q_j^{(2)}, \ldots, q_j^{(n)})$ are global states in $Q^{(1)} \times \ldots \times Q^{(n)}$. We say that σ *implements* strategy s_i^{ir} if, for any j where $q_j^{(i)} \neq q_{j+1}^{(i)}$, we have $(q_j^{(i)}, \alpha_j, q_{j+1}^{(i)}) \in s_i^{\mathrm{ir}}(q_j^{(i)})$ where $\alpha_j : I^{(i)} \to D$ and $\alpha_j(x) = \lambda(q_j)(x)$. A word $w = v_0v_1\ldots$ *implements* s_i^{ir} if it is derived by $\mathcal{S}$ with some trace σ implementing s_i^{ir}. The definitions for s_i^{iR} are analogous.

Definition 5 (Coalitional strategies). *Let $C \subseteq \{1, \ldots, n\}$ be a coalition of agents. A* joint memoryless strategy *s_C^{ir} for C is a collection of memoryless strategies s_i^{ir}, one per $i \in C$. We say that a trace σ (respectively a word w_σ)* implements *s_C^{ir} if it implements every strategy $s_i^{\mathrm{ir}}, i \in C$. The definitions for joint perfect recall strategies are analogous. Whenever a claim holds for both types of strategies, we will refer to them simply as "strategies."*

Semantics. Let $x \in \{\mathrm{ir}, \mathrm{iR}\}$ be a strategy type. The semantics of $\mathbf{ATL}^*_{-\mathbf{X}}$ is given below (we omit the standard clauses for Boolean operators etc.). By $w[i]$, we denote the ith item of sequence w, starting from 0.

$\mathcal{S}, q \models_x p(Y)$ if $\lambda(q)|_Y = p(Y)$;

$\mathcal{S}, q \models_x \langle\langle C\rangle\rangle\gamma$ if there exists an x-strategy s_C for C such that, for any word w starting in q that implements s_C, we have $\mathcal{S}, w \models \gamma$;
$\mathcal{S}, w \models \phi$ if $\mathcal{S}, w[0] \models \phi$;
$\mathcal{S}, w \models \gamma_1 \mathrm{U} \gamma_2$ if there exists j such that $\mathcal{S}, w[j,\infty] \models \gamma_2$, and $\mathcal{S}, w[i,\infty] \models \gamma_1$ for each $0 \leq i < j$.

Finally, we say that $\mathcal{S} \models_x \phi$ if $\mathcal{S}, q_0 \models_x \phi$, where q_0 is the initial state of $\mathcal{S}$.

Example 3. Let us consider the system $\mathcal{S}$ of Example 2 and the **1ATL*** formula $\phi \equiv \langle\langle 1,2\rangle\rangle(GFp^{(1)} \wedge GFp^{(2)})$, where $p^{(i)}(x^{(i)}) = 1$. That is, ϕ says that trains $1,2$ have a strategy so that each visits the tunnel infinitely many times. Consider the joint strategy (σ_1, σ_2) with $\sigma_i(w^{(i)}) = \{(w^{(i)}, i, T^{(i)})\}$, $\sigma_i(T^{(i)}) = \{(T^{(i)}, i, A^{(i)})\}$, and $\sigma_i(A^{(i)}) = \{(A^{(i)}, 3-i, w^{(i)})\}$. All the traces implementing (σ_1, σ_2) alternate the visits of the trains in the tunnel, making the **LTL** formula $GFp^{(1)} \wedge GFp^{(2)}$ satisfied. Thus, $\mathcal{S} \models_x \phi$ for $x \in \{\mathrm{ir}, \mathrm{iR}\}$.

By the same strategy, we get $\mathcal{S} \models_x \langle\langle 1,2\rangle\rangle(GFq^{(1)} \wedge GFq^{(2)})$, where $q^{(i)}(s) = i$.

4 Assumptions and Guarantees

Our assume-guarantee scheme reduces the complexity of model checking by "factorizing" the task into verification of strategies of single agents with respect to abstractions of the rest of the system. In this section, we formalize the notions of *assumption* and *guarantee*, which provide the abstractions in a way that allows for simulating the global behavior of the system.

4.1 Assumptions

Definition 6 (Assumption [32]). *An* assumption *or an* extended module $(M, F) = (X, I, Q, T, \lambda, q_0, F)$ *is a module augmented with a set of accepting states* $F \subseteq Q$.

For assumptions, we use Büchi accepting conditions. More precisely, the infinite word $w = q_0 q_1, \ldots$ is *accepted* by extended module (M, F) with computation $u = \alpha_0\alpha_1 \ldots$ if it is derived by M with a trace $\sigma = q_0\alpha_0 q_1\alpha_1 \ldots$ and $\mathit{inf}(\sigma) \cap F \neq \varnothing$. Thus, the assumptions have the expressive power of ω-regular languages. In practical applications, it might be convenient to formulate actual assumptions in **LTL** (which covers a proper subclass of ω-regular properties).

The definitions of Sections 2 and 3 generalize to assumptions in a straightforward way. In particular, we can compose a module M with an assumption $A' = (M', F')$, and obtain an extended composite module $A = (M|M', F)$, where $F = \{(q, q') \in Q \times Q' \mid q' \in F'\}$. We use the notation $A = M|A'$. Moreover, let $\mathcal{A} = (A, R^{(1)}, \ldots, R^{(m)})$ be a MAS based on the extended module A with repertoires related to all components of M. The semantics of $\mathbf{1ATL}^*_{\mathbf{-X}}$ extends naturally:

$\mathcal{A}, q \models_x \langle\langle C\rangle\rangle\phi$ iff there exists an x-strategy s_C for C such that, for any word $w = w[1]w[2]\ldots$ that implements s_C and is accepted by A, we have $\mathcal{A}, w \models_x \phi$.

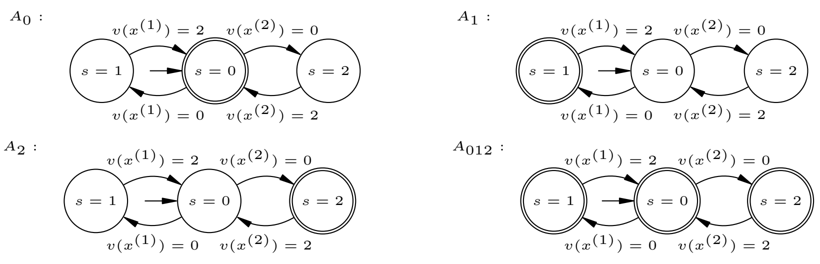


Fig. 3. Assumptions for the railway scenario

Example 4. Recall module $M^{(C(2))} = (X, I, Q, T, \lambda, q_0)$ of the controller for 2 trains, with $Q = \{r, g^{(1)}, g^{(2)}\}$. We define four different assumptions about the behavior of the rest of the system, depicted graphically in Figure 3:

- $A_0 = (X, I, Q, T, \lambda, q_0, \{r\})$
- $A_1 = (X, I, Q, T, \lambda, q_0, \{g^{(1)}\})$
- $A_2 = (X, I, Q, T, \lambda, q_0, \{g^{(2)}\})$
- $A_{012} = (X, I, Q, T, \lambda, q_0, \{r, g^{(1)}, g^{(2)}\})$.

Note that we can identify each valuation with an element of the set $\{0, 1, 2\}$, i.e., the value of the only variable s. This way A_0 as well as A_{012} accept all infinite words of the ω-regular language $L = (0(1|2))^\omega$, while A_1 and A_2 accept only proper subsets of this language, namely $L \setminus (0(1|2))^*(02)^\omega$ and $L \setminus (0(1|2))^*(01)^\omega$.

4.2 Guarantees

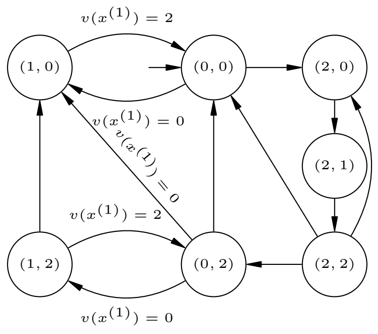
We say that a sequence $v = v_1 v_2 \ldots$ over D^Y is a *curtailment* of a sequence $u = u_1 u_2 \ldots$ over D^X (where $Y \subseteq X$) if there exists an infinite sequence c of indices $c_0 < c_1 < \ldots$ with $c_0 = 0$ such that $\forall_i \forall_{c_i \leq k < c_{i+1}} v_i = u_k|_Y$. We will denote a curtailment of u to D^Y by $u|_Y$ or $u|_Y^c$, and use it to abstract away from irrelevant variables and the stuttering of states.

Definition 7 (Guarantee). *Let $M^{(1)}, \ldots, M^{(k)}$ be pairwise asynchronous modules, and $A = (X^{(A)}, I^{(A)}, Q^{(A)}, T^{(A)}, \lambda^{(A)}, q_0^{(A)}, F^{(A)})$ be an assumption with $X^{(A)} \subseteq X = \bigcup_{i=1}^k X^{(i)}$ and $I^{(A)} \subseteq I = \bigcup_{i=1}^k I^{(i)}$.*

We say that $M = M^{(1)}| \ldots |M^{(k)}$ guarantees *the assumption A (denoted $M \models A$) if, for every infinite trace σ of M with $w \in (D^X)^\omega$ derived by M with σ and $u \in (D^I)^\omega$ admitted by M with σ, there exists a curtailment $w|_{X^{(A)}}^c$ ($c = c_1, c_2, \ldots$) accepted by A with the computation $u_{c_1-1}|_{I^{(A)}}\ u_{c_2-1}|_{I^{(A)}}\ \ldots$.*

That is, every trace of M must agree on the values of $X^{(A)}$ with some trace in A, modulo stuttering.

Example 5. Consider the system $M^{(C(2))}|M^{(1)}|M^{(2)}$ presented in Example 2, its subsystem $M^{(C(2))}|M^{(2)}$ from Figure 4, and the assumption A_{012} of Example 4.



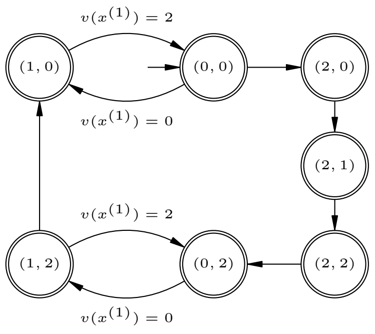


Fig. 4. Module $M^{(C(2))}|M^{(2)}$ (left). The edges are labeled only if the value of $x^{(1)}$ is relevant. Subsystem $M^{(2)}|A_{012}$ implementing strategy σ (right).

If we focus on the changes s, the following words can be derived: $(0(1|2))^\omega$ for the trains taking turns in the tunnel forever, $(0(1|2))^*01^\omega$ for the traces where the semaphore is stuck in state $(1, 0)$ because it never receives that $v(x^{(1)}) = 2$, and $(0(1|2))^*02^\omega$ for ones that cycle forever in the right-hand part of $M^{(C(2))}|M^{(2)}$. In consequence, we have that $M^{(C(2))}|M^{(2)} \models A_{012}$, but not $M^{(C(2))}|M^{(2)} \models A_1$.

It is possible to relate the traces of a subsystem with the traces of the entire system in such a way that it is possible to verify locally defined formulas.

5 Assume-Guarantee Reasoning for 1ATL*

Now we propose our assume-guarantee schemes that decompose abilities of coalition C into abilities of its subcoalitions, verified in suitable abstractions of their neighbor modules.

5.1 Assume-Guarantee Rule for Strategies

Let $\mathcal{S}$ be a system composed of asynchronous agents $(M^{(1)}, R^{(1)}), \ldots, (M^{(n)}, R^{(n)})$. By $N_1^{(i)}$, we denote the direct "neighborhood" of agent i, i.e., the set of agent indices j such that $I_{M^{(j)}} \cap X_{M^{(i)}} \neq \varnothing$ or $I_{M^{(i)}} \cap X_{M^{(j)}} \neq \varnothing$. By $N_k^{(i)}$, we denote the agents connected to i in at most k steps, i.e., $(N_{k-1}^{(i)} \cup \bigcup_{j \in N_{k-1}^{(i)}} N_1^{(j)}) \setminus \{i\}$. Finally $Comp_k^{(i)}$ denotes the composition of all modules of $N_k^{(i)}$. That is, if $N_k^{(i)} = \{a_1, ..., a_m\}$ then $Comp_k^{(i)} = M^{(a_1)}|...|M^{(a_m)}$.

Let ψ_i be an **LTL** formula (without "next"), where atomic propositions are local valuations of variables in $M^{(i)}$. Also, let $x \in \{\mathrm{ir}, \mathrm{iR}\}$. The scheme is formalized through a sequence of rules $\mathbf{R_k}$ which rely on the behaviour of the neighbourhoods of coalition C, limited by "distance" k:

$$\mathbf{R_k} \quad \frac{\begin{array}{c}\forall_{i \in C}\ (M^{(i)}|A_i, R^{(i)}) \models_x \langle\langle i \rangle\rangle \psi_i \\ \forall_{i \in C}\ Comp_k^{(i)} \models A_i\end{array}}{(M^{(1)}|...|M^{(n)}, R^{(1)}, \ldots, R^{(n)}) \models_x \langle\langle C \rangle\rangle \bigwedge_{i \in C} \psi_i}$$

The main challenge in applying the scheme is to define the right assumptions and to decompose the verified formula.

Example 6. Recall the multi-agent system $\mathcal{S}$ presented in Example 2, based on module $M^{(C(2))}|M^{(1)}|M^{(2)}$. We already argued that it satisfies $\phi \equiv \langle\langle 1,2\rangle\rangle(GFp^{(1)}) \wedge (GFp^{(2)})$ as well as $\phi' \equiv \langle\langle 1,2\rangle\rangle(GFq^{(1)}) \wedge (GFq^{(2)})$, for $p^{(i)}(x^{(i)}) = 1$ and $q^{(i)}(s) = i$, cf. Example 3. We will now see if the verification of the formulas can be decomposed using $\mathbf{R_k}$.

By Example 5 we know that $M^{(C(2))}|M^{(i)} \models A_{012}$, where A_{012} was an assumption defined in Example 4. It is easy to see that $M^{(C(2))} \models A_{012}$.

Consider the extended module $M^{(2)}|A_{012}$, which is nothing but $M^{(2)}|M^{(C(2))}$ with all the states marked as accepting. Assume further that agent 2 executes strategy σ_2 of Example 3. The resulting subsystem is presented in Figure 4. Note that, if we focus on the values of variable s, the ω-regular language accepted by this automaton is $((01)|(0222(01)^*011))^\omega$, hence it periodically satisfies $p(\{s\}) = 1$. In consequence, σ_2 can be used to demonstrate that $(M^{(2)}|A_{012}, R^{(2)}) \models_{\mathrm{ir}} \langle\langle 2\rangle\rangle GFq^{(1)}$, where $q^{(1)}(s) = 1$. Similarly, $(M^{(1)}|A_{012}, R^{(1)}) \models_{\mathrm{ir}} \langle\langle 1\rangle\rangle GFq^{(2)}$, where $q^{(2)}(s) = 2$.

As a result, we have decomposed formula ϕ' and constructed independent strategies for agents 1 and 2. By the use of rule $\mathbf{R_1}$, we conclude that

$$(M, R^{(C(2))}, R^{(1)}, R^{(2)}) \models_{\mathrm{ir}} \langle\langle 1,2\rangle\rangle(GFq^{(1)}) \wedge (GFq^{(2)}).$$

The situation for $\phi \equiv \langle\langle 1,2\rangle\rangle(GFp^{(1)}) \wedge (GFp^{(2)})$ is drastically different. We cannot use the analogous reasoning, because $\langle\langle i\rangle\rangle GFp^{(3-i)}$ is not a local constraint for $M^{(i)}$. There is a unique decomposition of ϕ into local constraints, but proving that $(M^{(1)}|A_{012}, R^{(1)}) \models_{\mathrm{ir}} \langle\langle 1\rangle\rangle GFp^{(1)}$ fails, as the system can get stuck in the state where s equals 2 or infinitely loop between the states where $s = 2$ and $s = 0$. Changing the assumption would not help, since we cannot avoid the infinite exclusion of the considered train. Thus, while the scheme can be used to derive that $\mathcal{S} \models_{\mathrm{ir}} \phi'$, it cannot produce the (equally true) statement $\mathcal{S} \models_{\mathrm{ir}} \phi$.

5.2 Soundness and Incompleteness

The following theorem says that, if each coalition member together with its assumption satisfies the decomposition of the formula, and its neighborhood satisfies the assumption, then the original verification task must return “true.”

Theorem 1. *The rule* $\mathbf{R_k}$ *is sound.*

Proof. Let $\forall_{i\in C}\ (M^{(i)}|A_i, R^{(i)}) \models_x \langle\langle i\rangle\rangle\psi_i$ with (memoryless or perfect recall) imperfect information strategy σ_i and $\forall_{i\in C}\ Comp_k^{(i)} \models A_i$. Here and in the rest of the proof, $x \in \{ir, iR\}$.

Let us consider $M = M^{(1)}|...|M^{(n)}$ such that $(M, R^{(1)}, \ldots, R^{(n)}) \models_x \langle\langle C\rangle\rangle\psi_i$ and fix its joint strategy σ for coalition C, where $\sigma(i) = \sigma_i$ for every $i \in C$.

We will prove the soundness by contradiction. Suppose that for every (memoryless or perfect recall) imperfect information joint strategy there exists an infinite word which implements this joint strategy, but do not satisfy $\bigwedge_{i\in C}\psi_i$, i.e. there exists $j \in C$ such that w does not satisfy ψ_j. Let $w = q_0 q_1 \ldots$ be such a word for the strategy σ and fix j.

Let us consider $M^{(j)}|A_j$, where $X_{M^{(j)}}$ and X_{A_j} are internal variables of $M^{(j)}$ and A_j, appropriately. By the construction and the presumption that $(M^{(j)}|A_j, \mathrm{R}^{(j)}) \models_x \langle\langle j\rangle\rangle\psi_j$ we get that every infinite word over $X_{M^{(j)}} \cup X_{A_j}$ which implement (memoryless or perfect recall) imperfect information strategy σ_j satisfy ψ_j.

However, the assumption A_j is guaranteed by $Comp_k^{(j)}$, hence for a word derived by $Comp_k^{(j)}$ we have its curtailment accepted by A_j. Moreover, every word accepted by $M^{(j)}|A_j$ is a curtailment of a word derived by M, and, in particular, w is such a word. However, there exists a curtailment $w|_{X_{M^{(j)}}\cup X_{A_j}}$ which satisfy strategy σ_j but is not accepted by $M^{(j)}|A_j$, which gives an obvious contradiction with $(M^{(j)}|A_j, R^{(j)}) \models_x \langle\langle i\rangle\rangle\psi_j$.

The obtained contradiction shows that there exists a joint strategy σ for the entire model and $(M^{(1)}|...|M^{(n)}, R^{(1)}, \ldots, R^{(n)}) \models_x \langle\langle C\rangle\rangle \bigwedge_{i\in C}\psi_i$, which concludes the proof.

Unfortunately, there does not always exist $k < n$ for which the rule $\mathbf{R_k}$ is complete, even in a very weak sense, where we only postulate the *existence* of appropriate assumptions.

Theorem 2. *The scheme consisting of rules* $\{\mathbf{R_k} \mid \mathbf{k} \in \mathbb{N}\}$ *is in general not complete.*

Proof. Follows directly from Example 6.

5.3 Coalitional Assume-Guarantee Verification

In Section 5.2, we showed that achievable coalitional goals may not decompose into achievable individual subgoals. As a result, the scheme proposed in Section 5.1 is incomplete. A possible way out is to allow for assume-guarantee reasoning about joint strategies of subcoalitions of C. We implement the idea by partitioning the system into smaller subsystems and allowing to explicitly consider the cooperation between coalition members.

Again, let $\mathcal{S} = (M^{(1)}, R^{(1)}),\ \ldots,\ (M^{(n)}, R^{(n)})$ be a system composed of asynchronous agents . Moreover, let $\{P_1, \ldots, P_k : P_i \subseteq \{1, 2, \ldots, n\}\}$, be a partitioning of coalition C, and let $\overline{C} = \{i : i \notin C\} = Ag \setminus C$ be the set of opponents of C. By $\mathcal{S}^{(P_i)}$ we denote the system composed of all the agents in $P_i = \{i_1, \ldots, i_s\}$, i.e., $(M^{(P_i)} = M^{(i_1)}|\ldots|M^{(i_s)}, R^{(i_1)}, \ldots, R^{(i_s)})$. $\mathcal{S}^{(\overline{C})}$ is defined analogously.

We extend the notion of neighbourhood to sets of agents as follows:

- $N_1^{P_i} = (\bigcup_{i\in P_i} N_1^{(i)}) \setminus P_i$, $N_k^{P_i} = (N_{k-1}^{P_i} \cup \bigcup_{j\in N_{k-1}^{P_i}} N_1^{(j)}) \setminus P_i$ for $k > 1$,
- $Comp_k^{P_i} = M^{(x_1)}|...|M^{(x_s)}$ for $N_k^{P_i} = \{x_1, ..., x_s\}$.

Let $x \in \{\mathrm{ir}, \mathrm{iR}\}$. The generalized assume-guarantee rule is defined below:

$$\mathbf{Part_k^P}\quad \frac{\begin{array}{c}\forall_{P_i\in P}\ (M^{(P_i)}|A_i, R^{(i_1)}, \ldots, R^{(i_s)}) \models_x \langle\langle P_i\rangle\rangle \bigwedge_{j\in P_i}\psi_j \\ \forall_{P_i\in P}\ Comp_k^{P_i} \models A_i\end{array}}{(M^{(1)}|...|M^{(n)}, R^{(1)}, \ldots, R^{(n)}) \models_x \langle\langle C\rangle\rangle \bigwedge_{i\in C}\psi_i}$$

As it turns out, the new scheme is sound, conservative with respect to enlarging the neighborhood, and complete.

Theorem 3. *The rule* $\mathbf{Part^P_k}$ *is sound.*

Proof. Intuitively, we can proceed similarly to the proof of Theorem 1. Note that each component P_i can be seen as single composed module with a imperfect information strategy (memoryless or with perfect recall) being a joint strategy for the subset of coalition C which is the component P_i. Moreover, we can take instead of $Comp^{P_i}_k$ the union U of all the components P_j (and possibly $\overline{C}$) which intersection with $Comp^{P_i}_k$ is non-empty. It is easy to see, that if $Comp^{P_i}_k \models A_i$ then also $U \models A_i$.

This way we could fix the strategy for coalition C, deduce that every infinite word as a composition of strategies σ_{P_i} for its parts $(C \cap P_i)_{P_i \in P}$, and deduce that for every word w which do not satisfy $\bigwedge_{i \in C} \psi_i$ there exists a single component P_j containing M_i such that ψ_j would not be satisfied for any curtailment of w, while one of them implements strategy σ_{P_i} being at the same time accepted by $M^{(P_i)}|A_i$.

Proposition 1. *If* $Comp^{P_i}_k \models A_i$ *then* $Comp^{P_i}_{k+1} \models A_i$*.*

Proof. Let $N^{P_i}_{k+1} = \{i_1, \ldots, i_t\}$, $M = M^{(i_1)}|\ldots M^{(i_t)}$, $N^{P_i}_k = \{j_1, \ldots, j_{t'}\}$ and $M' = M^{(j_1)}|\ldots M^{(j_{t'})}$. Let us consider an infinite trace σ of M, with $w \in (D^X)^\omega$ and $u \in (D^I)^\omega$ and $\sigma' = w_1|_{X'}\,(u_1|_{I' \cap I} \cup w_1|_{I' \cap X})\,w_2|_{X'}\ldots$. Note that one of the curtailments of a word $w_1|_{X'} w_2|_{X'} \ldots$ is derived by M', and thus its curtailment is accepted by A_i.

Theorem 4. *There exist a partition set* P *and* $k \leq n$ *such that the rule* $\mathbf{Part^P_k}$ *is complete.*

Proof. Straightforward, as we can take $k = n$ and singleton partition $P = \{P_1\}$, where A_1 is an automaton constructed on the base of the system $M^{(\overline{C})}$, where all the states are accepting ones (hence $Comp^k_{P_1} \models A_1$ as every word accepted by A_1 is derived with a trace of $M_{\overline{C}}$).

Hence $(M^{(P_1)}|A_1, R^{(i_1)}, \ldots, \mathrm{R}^{(i_s)}) \models_x \langle\langle P_1 \rangle\rangle \bigwedge_{j \in P_1} \psi_j$ is just an equivalent formulation of $(M^{(1)}|...|M^{(n)}, R^{(1)}, \ldots, R^{(n)}) \models_x \langle\langle C \rangle\rangle \bigwedge_{i \in C} \psi_i$, for $x \in \{ir, iR\}$.

Remark 1 (Complexity). The assume-guarantee schemes provide (one-to-many) reductions of the model checking problem. The resulting verification algorithm for $\mathbf{ATL}^*_{\mathrm{ir}}$ is $\mathbf{PSPACE}$-complete with respect to the size of the coalition modules, the size of the assumptions, and the length of the formula. In the very worst case (i.e., as the assumptions grow), this becomes $\mathbf{PSPACE}$-complete w.r.t. the size of the global model, i.e., no better than ordinary model checking for $\mathbf{ATL}^*$ with memoryless strategies. On the other hand, our method often allows to decompose the verification of the huge global model of the system to several smaller cases. For many systems one can propose assumptions that are exponentially smaller than the size of the full model, thus providing an exponential gain in complexity.

Note also that the first scheme provides a model checking algorithm for $\mathbf{ATL}^*_{\mathrm{iR}}$ that is $\mathbf{EXPTIME}$-complete with respect to the size of the coalition modules, the size of the assumptions, and the length of the formula, i.e., an incomplete but decidable algorithm for the generally undecidable problem.

5.4 Verification of Nested Strategic Operators

So far, we have concentrated on assume-guarantee specification of formulas without nested strategic modalities. Here, we briefly point out that the schemes $\mathbf{R_k}$ and $\mathbf{Part_k^P}$ extend to the whole language of $\mathbf{ATL}^*_{-\mathbf{X}}$ through the standard recursive model checking algorithm that verifies subformulas bottom-up. Such recursive application of the method to the verification of $\mathcal{S} \models \phi$ proceeds as follows:

- For each strategic subformula ϕ_j of ϕ, do assume-guarantee verification of ϕ_j in $\mathcal{S}$, and label the states where ϕ_j holds by a fresh atomic proposition $\mathsf{p_j}$;
- Replace all occurrences of ϕ_j in ϕ by $\mathsf{p_j}$, and do assume-guarantee verification of the resulting formula in $\mathcal{S}$.

The resulting algorithm is sound, though there is the usual price to pay in terms of computational complexity. The main challenge lies in providing decompositions of $\mathbf{LTL}$ objectives for multiple strategic formulas, as well as multiple Büchi assumptions (one for each subformula). A refinement of the schemes for nested strategic abilities is planned for future work.

6 Case Study and Experiments

In this section, we present an experimental evaluation of the assume-guarantee verification schemes of Section 5. As the benchmark, we use a variant of the factory scenario from [26], where a coalition of logistic robots cooperate to deliver packages from the production line to the storage area.

6.1 Experiments: Monolithic vs. Assume-Guarantee Verification

Decomposition to Individual Strategies. In the first set of experiments, we verified the formula

$$\psi \equiv \langle\langle R\rangle\rangle(\bigwedge_{r\in R} \mathsf{energy_r} > 0)\,\mathrm{U}\,\mathsf{delivered}$$

expressing that the coalition of robots R can maintain their energy level above zero until at least one package is delivered to the storage area. Guessing that the first robot has enough energy to deliver a package on his own, we can decompose the formula as the conjunction of the following components:

$$\psi_d \equiv \langle\langle r_1\rangle\rangle \mathrm{F}\,\mathsf{delivered}, \qquad \psi_e^{(i)} \equiv \langle\langle r_i\rangle\rangle \mathrm{G}\,\mathsf{energy_{r_i}} > 0, \quad i \in R.$$

Note that, if $\psi_d \wedge \bigwedge_{i>1} \psi_e^{(i)}$ is true, then ψ must be true, too.

The experiments used the first (incomplete) scheme of assume-guarantee verification. The results are presented in Table 1. The first column describes the configuration of the model, i.e., the number of robots, locations in the factory, and the initial energy level. Then, we report the performance of model checking algorithms that operate on the explicit model of the whole system. The running times are given in seconds. *DFS* is a straightforward implementation of depth-first strategy synthesis. *Apprx* refers to the (sound but incomplete) method of fixpoint-approximation in [22]; besides the time, we also report if the approximation was conclusive.

Conf	Monolithic verif.			Ass.-guar. verif.		
	#st	DFS	Apprx	#st	DFS	Apprx
2,2,2	8170	<0.01	0.6/No	1356	<0.01	<0.01/Yes
2,3,3	1.1e5	0.02	13/No	9116	<0.01	0.5/Yes
3,2,2	5.5e5	timeout		2.7e4	<0.01	3/Yes
3,3,3	memout			4.4e5	<0.01	58/Yes
4,2,2	memout			5.2e5	timeout	

Conf	Monolithic verif.			Ass.-guar. verif.		
	#st	DFS	Apprx	#st	DFS	Apprx
2,3,1	522	<0.01	<0.01/No	522	<0.01	<0.01/No
2,4,2	3409	<0.01	<0.01/No	3409	<0.01	<0.01/No
4,3,1	memout			4.8e4	<0.01	4/No
6,3,1	memout			5.8e5	0.36	42/No
8,3,1	memout			timeout		

Table 1. Results of assume-guarantee verification, scheme $\mathbf{R_k}$ (left), scheme $\mathbf{Part_k^P}$ (right).

Coalitional Assume-Guarantee Verification. For the second set of experiments, the robots were divided in two halves, initially located in different parts of the factory. We verified the following formula:

$$\psi \equiv \langle\!\langle R \rangle\!\rangle \mathrm{F}\,\mathrm{G}\,(\textstyle\bigwedge_{\mathsf{i}\in\{1,2,\ldots,\mathsf{n}/2\}}(\mathsf{delivered_i} \vee \mathsf{delivered_{i+n/2}})),$$

expressing that the coalition of robots can delivered at least one package per pair to the storage area. Depending on the initial energy level of robots, the storage may not be reachable from the production line. That means that the robots must work in pairs to deliver the packages. We use this insight to decompose the verification into the following formulas:

$$\psi^{(i)} \equiv \langle\!\langle r_i, r_{i+n/2} \rangle\!\rangle \mathrm{F}\,\mathrm{G}\,(\mathsf{delivered_i} \vee \mathsf{delivered_{i+n/2}}).$$

The results are presented in Table 1.

Discussion of Results. The experimental results show that assume-guarantee schemes presented here enables to verify systems of distinctly higher complexity than model checking of the full model. We have also conducted analogous experiments on the Simple Voting scenario of [22], with very similar results; we do not report them here due to lack of space.

Interestingly, Table 1 shows that the application of incomplete assume-guarantee scheme to fixpoint approximation (in itself an incomplete method of model checking) often turns inconclusive verification into conclusive one. This is because fixpoint approximation works rather well for individual abilities, but poorly for proper coalitions [22]. Rule $\mathbf{R_k}$ decomposes verification of coalitional abilities (very likely to resist successful approximation) to model checking individual abilities (likely to submit to approximation). It is not true in the case of the second experiment, as this time we did not reduce the tested coalitions to singleton ones.

7 Conclusions

In this paper we propose two schemes for assume-guarantee verification of strategic abilities. Importantly, they are both sound for the memoryless as well as perfect recall semantics of abilities under imperfect information. Moreover, the second scheme is complete (albeit in a rather weak sense). The experiments show that both schemes can provide noticeable improvement in verification of large systems consisting of asynchronous agents with independent goals. Note also that the scheme $\mathbf{R_k}$ provides an (incomplete) reduction of the undecidable model checking problem for coalitions with perfect recall to decidable verification of individual abilities.

Clearly, the main challenge is to formulate the right assumptions and to decompose the verified formula. In the future, we would like to work on the automated generation of assumptions. The first idea is to obtain a larger granularity of the global model by decomposing agents into even smaller subsystems (and recomposing some of them as assumptions). This can be combined with abstraction refinement of the assumptions in case they are still too complex. We also plan to extend the notion of assumptions to capture the agents' knowledge about the strategic abilities of their coalition partners. Positive results in that direction would significantly increase the applicability of assume-guarantee schemes for model checking of asynchronous MAS.

Acknowledgement The work was supported by NCBR Poland and FNR Luxembourg under the PolLux/FNR-CORE project STV (POLLUX-VII/1/2019), and the CHIST-ERA grant CHIST-ERA-19-XAI-010 by NCN Poland (2020/02/Y/ST6/00064). The work of Damian Kurpiewski was also supported by the CNRS IEA project MoSART.

References

1. Alur, R., Henzinger, T.: Reactive modules. Form. Meth. Syst. Design **15**(1), 7–48 (1999)
2. Alur, R., Henzinger, T., Kupferman, O.: Alternating-time Temporal Logic. J. of the ACM **49**, 672–713 (2002)
3. Alur, R., Henzinger, T., Mang, F., Qadeer, S., Rajamani, S., Tasiran, S.: MOCHA: Modularity in model checking. In: Proc. of CAV. LNCS, vol. 1427, pp. 521–525. Springer (1998)
4. Alur, R., Henzinger, T., Vardi, M.: Parametric real-time reasoning. In: Proc. of STOC. pp. 592–601. ACM (1993)
5. Bulling, N., Dix, J., Jamroga, W.: Model checking logics of strategic ability: Complexity. In: Specification and Verification of Multi-Agent Systems, pp. 125–159. Springer (2010)
6. Busard, S., Pecheur, C., Qu, H., Raimondi, F.: Reasoning about memoryless strategies under partial observability and unconditional fairness constraints. Inf. and Comp. **242**, 128–156 (2015)
7. Cermák, P., Lomuscio, A., Murano, A.: Verifying and synthesising multi-agent systems against one-goal strategy logic specifications. In: Proc. of AAAI. pp. 2038–2044 (2015)
8. Chatterjee, K., Henzinger, T.: Assume-guarantee synthesis. In: International Conference on Tools and Algorithms for the Construction and Analysis of Systems. pp. 261–275. Springer (2007)
9. Chen, Y., Clarke, E., Farzan, A., He, F., Tsai, M., Tsay, Y., Wang, B., Zhu, L.: Comparing learning algorithms in automated assume-guarantee reasoning. In: Proc. of ISoLA'10, Part I. LNCS, vol. 6415, pp. 643–657. Springer (2010)
10. Clarke, E., Long, D., McMillan, K.: Compositional model checking. In: Proc. of LICS. pp. 353–362. IEEE Comput. Soc. Press (1989)
11. Devereux, B.: Compositional reasoning about aspects using alternating-time logic. In: Proc. of FOAL. pp. 45–50 (2003)
12. Dima, C., Tiplea, F.: Model-checking ATL under imperfect information and perfect recall semantics is undecidable. CoRR **abs/1102.4225** (2011)
13. Fijalkow, N., Maubert, B., Murano, A., Vardi, M.: Assume-guarantee synthesis for prompt linear temporal logic. In: Proc. of IJCAI. pp. 117–123. ijcai.org (2020)
14. Finkbeiner, B., Passing, N.: Compositional synthesis of modular systems. Innovations in Systems and Software Engineering pp. 1–15 (2022)

15. Giannakopoulou, D., Pasareanu, C., Barringer, H.: Component verification with automatically generated assumptions. Autom. Softw. Eng. **12**(3), 297–320 (2005)
16. Guelev, D., Dima, C., Enea, C.: An alternating-time temporal logic with knowledge, perfect recall and past: axiomatisation and model-checking. J. Appl. Non-Classical Log. **21**(1), 93–131 (2011)
17. He, F., Mao, S., Wang, B.: Learning-based assume-guarantee regression verification. In: Proc. of CAV'16, Part I. LNCS, vol. 9779, pp. 310–328. Springer (2016)
18. Henzinger, T., Qadeer, S., Rajamani, S.: You assume, we guarantee: Methodology and case studies. In: Proc. of CAV. LNCS, vol. 1427, pp. 440–451. Springer (1998)
19. Hoare, C.: An axiomatic basis for computer programming. Commun. ACM **12**(10), 576–580 (1969)
20. Huang, X., van der Meyden, R.: Symbolic model checking epistemic strategy logic. In: Proc. of AAAI. pp. 1426–1432 (2014)
21. Jamroga, W., Kim, Y., Kurpiewski, D., Ryan, P.: Towards model checking of voting protocols in uppaal. In: Proc. of E-Vote-ID. LNCS, vol. 12455, pp. 129–146. Springer (2020)
22. Jamroga, W., Knapik, M., Kurpiewski, D., Mikulski, Ł.: Approximate verification of strategic abilities under imperfect information. Artif. Int. **277** (2019)
23. Jamroga, W., Penczek, W., Sidoruk, T.: Strategic abilities of asynchronous agents: Semantic side effects and how to tame them. In: Proc. of KR. pp. 368–378 (2021)
24. Jones, C.: Specification and design of (parallel) programs. In: Proc. of IFIP. pp. 321–332. North-Holland/IFIP (1983)
25. Kong, S., Jung, Y., David, C., Wang, B., Yi, K.: Automatically inferring quantified loop invariants by algorithmic learning from simple templates. In: Proc. of APLAS. LNCS, vol. 6461, pp. 328–343. Springer (2010)
26. Kurpiewski, D., Marmsoler, D.: Strategic logics for collaborative embedded systems. SICS Soft.-Inten. Cyber-Phys. Syst. **34**(4), 201–212 (2019)
27. Kurpiewski, D., Pazderski, W., Jamroga, W., Kim, Y.: STV+Reductions: Towards practical verification of strategic ability using model reductions. In: Proc. of AAMAS. pp. 1770–1772. ACM (2021)
28. Kurpiewski, D., Mikulski, Ł., Jamroga, W.: STV+AGR: Towards verification of strategic ability using assume-guarantee reasoning. In: Proceedings of PRIMA (2022), to appear
29. Kwiatkowska, M., Norman, G., Parker, D., Qu, H.: Assume-guarantee verification for probabilistic systems. In: Proc. of TACAS. LNCS, vol. 6015, pp. 23–37. Springer (2010)
30. Lomuscio, A., Qu, H., Raimondi, F.: MCMAS: An open-source model checker for the verification of multi-agent systems. Int. J. Soft. Tools Tech. Trans. **19**(1), 9–30 (2017)
31. Lomuscio, A., Strulo, B., Walker, N., Wu, P.: Assume-guarantee reasoning with local specifications. In: Proc. of ICFEM. LNCS, vol. 6447, pp. 204–219. Springer (2010)
32. Lomuscio, A., Strulo, B., Walker, N., Wu, P.: Assume-guarantee reasoning with local specifications. Int. J. Found. Comput. Sci. **24**(4), 419–444 (2013)
33. Mikulski, Ł., Jamroga, W., Kurpiewski, D.: Towards assume-guarantee verification of strategic ability. In: Proc. of AAMAS'22. pp. 1702–1704. IFAAMAS (2022)
34. Mogavero, F., Murano, A., Perelli, G., Vardi, M.: Reasoning about strategies: On the model-checking problem. ACM Trans. Comp. Log. **15**(4), 1–42 (2014)
35. Owicki, S., Gries, D.: Verifying properties of parallel programs: An axiomatic approach. Commun. ACM **19**(5), 279–285 (1976)
36. Pnueli, A.: In transition from global to modular temporal reasoning about programs. In: Logics and Models of Concurrent Systems. NATO/ASIs, vol. 13, pp. 123–144. Springer (1984)
37. Schobbens, P.: Alternating-time logic with imperfect recall. Electr. Not. Theor. Comput. Sci. **85**(2), 82–93 (2004)