Piotr Dembiński, Wojciech Jamroga
Antoni Mazurkiewicz, Wojciech Penczek

Towards Partial Order Reductions for Fragments of Alternating-Time Temporal Logic

Nr 1036

Warsaw, December 2016

Abstract

A general semantics of strategic abilities of agents in asynchronous systems with and without perfect information is proposed, and some general complexity results for verification of strategic abilities in asynchronous systems are presented. A methodology for *partial order reduction (POR)* in verification of agents with imperfect information is developed, based on the notion of *traces* introduced by Mazurkiewicz. Two semantics of $\mathbf{ATL}^*_{-\mathbf{X}}$ are considered and it is shown that for memoryless imperfect information ($\models_{ir}$) contrary to memoryless perfect information ($\models_{Ir}$), one can apply techniques known for $\mathbf{LTL}_{-\mathbf{X}}$.

Keywords: Alternating-Time Temporal Logic, asynchronous systems, partial order reduction, traces

Streszczenie

O redukcjach częścio-porządkowych dla fragmentów logiki temporalnej czasu alternującego

Raport definiuje ogólną semantykę dla strategicznych umiejętności agentów w systemach asynchronicznych z pełną i niepełną informacją, oraz prezentuje ogólne wyniki dotyczące złożoności weryfikacji strategicznych umiejętności w systemach asynchronicznych. Metoda redukcji częścio-porządkowych, wykorzystująca ślady Mazurkiewicza, została zastosowana do weryfikacji agentów z niepełną informacją. Dla rozważanych semantyk logiki $\mathbf{ATL}^*_{-\mathbf{X}}$ zostało pokazane, że dla bezpamięciowej niepełnej informacji ($\models_{ir}$) w przeciwieństwie do bezpamięciowej pełnej informacj ($\models_{Ir}$), można zastosować metody znane dla $\mathbf{LTL}_{-\mathbf{X}}$.

Słowa kluczowe: Logika temporalna czasu alternującego, systemy asynchroniczne, redukcje częściowo-porządkowe, ślady

Contents

1 Introduction

Multi-agent systems describe interactions of multiple entities called *agents*, often assumed to be intelligent and autonomous. *Alternating-time temporal logic* $\mathbf{ATL}^*$ and its fragment $\mathbf{ATL}$ [3, 6] are logics which allow for reasoning about strategic interactions in such systems. The main idea is to extend the framework of temporal logic with the game-theoretic notion of *strategic ability*. Hence, $\mathbf{ATL}^*$ enables to express statements about what agents (or groups of agents) can achieve. For example, $\langle\!\langle a \rangle\!\rangle \mathrm{F}\mathsf{win_a}$ says that agent a has the ability to eventually win no matter what the other agents do, while $\langle\!\langle a, b \rangle\!\rangle \mathrm{G}\mathsf{safe}$ expresses that agents a and b together can force the system to always remain in a safe state. Such properties can be useful for specification, verification, and reasoning about interaction in agent systems. They have become especially relevant due to active development of algorithms and tools for verification where the "correctness" property is given in terms of strategic ability [4, 5, 36, 45, 14, 28, 11, 59, 50]. However, there are two caveats.

First, most of the tools and algorithmic solutions focus on agents with perfect information, i.e., agents who at any point of the game know exactly the global state of the game. This assumption is clearly unrealistic in all but the simplest multi-agent scenarios. Still, the tendency is somewhat easy to understand, since model checking of $\mathbf{ATL}$ variants with imperfect information has been proved $\mathbf{\Delta_2^P}$- to $\mathbf{PSPACE}$-complete for agents playing memoryless (a.k.a. positional) strategies [62, 33, 10] and $\mathbf{EXPTIME}$-complete to undecidable for agents with perfect recall of the past [18, 26]. Moreover, the imperfect information semantics of $\mathbf{ATL}$ does not admit alternation-free fixpoint characterizations [8, 19, 20], which makes incremental synthesis of strategies impossible, or at least difficult to achieve. Some early attempts at verification of imperfect information strategies made their way into the MCMAS model-checker [46, 60, 47, 50], but the issue has never been at the heart of the tool. More dedicated attempts have begun to emerge only recently [59, 11, 28, 12, 35]. Up until now, experimental results confirm that the initial intuition was right: model checking strategic modalities for imperfect information is hard, and dealing with it requires innovative algorithms and verification techniques.

Secondly, the semantics of strategic logics are almost exclusively based on synchronous concurrent game models. That is, one implicitly assumes the existence of a global clock that triggers subsequent global events in the system. At each tick of the clock, all the agents choose their actions, and the system proceeds accordingly with the corresponding global transition. However, many real-life systems are inherently asynchronous, and do not operate on a global clock that perfectly synchronizes the atomic steps of all the components. As an example, consider robots interacting in an environment with faulty communication and/or non-negligible delays in execution of actions. No less importantly, many systems whose implementation may be synchronous at the implementation level (say, the level of the virtual machine) can be conveniently modeled as asynchronous on a more abstract level, because when we abstract away the implementation details it is not clear anymore how transitions initiated by different agents are exactly arranged in a particular temporal order. For instance, the actual implementation of a soccer match in the simulated RoboCup competition can be executed on a single computer with a global clock ticking every 0.3 ns, but the corresponding synchronous model would be huge and in consequence useless for any kind of analysis. Instead, one can remove a lot of unnecessary details by assuming that the players execute their actions asynchronously – without clear temporal relationship between their execution times – and synchronize only when a particular event has to be executed *jointly*. In many scenarios, both aspects combine. For example, when modeling a national election, one must take into account both the truly asynchronous nature of events happening at different polling stations, and the best level of granularity for modeling the events happening within a single polling station.

In this paper, we make the first step towards strategic analysis of such systems. Our contribution is threefold:

1.

2. We propose a general semantics of strategic abilities of agents in asynchronous systems, with and without perfect information.

3. We present some general complexity results for verification of strategic abilities in asynchronous systems.

4. We develop a methodology for *partial order reduction (POR)* in ver-

ification of agents with imperfect information, based on the notion of *traces* introduced by Mazurkiewicz in [53]. We also observe that, interestingly, the scheme does *not* work for verification of agents with perfect information.

Partial order reduction is one of the most widely known techniques in verification of reactive systems. Our main goal is to obtain a variant of POR that can be used for verification of strategic properties in multi-agent systems.

Related work. Related relevant work is relatively scarce. Alur, Henzinger and Kupferman mentioned asynchronous systems in their seminal paper on **ATL** [6], but they modeled them as a special case of synchronous systems. Reactive modules [2], the class of representations behind the Mocha model checker [4, 5], feature several modes of asynchronous execution, but – to the best of our knowledge – this aspect has never been given a more systematic analysis.

Asynchronous semantics and partial order reduction were extensively studied in [54, 55, 56, 38, 24, 23, 57, 22, 40, 58]. The most recent approaches include dynamic POR [21, 1, 13] and combine POR with symbolic methods [37, 39]. Still, the only efficient approach to partial order reduction in a MAS context [48, 49] presents results for standard epistemic-temporal logics (LTLK_{-X}, $\mathrm{CTL^*K}_{-X}$) interpreted over interleaved interpreted systems. It is by no means immediately clear how those approaches extend to modeling and verification of strategic play from autonomous, rational, and purposeful players.

The work that comes closest to our new proposal is [17] where a variant of **ATL** was proposed for the special case of agent-oriented agent programs written in 2APL with asynchronous execution semantics. A very crude and rather impractical notion of stuttering equivalence was also proposed there, as the first step towards a partial order reduction scheme.

It should also be mentioned that our complexity results largely coincide with the pattern already known for model checking of synchronous systems, cf. e.g. [10].

2 Preliminaries

We begin with defining asynchronous multi-agent systems and their semantics in terms of interleaved interpreted systems.

2.1 Asynchronous Multi-Agent Systems

We will model multi-agent systems as networks of automata that execute asynchronously by interleaving of local transitions, and synchronize their moves whenever a shared action is executed. Formally, consider a MAS composed of n agents $\mathcal{A} = \{1, \ldots, n\}$.[1] Each agent $i \in \mathcal{A}$ is associated with a set of *possible local states* $L_i = \{l_i^1, l_i^2, \ldots, l_i^{nl_i}\}$ and a set of *actions* $Act_i = \{\epsilon, a_i^1, a_i^2, \ldots, a_i^{na_i}\}$. The special action ϵ, called the "null" action of agent i, does not change the local state of i, for each $i \in \mathcal{A}$. Notice that the sets of actions of the agents do not need to be disjoint also for actions different than ϵ. $Act = \bigcup_{i \in \mathcal{A}} Act_i$ is the union of all the sets Act_i, whereas $Loc = \bigcup_{i \in \mathcal{A}} L_i$ is the union of all the sets L_i. For each action $a \in Act$ the set $Agent(a) \subseteq \mathcal{A}$ contains these agents i for which $a \in Act_i$, i.e., these which can potentially execute a.

We define the independency I on Act as follows: $I = \{(a, b) \in Act \times Act \mid Agent(a) \cap Agent(b) = \emptyset\}$. Notice that ϵ is dependent on all the other actions of Act.

Following the interpreted system model, for each agent, a *local protocol* is defined, which models the program the agent is executing. Formally, a *local protocol* $P_i : L_i \to 2^{Act_i}$ for each agent i, selects actions which can be executed at each local state. For each agent i, we define an evolution (partial) function $t_i : L_i \times A_i \to L_i$, where $t_i(l_i, \epsilon) = l_i$ if $\epsilon \in P_i(l_i)$, for each $l_i \in L_i$.

A *global state* $q = (l_1, \ldots, l_n) \in L_1 \times \ldots \times L_n$ is a tuple of local states for all the agents in the MAS. By $q^i = l_i$ we mean the local component of agent $i \in \mathcal{A}$ in q. Now the global transitions are defined.

Definition 1 (Interleaved semantics)**.** *Let $St = L_1 \times \ldots \times L_n$ be a set of global states. The global interleaved evolution function $T : St \times Act \to St$ is defined*

[1] Note that we do not consider the environment component, which may be added with no technical difficulty.

as follows: $T(q,a) = q_1$ *iff* $t_i(q^i,a) = q_1^i$ *for all* $i \in Agent(a)$*, and* $q^i = q_1^i$ *for all* $i \in \mathcal{A} \setminus Agent(a)$*. The above is denoted as* $q \xrightarrow{a} q_1$.

Notice that $q \xrightarrow{\epsilon} q$ if $\epsilon \in P_i(q^i)$ for each $i \in \mathcal{A}$. The global transition relation is assumed to be total, i.e., for each $q \in St$ there exists an $a \in Act$ such that $q \xrightarrow{a} q_1$, for some $q_1 \in St$. An infinite sequence of global states and actions $\pi = q_0a_0q_1a_1q_2\ldots$ is called an (interleaved) path, starting at q_0 if there is a sequence of global transitions from q_0 onwards, i.e., if $q_i \xrightarrow{a_i} q_{i+1}$ for every $i \geq 0$. By $Act(\pi) = a_0a_1a_2\ldots$ we denote the sequence of actions of π, while by $\Pi(q)$ - the set of all interleaved paths starting at q.

2.2 Interleaved Interpreted Systems

In order to define a semantics of $\mathbf{ATL}^*$ we need a notion of model.

Definition 2 (Interleaved Interpreted System)**.** *Let* PV *be a set of propositional variables. An interleaved interpreted system (IIS) or a model, is a 4-tuple* $M = (St, \iota, T, V)$*, where* St *is a set of global states,* $\iota \in St$ *is an initial (global) state,* T *is the global transition relation, and* $V : St \to 2^{PV}$ *is a valuation function.*

Figure 1 presents the three agents of the interleaved interpreted system (the untimed version of the original Train-Gate-Controller (TGC) [4, 27]): a controller and two trains. Each train runs on a circular track and both tracks pass through a narrow tunnel (state "T"), allowing one train only to go through it (to state "A" - (Away) at any time. The controller operates the signal (Green ("G") and Red ("R")) to let trains enter and leave the tunnel. In the figure, the initial states of the controller and the train are "G" and "W" (Waiting) respectively, and the transitions with the same label are synchronised. Silent ϵ actions are omitted in the figure.

Definition 3 (Reduced Model)**.** *Given two IIS (models)* $M = (St, \iota, T, V)$ *and* $M' = (St', \iota', T', V')$*. If* $St' \subseteq St$*,* $\iota' = \iota$*,* T *is an extension of* T'*, and* $V' = V|St'$*, then we write* $M' \subseteq M$ *and call* M' *a submodel of* M*, and* M' *- a* reduced *model of* M*.*

It is easy to show that for each $q \in St'$ we have $\Pi'(q) \subset \Pi(q)$, where $\Pi(q)$ ($\Pi'(q)$), is the set of al interleaved paths of M (M', resp.) starting at q.

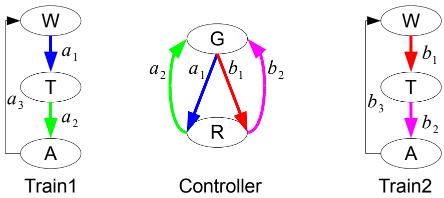

Figure 1: An IIS of TGC composed of two trains [48, 49]

In order to generate reduced models we will need a notion of invisibility of actions with respect a subset PV' of PV.

Definition 4. *Given a model* $M = (St, \iota, T, V)$. *An action* $a \in Act$ *is* invisible *for* PV' *if for each two global states* $q, q' \in St$*: if* $q \xrightarrow{a} q'$*, then* $V(q) \cap PV' = V(q') \cap PV'$.
The set of all actions invisible *for* PV' *is denoted by* $Invis_{PV'}$ *and its closure by* $Vis_{PV'} = Act \setminus Invis_{PV'}$.

Intuitively, invisible actions do not change valuations of the propositions of PV' in M.

3 Reasoning about Agents' Abilities

Many important properties in a MAS can be specified in reference to the existence strategic ability (or inability) of some agents to achieve a given goal. Examples include the ability of a voter to cast her vote according to her intent in an election, the inability of a coercer to disturb the outcome of the election by coercion and/or vote buying, the ability of two parties to successfully communicate or sign a contract, and the existence of a suitable collective strategy for trains in a railway network so that neither a crash nor a deadlock can occur. Such properties can be specified by formulae of alternating-time logic (**ATL**). The semantics of **ATL** is typically defined for models of synchronous systems. In this section, we show how the semantics can be redefined for interleaved interpreted systems.

3.1 Alternating-Time Temporal Logic

Alternating-time temporal logic [3, 6] generalizes the branching-time temporal logic $\mathbf{CTL}$ [15] by replacing path quantifiers E, A with *strategic modalities* $\langle\!\langle A \rangle\!\rangle$. Informally, $\langle\!\langle A \rangle\!\rangle\gamma$ expresses that the group of agents A has a collective strategy to enforce temporal property γ. Formulas of $\mathbf{ATL}$ make use of temporal operators: "X" ("in the next state"), "G" ("always from now on"), "F" ("now or sometime in the future"), U ("strong until"), and R ("release"). The logic comes in several syntactic variants, the most popular of which are $\mathbf{ATL}^*$ and "vanilla $\mathbf{ATL}$" (the latter often called simply "$\mathbf{ATL}$").

Definition 5 (Syntax of $\mathbf{ATL}^*$)**.** *Let PV be a set of propositional variables and $\mathcal{A}$ the set of all agents. The language of $\mathbf{ATL}^*$ is defined by the following grammar:*

$$\varphi ::= \mathsf{p} \mid \neg\varphi \mid \varphi \wedge \varphi \mid \langle\!\langle A \rangle\!\rangle\gamma,$$
$$\gamma ::= \varphi \mid \neg\gamma \mid \gamma \wedge \gamma \mid \mathrm{X}\gamma \mid \gamma \,\mathrm{U}\, \gamma.$$

where $\mathsf{p} \in PV$ and $A \subseteq \mathcal{A}$.

Disjunction and Boolean constants are defined as usual. The "release" operator can be defined as $\gamma_1 \,\mathrm{R}\, \gamma_2 \equiv \neg((\neg\gamma_1) \,\mathrm{U}\, (\neg\gamma_2))$. The "sometime" and "always" operators can be defined as $\mathrm{F}\gamma \equiv \top \,\mathrm{U}\, \gamma$ and $\mathrm{G}\gamma \equiv \bot \,\mathrm{R}\, \gamma$.

Definition 6 (Syntax of $\mathbf{ATL}$)**.** *In "vanilla $\mathbf{ATL}$," every occurrence of a strategic modality is immediately followed by a temporal operator. Note that, in that case, "release" is not definable from "until" anymore [44], and it must be added to the syntax as another primitive operator. Formally, the language of $\mathbf{ATL}$ is defined by the following grammar:*

$$\varphi ::= \mathsf{p} \mid \neg\varphi \mid \varphi \wedge \varphi \mid \langle\!\langle A \rangle\!\rangle\mathrm{X}\varphi \mid \langle\!\langle A \rangle\!\rangle\mathrm{G}\varphi \mid \langle\!\langle A \rangle\!\rangle\varphi \,\mathrm{U}\, \varphi.$$

In the rest of the paper, we will be mainly interested in formulae that do not use the next step operator X, and do not contain nested strategic modalities. We denote the corresponding subsets of $\mathbf{ATL}^*$ and $\mathbf{ATL}$ by $\mathbf{sATL}^*$ ("simple $\mathbf{ATL}^*$") and $\mathbf{sATL}$ ("simple $\mathbf{ATL}$"), respectively. Moreover, $\mathbf{1ATL}^*$ denotes the fragment of $\mathbf{sATL}^*$ that admits only formulae consisting of a strategic modality, followed by an $\mathbf{LTL}$ formula (i.e., $\langle\!\langle A \rangle\!\rangle\gamma$, where $\gamma \in \mathbf{LTL}$).

3.2 Strategies and Outcomes

Let $M = (St, \iota, T, V)$ be an IIS and $i \in \mathcal{A}$. A *strategy* of agent i is a conditional plan that specifies what i is going to do in any potential situation. A number of semantic variations are possible. Here, we follow Schobbens [62], and adopt his taxonomy of four "canonical" strategy types: IR, iR, Ir, and ir. In the notation, *R* (resp. *r*) stands for perfect (resp. imperfect) *recall*, and *I* (resp. *i*) refers to perfect (resp. imperfect) *information*.

In this paper, we consider strategies of types Ir and ir. Formally:

- A *memoryless perfect information strategy for agent* i is a function $\sigma_i \colon St \to A_i$ such that $\sigma_i(q) \in P_i(q^i)$ for each state $q \in St$. We denote the set of such strategies by Σ_{Ir}.

- A *memoryless imperfect information strategy for agent* i is a function $\sigma_i \colon St \to A_i$ such that $\sigma_i(q) \in P_i(q^i)$ for each state $q \in St$, and for each $q, q_1 \in St$ if $q^i = q_1^i$, then $\sigma_i(q) = \sigma_i(q_1)$. Equivalently, the strategy can be defined as $\sigma_i \colon L_i \to A_i$ such that $\sigma_i(l) \in P_i(l)$. We denote the set of such strategies by Σ_{ir}.

Thus, an Ir strategy can assign different actions of agent i to any two global states, while in an ir strategy the agent's choices can only depend on the local state (i.e., the location) of the agent.

A *joint strategy* σ_A for a coalition $A \subseteq \mathcal{A}$ is a tuple of strategies, one per agent $i \in A$. Let $A = \{1, \ldots, k\}$ for some $k \in \mathcal{N}$ and $\sigma_A = (\sigma_1, \ldots, \sigma_k)$ be a joint strategy for A. For each $i \in \mathcal{N}$ and $q \in St$ we denote $\sigma_A(q) = (\sigma_1(q), \ldots, \sigma_k(q))$. By $Act(\sigma_A)$ we denote the set of actions assigned to the states by the strategies of σ_A. We will consider two types of joint strategies: Ir-joint strategies consisting of memoryless perfect information strategies and ir-joint strategies consisting of memoryless imperfect information strategies.

Definition 7 (Outcome)**.** *The* outcome *of a joint strategy σ_A in a state $q \in St$ is the set $out_M(q, \sigma_A) \subseteq St^\omega$ such that $\pi \in out_M(q, \sigma_A)$ iff $\pi[0] = q$ and for each $i \in \mathcal{N}$:*

- $\pi[i] \xrightarrow{a_i} \pi[i+1]$ *for some $a_i \in Act$,*

- *for each $j \in A$ if $j \in Agent(a_i)$, then $a_i \in \sigma_j(\pi[i])$,* Intuitively, the outcome of a joint strategy σ_A in a state q is the set of all the possible paths that can occur when the agents of coalition A execute the strategy σ_A and each other agent, in $\overline{A}$, follows its own protocol.

3.3 Asynchronous Semantics of ATL and ATL*

The semantics of $\mathbf{ATL}^*$, parameterised with the strategy type Y, is defined below.

Definition 8 (Semantics of $\mathbf{ATL}^*$)**.** *Let $M = (St, \iota, T, V)$ be an IIS, $q \in St$, $A \subseteq \mathcal{A}$, and $Y \in \{\mathrm{Ir}, \mathrm{ir}\}$. The Y-semantics of $\mathbf{ATL}^*$ and $\mathbf{ATL}$ is given by the following clauses:*

$\mathcal{M}, q \models_Y \mathsf{p}$ *iff* $\mathsf{p} \in V(q)$;

$\mathcal{M}, q \models_Y \neg\varphi$ *iff* $\mathcal{M}, q \not\models_Y \varphi$;

$\mathcal{M}, q \models_Y \varphi_1 \wedge \varphi_2$ *iff* $\mathcal{M}, q \models_Y \varphi_1$ *and* $\mathcal{M}, q \models_Y \varphi_2$;

$\mathcal{M}, q \models_Y \langle\langle A \rangle\rangle\gamma$ *iff there is a joint Y-strategy σ_A for agents A such that, for each path $\pi \in out_{\mathcal{M}}(q, \sigma_A)$, we have $\mathcal{M}, \pi \models_Y \gamma$;*

$\mathcal{M}, \pi \models_Y \varphi$ *iff* $\mathcal{M}, \pi[0] \models_Y \varphi$;

$\mathcal{M}, \pi \models_Y \neg\gamma$ *iff* $\mathcal{M}, \pi \not\models_Y \gamma$;

$\mathcal{M}, \pi \models_Y \gamma_1 \wedge \gamma_2$ *iff* $\mathcal{M}, \pi \models_Y \gamma_1$ *and* $\mathcal{M}, \pi \models_Y \gamma_2$;

$\mathcal{M}, \pi \models_Y \mathrm{X}\gamma$ *iff* $\mathcal{M}, \pi[1, \infty] \models_Y \gamma$;

$\mathcal{M}, \pi \models_Y \gamma_1 \,\mathrm{U}\, \gamma_2$ *iff there is $i \geq 0$ such that $\mathcal{M}, \pi[i, \infty] \models_Y \gamma_2$ and $\mathcal{M}, \pi[j, \infty] \models_Y \gamma_1$ for all $0 \leq j < i$.*

The semantics of "vanilla $\mathbf{ATL}$" can be given entirely with respect to states.

Definition 9 (State-based semantics of $\mathbf{ATL}$)**.** *The Y-semantics of $\mathbf{ATL}$ can be equivalently defined by the following clauses:*

$\mathcal{M}, q \models_Y \mathsf{p}$ *iff* $\mathsf{p} \in V(q)$;

$\mathcal{M}, q \models_Y \neg\varphi$ *iff* $\mathcal{M}, q \not\models_Y \varphi$;

$\mathcal{M}, q \models_Y \varphi_1 \wedge \varphi_2$ *iff* $\mathcal{M}, q \models_Y \varphi_1$ *and* $\mathcal{M}, q \models_Y \varphi_2$;

$\mathcal{M}, q \models_Y \langle\!\langle A \rangle\!\rangle \mathrm{X}\varphi$ *iff there is a* Y*-strategy* σ_A *such that, for each* $\pi \in out_{\mathcal{M}}(q, \sigma_A)$*, we have* $\mathcal{M}, \pi[1] \models_Y \varphi$*;*

$\mathcal{M}, q \models_Y \langle\!\langle A \rangle\!\rangle \varphi_1 \,\mathrm{U}\, \varphi_2$ *iff there is a* Y*-strategy* σ_A *such that, for each* $\pi \in out_{\mathcal{M}}(q, \sigma_A)$*, there exists* $i \geq 0$ *with* $\mathcal{M}, \pi[i] \models_Y \varphi_2$ *and* $\mathcal{M}, \pi[j] \models_Y \varphi_1$ *for all* $0 \leq j < i$*;*

$\mathcal{M}, q \models_Y \langle\!\langle A \rangle\!\rangle \varphi_1 \,\mathrm{R}\, \varphi_2$ *iff there is a* Y*-strategy* σ_A *such that, for all* $\pi \in out_{\mathcal{M}}(q, \sigma_A)$ *and* $i \geq 0$*, either* $\mathcal{M}, \pi[i] \models_Y \varphi_2$ *or* $\mathcal{M}, \pi[j] \models_Y \varphi_1$ *for some* $0 \leq j < i$*.*

For a syntax $\mathcal{L}$ and the semantic relation $\models_Y$, we will denote the logical system $(\mathcal{L}, \models_Y)$ by $\mathcal{L}_Y$. Thus, $\mathbf{ATL}_{\mathrm{Ir}}$ is the "vanilla $\mathbf{ATL}$" with memoryless perfect information semantics, $\mathbf{sATL}^*_{\mathrm{ir}}$ is the "simple $\mathbf{ATL}^*$" with memoryless imperfect information strategies, and so on.

Remark 1. *We observe that the relation* $\models_{\mathrm{ir}}$ *captures the "objective" notion of ability under imperfect information [32, 9]. That is,* $\langle\!\langle A \rangle\!\rangle\gamma$ *holds iff agents in* A *have a collective strategy to enforce* γ *from the current global state of the system. We expect to obtain analogous results for the semantic variant based on "subjective" ability [62, 34, 9], but a detailed study of the latter case is outside the scope of this report.*

4 Model Checking sATL and sATL^* for Asynchronous MAS

In this work, we focus on simple specifications of strategic ability, i.e., ones that can be written formally without nesting strategic modalities. We believe that an overwhelming majority of properties, relevant in actual application domains, follow that pattern. We usually want to require (or ask if) a given player has a strategy to eventually win the game, two trains can persistently avoid the crash, Alice and Bob can exchange a secret without Cathy learning it on the way, etc. The three example properties can be tentatively specified by the following formulae:

1. $\langle\!\langle i \rangle\!\rangle \mathrm{F}\mathsf{win_i}$,
2. $\langle\!\langle t_1, t_2 \rangle\!\rangle \mathrm{G}\neg\mathsf{crash}$,

3. $\langle\!\langle a,b\rangle\!\rangle\big(\mathrm{F}(\mathsf{knowsSecr_a} \land \mathsf{knowsSecr_b}) \land \mathrm{G}\neg\mathsf{knowsSecr_c}\big)$.

Note that (1) and (2) are formulae of $\mathbf{sATL}$ (in fact, $\mathbf{1ATL}$), while (3) is a formula of $\mathbf{sATL}^*$. Also, specification (3) suggests that many interesting properties can be more conveniently specified with a combination of strategic and epistemic modalities, which seems an interesting path for future work.

Note also that, in all realistic scenarios, players have only partial knowledge of the current global state of the world during the interaction. Thus, a semantics with imperfect information strategies must be used. Since verification of $\mathbf{ATL}$ with imperfect information and perfect recall is undecidable [18], we focus on the memoryless imperfect information semantics $\models_{\mathrm{ir}}$.

In this section, we establish the complexity of model checking for some relevant fragments of $\mathbf{sATL}^*_{\mathrm{ir}}$. We observe that the complexity can be given with respect to the logical *model* of the system (i.e., an interleaved interpreted system, cf. Section 2.2), or the usual *representation* of the system (in our case, an asynchronous automata network, cf. Section 2.1). We give both kinds of results here. We also briefy look at the *program complexity* of model checking, i.e., the complexity of the problem when the input formulae are of fixed or bounded length.

4.1 Model checking $\mathbf{1ATL}_{\mathrm{ir}}$

We begin by looking at the verification complexity for simplest specifications, consisting of a single strategic modality $\langle\!\langle A\rangle\!\rangle$ immediately followed by a single temporal modality.

Proposition 2. *Model checking* $\mathbf{1ATL}_{\mathrm{ir}}$ *is* $\mathbf{NP}$*-complete in the size of the model and the length of the formula. It remains* $\mathbf{NP}$*-complete even for formulae of bounded length.*

Proof. (sketch) Analogous to the result in [62] for $\langle\!\langle\Gamma\rangle\!\rangle$-$\mathbf{ATL}_{\mathrm{ir}}$.

For the upper bound, observe that model checking of $\langle\!\langle A\rangle\!\rangle\gamma$ in M,q can be done by (1) guessing a uniform strategy s_A, (2) pruning M according to s_A, and (3) model checking the $\mathbf{CTL}$ formula $\mathsf{A}\gamma$ ("for all paths, γ") in state q of the resulting model. Since s_A is of at most linear size with respect to $|M|$, and model checking of $\mathsf{A}\gamma$ can be done in deterministic polynomial time w.r.t. $|M|$, we obtain the bound.

For the lower bound, we use the reduction from [62] of the Boolean satisfiability problem (SAT) to model checking formula $\langle\!\langle 1 \rangle\!\rangle \mathrm{F}\mathsf{yes}$ in a single-agent model (note that single-agent systems can be seen as special cases of both synchronous and asynchronous systems). Note that the lower bound does not rely on the length of the formula, as formulae of length 3 are sufficient to construct the reduction. □

Proposition 3. *Model checking* $\mathbf{1ATL}_{\mathrm{ir}}$ *is* $\mathbf{PSPACE}$*-complete in the size of the representation (even for formulae of bounded length).*

Proof. (sketch) For the upper bound, observe that model checking of formula $\langle\!\langle A \rangle\!\rangle\gamma$ in representation (automata network) R can be done by: (1) guessing a uniform s_A as a deterministic restriction of the protocols P_i, $i \in \mathcal{A}$, (2) pruning M, and (3) model checking the $\mathbf{LTL}$ formula γ in the resulting representation R'. Since the size of s_A is linear with respect to $|R|$, and model checking $\mathbf{LTL}$ is in $\mathbf{PSPACE}$ with respect to $|R|$ (cf. [61]), we obtain the bound.

To prove the lower bound, we adapt the construction from [41, Theorem 6.1]. Given a Turing machine T with space complexity $s(n)$, we construct the concurrent program $P(T)$ as in [41, Theorem 6.1]. To obtain an asynchronous MAS P' with a similar behavior, we sequentialize the concurrent actions of modules in $P(T)$ by adding an extra "synchronizer" module which enforces that each agent $i \in \{1, \ldots, n\}$ takes the ith step in each "execution cycle." That is, every concurrent transition in $P(T)$ is decomposed into a sequence of n asynchronous transitions in P'. Now, there exists a computation of T on the empty tape which eventually reaches an accepting state iff $P(T) \models_{\mathbf{CTL}} \mathsf{EFaccept}$ iff $P' \not\models_{\mathrm{ir}} \langle\!\langle \emptyset \rangle\!\rangle \mathrm{G}\neg\mathsf{accept}$. This way we obtain the $\mathbf{co}$-$\mathbf{PSPACE}$-hardness for $\mathbf{1ATL}_{\mathrm{ir}}$. Since $\mathbf{co}$-$\mathbf{PSPACE}$ = $\mathbf{PSPACE}$, the lower bound follows.

Again, we note that the reduction does not rely on the length of the formula. □

4.2 Model checking $\mathbf{sATL}_{\mathrm{ir}}$

The verification complexity for Boolean combinations of formulae from $\mathbf{1ATL}$ is almost the same.

Proposition 4. *Model checking* $\mathbf{sATL_{\mathrm{ir}}}$ *is* $\mathbf{NP}$*-hard and in* $\mathbf{\Theta_2^P}$ *in the size of the model and the length of the formula (even for formulae of bounded length).*[2]

Proof. (sketch) The lower bound follows from Proposition 2.

The following algorithm for checking φ in M, q demonstrates the upper bound. First, the nondeterministic algorithm in Proposition 2 is used as an oracle that determines the truth value for each subformula $\langle\langle A\rangle\rangle\gamma$ of φ. It is easy to see that the oracle is called at most $|\varphi|$ times, and the input in the next call does not depend on the output of the preceding calls. Finally, based on the output of the calls, the value of φ is calculated according to the truth tables of Boolean operators. □

Proposition 5. *Model checking* $\mathbf{sATL_{\mathrm{ir}}}$ *is* $\mathbf{PSPACE}$*-complete in the size of the representation and the length of the formula (even for formulae of bounded length).*

Proof. (sketch) The lower bound follows from Proposition 3. For the upper bound, we use the algorithm from Proposition 4, but with the algorithm from Proposition 3 as the oracle. Since $\mathbf{P^{PSPACE}} = \mathbf{PSPACE}$, we obtain the result. □

4.3 Model checking $\mathbf{sATL^*_{\mathrm{ir}}}$ and $\mathbf{1ATL^*_{\mathrm{ir}}}$

Finally, we examine the complexity of verification for specifications with arbitrary $\mathbf{LTL}$ subformulae.

Proposition 6.

1. *Model checking* $\mathbf{sATL^*_{\mathrm{ir}}}$ *and* $\mathbf{1ATL^*_{\mathrm{ir}}}$ *is* $\mathbf{PSPACE}$*-complete in the size of the model and the length of the formula.*

2. *For formulae of bounded length, the problem is* $\mathbf{NP}$*-complete for* $\mathbf{1ATL^*_{\mathrm{ir}}}$ *and between* $\mathbf{NP}$ *and* $\mathbf{\Theta_2^P}$*-complete for* $\mathbf{sATL^*_{\mathrm{ir}}}$.

[2]Where $\mathbf{\Theta_2^P} = \mathbf{P}^{||\mathbf{NP}}$ is the class of problems solvable by a deterministic polynomial-time Turing machine that can make polynomially many nonadaptive calls to an $\mathbf{NP}$ oracle.

Proof. (sketch) For (1), the lower bound follows from the $\mathbf{PSPACE}$-completeness of $\mathbf{LTL}$ model checking [61]. The upper bound can be obtained by polynomially many calls to an $\mathbf{LTL}$ model checking algorithm, one per strategic subformula in φ (recall that $\mathbf{P}^{\mathbf{PSPACE}} = \mathbf{PSPACE}$), and computing the Boolean combination.

For (2), the lower bound follows from Proposition 2. The inclusion in $\mathbf{NP}$ for $\mathbf{1ATL}^*_{\mathrm{ir}}$ can be obtained by an algorithm similar to that in Proposition 2, only an $\mathbf{LTL}$ rather than $\mathbf{CTL}$ model checker is called. Since $\mathbf{LTL}$ model checking is $\mathbf{NLOGSPACE}$-complete for formulae of bounded size [61], and $\mathbf{NLOGSPACE} \subseteq \mathbf{P}$, the upper bound follows.

The upper bound for $\mathbf{sATL}^*_{\mathrm{ir}}$ is obtained by an algorithm similar to that in Proposition 4, only an $\mathbf{LTL}$ rather than $\mathbf{CTL}$ model checker is called inside the oracle. □

Proposition 7. *Model checking* $\mathbf{sATL}^*_{\mathrm{ir}}$ *and* $\mathbf{1ATL}^*_{\mathrm{ir}}$ *is* $\mathbf{PSPACE}$*-complete in the size of the representation and the length of the formula (even for formulae of bounded length).*

Proof. (sketch) The lower bounds follow from Proposition 3. The upper bounds can be obtained by the same algorithm as for Proposition 5. □

4.4 Discussion

Models (interleaved interpreted systems) are usually exponentially lager than the automata network representations from which they arise. Thus, for practical verification it is essential to provide the input to the model checker using the latter rather than the former. Unfortunately, the above complexity results strongly suggest that model checking fragments of $\mathbf{sATL}^*_{\mathrm{ir}}$ on automata networks is hard, and the size of the representation is the main factor for this hardness. Thus, it is essential to use as small a representation as possible. If the input is given beforehand (e.g., prepared by the user), then any *reduction* of the representation increases the likelihood that the verification task becomes feasible.

In the next sections, we recall the idea of *partial order reduction*, very important for verification of asynchronous systems, and show how it can be used to model-check $\mathbf{sATL}^*_{\mathrm{ir}}$ and its fragments.

5 Equivalences for $\mathbf{LTL}_{-\mathbf{X}}$

Partial order reductions have been defined for temporal logics without the next step operator X since it is known how to generate reduced models preserving either stuttering trace equivalence (for $\mathbf{LTL}_{-\mathbf{X}}$) or stuttering bisimulation (for $\mathbf{CTL}^*_{-\mathbf{X}}$). Since stuttering trace equivalence is less discriminative than stuttering bisimulation, partial order reductions preserving $\mathbf{LTL}_{-\mathbf{X}}$ are more efficient than these for $\mathbf{CTL}^*_{-\mathbf{X}}$. Since $\mathbf{ATL}^*_{-\mathbf{X}}$ has a more distinguishing power than $\mathbf{CTL}^*_{-\mathbf{X}}$, one can expect that the equivalence preserving $\mathbf{ATL}^*_{-\mathbf{X}}$ would be very discriminative, which would likely result in inefficient reductions. Therefore, aware of the above and motivated by practical applications, in this paper we take another route. We are investigating subsets of $\mathbf{ATL}^*_{-\mathbf{X}}$ for which known partial order reduction methods apply [54, 58, 22, 48, 49].

5.1 Stuttering (trace) equivalence

So, we start with definitions of stuttering equivalence and stuttering trace equivalence.

Definition 10 (Stuttering Equivalence)**.** *A path π in M and a path π' in M' are called* stuttering equivalent*, denoted $\pi \equiv_s \pi'$, if there exists a partition $B_1, B_2 \ldots$ of the states of π, and a partition $B_1', B_2' \ldots$ of the states of π' such that for each $j \geq 0$ we have that B_j and B'_j are nonempty and finite, and for every state q in B_j and every state q' in B_j' we have $V(q) = V'(q')$.*

Notice that in the above definition in each block B all the states share the same valuation.

Definition 11 (Stuttering Trace Equivalence)**.** *Two states q in M and q' in M' are said to be* stuttering trace equivalent*, denoted $q \equiv_s q'$, if*

1. *for each path π in M starting at q, there is a path π' in M' starting at q' such that $\pi \equiv_s \pi'$;*
2. *for each path π' in M' starting at q', there is a path π in M starting at q such that $\pi' \equiv_s \pi$.*

Two models M and M' are stuttering trace equivalent*, denoted $M \equiv_s M'$, if $\iota \equiv_s \iota'$.*

5.2 Preserving $\mathbf{LTL}_{-\mathbf{X}}$

The following theorem connects $\mathbf{LTL}_{-\mathbf{X}}$ with stuttering trace equivalence:

Theorem 8. *Let M and M' be two stuttering trace equivalent models, where $M' \subseteq M$. Then, $M, \iota \models \varphi$ iff $M', \iota' \models \varphi$, for any $\mathbf{LTL}_{-\mathbf{X}}$ formula φ over PV.*

Proof. See [25]. □

6 Partial Order Reductions for $\mathbf{sATL}^*_{\mathrm{ir}}$

In what follows we propose how to obtain partial order reduction for $\mathbf{sATL}^*_{\mathrm{ir}}$ and its fragments, with very promising results. Interestingly, it turns out that our approach does not apply to $\mathbf{sATL}^*_{\mathrm{Ir}}$; we show it the end of this section. This suggests that $\mathbf{ATL}$ with imperfect information, besides conceptual advantage, can also offer some technical advantage over $\mathbf{ATL}$ with perfect information.

6.1 Traces

Partial order reductions are based on Mazurkiewicz traces as introduced in [51] and used in e.g. [52, 53]. Consider two words $w, w' \in Act^*$. We say that $w \sim_I w'$ iff $w = w_1 ab w_2$ and $w' = w_1 ba w_2$, for some $w_1, w_2 \in Act^*$ and $(a, b) \in I$. Let $\equiv_I$ be the reflexive and transitive closure of $\sim_I$. By (finite) traces we mean equivalence classes of $\equiv_I$, denoted by $[w]_{\equiv_I}$. Formally, a trace $[w]_{\equiv_I} = \{w' \in Act^* \mid w' \equiv_I w\}$.

To define infnite traces we need more definitions. For $v, v' \in Act^\omega$ the relation $\leq_I$ is defined as follows: $v \leq_I v'$ iff $\forall u \in Pref(v)\ \exists w \in Pref(v')\ \exists z \in Act* \ (w \equiv_I z \wedge u \in Pref(z))$. That means, each finite prefix of v is a prefix of a permutation (under commuting adjacent independent actions) of some prefix of v'.

Infinite traces are defined as equivalence classes of the following relation $\equiv_I^\omega$, where $v \equiv_I^\omega v'$ iff $v \leq_I v'$ and $v' \leq_I v$, denoted by $[v]_{\equiv_I^\omega}$, where $v \in Act^\omega$.

Lemma 9. *Let M be an IIS and $PV' \subseteq PV$. Consider two paths $\pi, \pi' \in St^\omega$ starting at the same state such that $Act(\pi) = w$ and $Act(\pi') = w'$, and $w \equiv_I^\omega w'$. Assume that: **) for any two actions a, b occurring in w or w' if $(a, b) \in I$, then either $a \in Invis_{PV'}$ or $b \in Invis_{PV'}$. Then, π and π' are stuttering equivalent wrt. PV'.*

Proof. See [54]. □

We can always make I to satisfy **) by restricting it to $I \setminus (Vis_{PV'} \times Vis_{PV'})$.

Since stuttering equivalence preserves $\mathbf{LTL_{-X}}$, the above lemma implies that the paths over representatives of the same infinite trace cannot be distinguished by any $\mathbf{LTL_{-X}}$ formula using propositions of PV'. This fact is used for defining partial order reductions for $\mathbf{LTL_{-X}}$. Rather than generating the IIS (model) M for a MAS, one can generate a reduced model M' satisfying the following property:
(*) for each $\pi \in \Pi(\iota)$, there is $\pi' \in \Pi'(\iota)$ such that $Act(\pi) \equiv_I^\omega Act(\pi')$.
The reduced model M' preserves the $\mathbf{LTL_{-X}}$ formulas over PV'.

Our aim is to show that M' preserves also the $\mathbf{sATL}^*_{\mathrm{ir}}$ formulas over PV'. To this aim we show that each set $out_M(q, \sigma_A)$ is trace complete in the sense that with each path π s.t. $Act(\pi) = w$, it contains a path over any $w' \in [w]_{\equiv_I^\omega}$.

Lemma 10. *Let $\pi \in out_M(\iota, \sigma_A)$ and $Act(\pi) = w$. Then, for each $w' \in [w]_{\equiv_I^\omega}$ there is $\pi' \in out_M(\iota, \sigma_A)$ such that $Act(\pi') = w'$.*

Proof. Consider a MAS in which the protocol of each agent $i \in A$ is equal to σ_i. Then, the set of paths $\Pi(\iota)$ of IIS for MAS is trace complete. But, this set of paths is equal to $out_M(\iota, \sigma_A)$, which ends the proof.

□

The above lemma suggests a method of partial order reduction for the formulas of $\mathbf{sATL}^*_{\mathrm{ir}}$.

Lemma 11. *Let M be a model, $PV' \subseteq PV$, and M' be a reduced model satisfying the property *). Then, for each strategy σ_A, for each $\pi \in out_M(\iota, \sigma_A)$ there is $\pi' \in out_{M'}(\iota, \sigma_A)$ such that $Act(\pi) \equiv_I^\omega Act(\pi')$.*

Proof. Follows easily from the property *) of M and Lemma 10. □

Theorem 12. *Let M be a model, $PV' \subseteq PV$, and M' be a reduced model satisfying the property *). For each $\mathbf{sATL}^*_{\mathrm{ir}}$ formula φ over PV' we have:*

$$M, \iota \models \varphi \Leftrightarrow M', \iota' \models \varphi$$

Proof. The proof is by induction on the complexity of a formula. Consider $\varphi = \langle\!\langle A \rangle\!\rangle \gamma$.

$M, \iota \models \langle\!\langle A \rangle\!\rangle \gamma$ iff

there is a joint strategy σ_A such that for each $\pi \in out_M(\iota, \sigma_A)$ we have $M, \pi \models \gamma$ iff (by Lemmas 11 and 9)

there is a joint strategy σ_A such that for each $\pi \in out_{M'}(\iota, \sigma_A)$ we have $M', \pi \models \gamma$ iff

$M', \iota \models \langle\!\langle A \rangle\!\rangle \gamma$.

The cases of negation and conjunction are straightforward. □

The next example shows that Lemma 10 does not hold for memoryless perfect information strategies.

Example 1. *Consider the MAS composed of two agents $\{1, 2\}$ such that*

- $L_1 = \{l_1^1, l_1^2\}$, $L_2 = \{l_2^1, l_2^2\}$,
- $Act_1 = \{\epsilon, a\}$, $Act_2 = \{\epsilon, b\}$,
- $P_1(l_1^1) = \{a, \epsilon\}$, $P_1(l_1^2) = \{\epsilon\}$,
- $P_2(l_2^1) = \{b\}$, $P_2(l_2^2) = \{\epsilon\}$,
- $t_1(l_1^1, a) = l_1^2$, $t_2(l_2^1, b) = l_2^2$.

Define an Ir-strategy for each agent:

1: $\sigma_1(l_1^1, l_2^1) = a$, $\sigma_1(l_1^1, l_2^2) = \sigma_1(l_1^2, l_2^2) = \epsilon$,

2: $\sigma_2(l_1^1, l_2^1) = \sigma_2(l_1^2, l_2^1) = b$, $\sigma_2(l_1^2, l_2^2) = \epsilon$.

It is easy to see that $out((l_1^1, l_2^1), \sigma_{\{1,2\}})$ is not trace complete. Notice that $(a, b) \in I$, but while $out((l_1^1, l_2^1), \sigma_{\{1,2\}})$ contains the path over $ab(\epsilon)^\omega$, it does not contain any path over $ba(\epsilon)^\omega$.

6.2 Algorithms for Partial Order Reduction

As mentioned above, the idea of verification by model checking with partial order reductions is to define an algorithm reducing the size of models while preserving satisfaction for a class of formulas. This requires a notion of equivalence between models. For the case of $\mathbf{sATL}^*_{\mathrm{ir}}$ we know that the notion of stuttering trace equivalence presented above suffices. Traditionally, in partial order reduction the exploration is carried out either by depth-first-search (DFS) (see [22]), or double-depth-first-search (DDFS) [16].

In this context DFS is used to search states and transitions that will make up the reduced model by exploring systematically the possible computation tree and selecting only some of the posssible states and transitions generated. In the following, a stack represents the path $\pi = g_0 a_0 g_1 a_1 \cdots g_n$ currently being visited. For the top element of the stack g_n the following three operations are computed in a loop:

1. The set $en(g_n) \subseteq Act$ of enabled actions (not including the ϵ action) is identified and a subset $E(g_n) \subseteq en(g_n)$ of possible actions is heuristically selected (see below).

2. For any action $a \in E(g_n)$ compute the successor state g' such that $g_n \stackrel{a}{\rightarrow} g'$, and add g' to the stack thereby generating the path $\pi' = g_0 a_0 g_1 a_1 \cdots g_n a g'$. Recursively proceed to explore the submodel originating at g' in the same way by means of the present algorithm beginning at step 1.

3. Remove g_n from the stack.

The algorithm begins with a stack comprising of the initial state and terminates when the stack is empty. The model generated by the algorithm is a submodel of the full one. Its size crucially depends on the ratio $E(g)/en(g)$. The choice of $E(q)$ is constrained to preserve the stuttering trace equivalence between the original and a submodel generated.

6.3 Preserving $\mathbf{sATL}^*_{\mathrm{ir}}$

In the sequel, let ϕ be an $\mathbf{sATL}^*_{\mathrm{ir}}$ formula to be checked over the model M and let M' be a submodel of M, generated by the algorithm. The states and the actions connecting states in M' define a directed *state graph*. We give conditions defining a heuristics for the selection of $E(g)$ (such that $E(g) \neq en(g)$) while visiting state g in the algorithm below.

C1 No action $a \in \mathit{Act} \setminus E(g)$ that is dependent (see Subsection 2.2) on an action in $E(g)$ can be executed before an action in $E(g)$ is executed.

C2 For every cycle in the constructed state graph there is at least one node g in the cycle for which $E(g) = en(g)$, i.e., for which all the successors of g are expanded.

C3 All actions in $E(g)$ are invisible (see Subsection 2.2).

The conditions $\mathbf{C1} - \mathbf{C3}$ are inspired from [54].

Theorem 13. *Let M be a model and $M' \subseteq M$ be the reduced model generated by the DFS algorithm described above in which the choice of $E(g')$ for $g' \in G'$ is given by* **C1, C2, C3** *above. Then, M and M' are stuttering trace equivalent.*

Proof. Although the setting is slightly different it can be shown similarly to Theorem 3.11 in [55] that the conditions **C1, C2, C3** guarantee that the models M and M' are stuttering trace equivalent. More precisely, for each path $\pi = g_0 a_0 g_1 a_1 \cdots$ with $g_0 = \iota$ in M there is a stuttering equivalent path $\pi' = g'_0 a'_0 g'_1 a'_1 \cdots$ with $g'_0 = \iota$ in M' and a partition $B_1, \ldots, B_j, ..$ of the states of π and a partition $B'_1, \ldots, B'_j, ..$ of the states of π' satisfying for each $i, j \geq 0$ the following two conditions:

I. if $g_i \xrightarrow{a} g_{i+1}$ is a transition such that $g_i, g_{i+1} \in B_j$, then $a \in Invis$, and if $g'_i \xrightarrow{a'} g'_{i+1}$ is a transition such that $g'_i, g'_{i+1} \in B'_j$, then $a' \in Invis$,

II. if $g_i \xrightarrow{a} g_{i+1}$ is a transition such that $g_i \in B_j$ and $g_{i+1} \in B_{j+1}$, and $g'_{i'} \xrightarrow{a'} g'_{i'+1}$ is a transition such that $g'_{i'} \in B'_j$ and $g'_{i'+1} \in B'_{j+1}$, then $a = a'$.

□

Algorithms generating reduced models in which the choice of $E(g')$ for $g' \in G'$ is given by **C1, C2, C3** can be found in many papers [55, 54, 58, 22, 48, 49].

7 Conclusions and Future Work

Many important properties of multi-agent systems are underpinned by the ability of some agents (or groups of agents) to achieve a given goal. Requirements of this kind can be conveniently specified using logics of strategic ability, such as $\mathbf{ATL}$ and $\mathbf{ATL}^*$. However, their semantics typically use concurrent synchronous models, whereas, for many systems, asynchronous modeling would be more appropriate. In this paper, we propose a general semantics of $\mathbf{ATL}$ and $\mathbf{ATL}^*$ for asynchronous MAS, and consider the model checking problem for a relevant subset of $\mathbf{ATL}^*$ formulae.

Model checking of strategic abilities under imperfect information is a notoriously hard problem, for which attempts at practical algorithms and tools started emerging only recently. We establish the theoretical complexity of the problem, and, more importantly, propose a partial order reduction scheme that can substantially decrease the practical complexity of verification. Interestingly, it turns out that the scheme does *not* work for perfect information strategies. Until now, virtually all the results have suggested that verification of strategic abilities is significantly easier for agents with perfect information. Thus, we identify an aspect of verification that might be in favor of imperfect information strategies in some contexts.

Considering potential practical applications, we can verify correctness of authentication protocols as specified and discussed in [31] as well as of timed authentication protocols [29, 30, 42] after extending our results to Timed ATL [43]. For example we can check the $\mathbf{sATL}$ property expressing that an intruder does not have a strategy to possess an 'insecure' information.

In the future, we plan also to extend our method to a larger subset of $\mathbf{ATL}^*$ specifications. Adapting the POR scheme to combinations of strategic and epistemic modalities is another interesting path for future work. Finally, we would like to investigate whether our partial order reduction scheme can be combined with the bisimulation-based reduction for $\mathbf{ATL}_{\mathrm{ir}}$, proposed very recently in [7].

References

[1] P. A. Abdulla, S. Aronis, B. Jonsson, and K. F. Sagonas. Optimal dynamic partial order reduction. In *The 41st Annual ACM SIGPLAN-SIGACT Symposium on Principles of Programming Languages, POPL '14, San Diego, CA, USA, January 20-21, 2014*, pages 373–384, 2014.

[2] R. Alur and T. A. Henzinger. Reactive modules. *Formal Methods in System Design*, 15(1):7–48, 1999.

[3] R. Alur, T. A. Henzinger, and O. Kupferman. Alternating-time Temporal Logic. In *Proceedings of the 38th Annual Symposium on Foundations of Computer Science (FOCS)*, pages 100–109. IEEE Computer Society Press, 1997.

[4] R. Alur, T. Henzinger, F. Mang, S. Qadeer, S. Rajamani, and S. Tasiran. MOCHA: Modularity in model checking. In *Proceedings of CAV*, volume 1427 of *Lecture Notes in Computer Science*, pages 521–525. Springer, 1998.

[5] R. Alur, L. de Alfaro, R. Grossu, T. Henzinger, M. Kang, C. Kirsch, R. Majumdar, F. Mang, and B.-Y. Wang. jMocha: A model-checking tool that exploits design structure. In *Proceedings of ICSE*, pages 835–836. IEEE Computer Society Press, 2001.

[6] R. Alur, T. A. Henzinger, and O. Kupferman. Alternating-time Temporal Logic. *Journal of the ACM*, 49:672–713, 2002. doi: 10.1145/585265.585270.

[7] F. Belardinelli, R. Condurache, C. Dima, W. Jamroga, and A. Jones. Bisimulations for verification of strategic abilities with application to ThreeBallot voting protocol. In *Proceedings of the 16th International Conference on Autonomous Agents and Multiagent Systems AAMAS 2017*. IFAAMAS, 2017. To appear.

[8] N. Bulling and W. Jamroga. Alternating epistemic mu-calculus. In *Proceedings of IJCAI-11*, pages 109–114, 2011.

[9] N. Bulling and W. Jamroga. Comparing variants of strategic ability: How uncertainty and memory influence general properties of games. *Journal of Autonomous Agents and Multi-Agent Systems*, 28(3):474–518, 2014.

[10] N. Bulling, J. Dix, and W. Jamroga. Model checking logics of strategic ability: Complexity. In M. Dastani, K. Hindriks, and J.-J. Meyer, editors, *Specification and Verification of Multi-Agent Systems*, pages 125–159. Springer, 2010.

[11] S. Busard, C. Pecheur, H. Qu, and F. Raimondi. Improving the model checking of strategies under partial observability and fairness constraints. In *Formal Methods and Software Engineering*, volume 8829 of *Lecture Notes in Computer Science*, pages 27–42. Springer, 2014. ISBN 978-3-319-11736-2. doi: 10.1007/978-3-319-11737-9_3.

[12] S. Busard, C. Pecheur, H. Qu, and F. Raimondi. Reasoning about memoryless strategies under partial observability and unconditional fairness constraints. *Information and Computation*, 242:128–156, 2015. doi: 10.1016/j.ic.2015.03.014.

[13] K. Chatterjee, A. Pavlogiannis, N. Sinha, and K. Vaidya. Data-centric dynamic partial order reduction. *CoRR*, abs/1610.01188, 2016.

[14] T. Chen, V. Forejt, M. Kwiatkowska, D. Parker, and A. Simaitis. PRISM-games: A model checker for stochastic multi-player games. In *Proceedings of TACAS*, volume 7795 of *Lecture Notes in Computer Science*, pages 185–191. Springer, 2013.

[15] E. Clarke and E. Emerson. Design and synthesis of synchronization skeletons using branching time temporal logic. In *Proceedings of Logics of Programs Workshop*, volume 131 of *Lecture Notes in Computer Science*, pages 52–71, 1981.

[16] C. Courcoubetis, M. Vardi, P. Wolper, and M. Yannakakis. Memory-efficient algorithms for the verification of temporal properties. *Formal Methods in System Design*, 1(2/3):275–288, 1992.

[17] M. Dastani and W. Jamroga. Reasoning about strategies of multi-agent programs. In *Proceedings of AAMAS2010*, pages 625–632, 2010.

[18] C. Dima and F. Tiplea. Model-checking ATL under imperfect information and perfect recall semantics is undecidable. *CoRR*, abs/1102.4225, 2011.

[19] C. Dima, B. Maubert, and S. Pinchinat. The expressive power of epistemic μ-calculus. *CoRR*, abs/1407.5166, 2014.

[20] C. Dima, B. Maubert, and S. Pinchinat. Relating paths in transition systems: The fall of the modal mu-calculus. In *Proceedings of MFCS*, volume 9234 of *Lecture Notes in Computer Science*, pages 179–191. Springer, 2015. doi: 10.1007/978-3-662-48057-1_14.

[21] C. Flanagan and P. Godefroid. Dynamic partial-order reduction for model checking software. In *Proceedings of the 32nd ACM SIGPLAN-SIGACT Symposium on Principles of Programming Languages, POPL 2005, Long Beach, California, USA, January 12-14, 2005*, pages 110–121, 2005.

[22] R. Gerth, R. Kuiper, D. Peled, and W. Penczek. A partial order approach to branching time logic model checking. *Information and Computation*, 150:132–152, 1999.

[23] P. Godefroid. Using partial orders to improve automatic verification methods. In E. M. Clarke and R. P. Kurshan, editors, *Proceedings of the 2nd International Conference on Computer Aided Verification (CAV'90)*, volume 3 of *ACM/AMS DIMACS Series*, pages 321–340, 1991.

[24] P. Godefroid and P. Wolper. A partial approach to model checking. *Information and Computation*, 110(2):305–326, 1994.

[25] U. Goltz, R. Kuiper, and W. Penczek. Propositional temporal logics and equivalences. In *Proceedings of the 3rd International Conference on Concurrency Theory (CONCUR'92)*, volume 630 of *LNCS*, pages 222–236. Springer-Verlag, 1992.

[26] D. Guelev, C. Dima, and C. Enea. An alternating-time temporal logic with knowledge, perfect recall and past: axiomatisation and model-checking. *Journal of Applied Non-Classical Logics*, 21(1):93–131, 2011.

[27] W. Hoek and M. Wooldridge. Tractable multiagent planning for epistemic goals. In M. Gini, T. Ishida, C. Castelfranchi, and W. L. Johnson, editors, *Proceedings of the First International Joint Conference on Autonomous Agents and Multiagent Systems (AAMAS'02)*, pages 1167–1174. ACM Press, 2002.

[28] X. Huang and R. van der Meyden. Symbolic model checking epistemic strategy logic. In *Proceedings of AAAI*, pages 1426–1432, 2014.

[29] G. Jakubowska and W. Penczek. Modelling and checking timed authentication of security protocols. *Fundam. Inform.*, 79(3-4):363–378, 2007.

[30] G. Jakubowska and W. Penczek. Is your security protocol on time ? In *Proceedings of International Symposium on Fundamentals of Software Engineering, International Symposium, FSEN*, pages 65–80, 2007.

[31] G. Jakubowska, P. Dembiński, W. Penczek, and M. Szreter. Simulation of security protocols based on scenarios of attacks. *Fundam. Inform.*, 93 (1-3):185–203, 2009.

[32] W. Jamroga. Some remarks on alternating temporal epistemic logic. In B. Dunin-Keplicz and R. Verbrugge, editors, *Proceedings of Formal Approaches to Multi-Agent Systems (FAMAS 2003)*, pages 133–140, 2003.

[33] W. Jamroga and J. Dix. Model checking ATL_{ir} is indeed Δ_2^P-complete. In *Proceedings of EUMAS'06*, volume 223 of *CEUR Workshop Proceedings*. CEUR-WS.org, 2006.

[34] W. Jamroga and W. van der Hoek. Agents that know how to play. *Fundamenta Informaticae*, 63(2–3):185–219, 2004.

[35] W. Jamroga, M. Knapik, and D. Kurpiewski. Fixpoint approximation of strategic abilities under imperfect information. In *Proceedings of the 16th International Conference on Autonomous Agents and Multiagent Systems AAMAS 2017*. IFAAMAS, 2017. To appear.

[36] M. Kacprzak and W. Penczek. Unbounded model checking for alternating-time temporal logic. In *Proceedings of AAMAS-04*, pages

646–653. IEEE Computer Society, 2004. doi: 10.1109/AAMAS.2004.10089.

[37] V. Kahlon, C. Wang, and A. Gupta. Monotonic partial order reduction: An optimal symbolic partial order reduction technique. In *Computer Aided Verification, 21st International Conference, CAV 2009, Grenoble, France, June 26 - July 2, 2009. Proceedings*, pages 398–413, 2009.

[38] I. Kokkarinen, D. Peled, and A. Valmari. Relaxed visibility enhances partial order reductions. In *Proceedings of the 9th International Conference on Computer Aided Verification (CAV'97)*, volume 1254 of *LNCS*, pages 328–340. Springer-Verlag, 1997.

[39] I. Konnov, H. Veith, and J. Widder. SMT and POR beat counter abstraction: Parameterized model checking of threshold-based distributed algorithms. In *Computer Aided Verification - 27th International Conference, CAV 2015, San Francisco, CA, USA, July 18-24, 2015, Proceedings, Part I*, pages 85–102, 2015.

[40] L. M. Kristensen and A. Valmari. Improved question-guided stubborn set methods for state properties. In *Proceedings of the 21st International Conference on Applications and Theory of Petri Nets (ICATPN'00)*, volume 1825 of *LNCS*, pages 282–302. Springer-Verlag, 2000.

[41] O. Kupferman, M. Vardi, and P. Wolper. An automata-theoretic approach to branching-time model checking. *Journal of the ACM*, 47(2):312–360, 2000.

[42] M. Kurkowski and W. Penczek. Verifying security protocols modelled by networks of automata. *Fundamenta Informaticae*, 79(3-4):453–471, 2007.

[43] F. Laroussinie, N. Markey, and G. Oreiby. Model checking timed ATL for durational concurrent game structures. In *FORMATS*, volume 4202 of *LNCS*, pages 245–259. Springer, 2006.

[44] F. Laroussinie, N. Markey, and G. Oreiby. On the expressiveness and complexity of ATL. *Logical Methods in Computer Science*, 4:7, 2008.

[45] A. Lomuscio and F. Raimondi. MCMAS : A model checker for multi-agent systems. In *Proceedings of TACAS*, volume 4314 of *Lecture Notes in Computer Science*, pages 450–454. Springer, 2006.

[46] A. Lomuscio and F. Raimondi. Model checking knowledge, strategies, and games in multi-agent systems. In *Proceedings of AAMAS*, pages 161–168, 2006. doi: 10.1145/1160633.1160660.

[47] A. Lomuscio, H. Qu, and F. Raimondi. MCMAS : A model checker for the verification multi-agent systems. In *Proceedings of CAV*, volume 5643 of *Lecture Notes in Computer Science*, pages 682–688. Springer, 2009.

[48] A. Lomuscio, W. Penczek, and H. Qu. Partial order reductions for model checking temporal epistemic logics over interleaved multi-agent systems. In *9th International Conference on Autonomous Agents and Multiagent Systems (AAMAS 2010), Toronto, Canada, May 10-14, 2010, Volume 1-3*, pages 659–666, 2010.

[49] A. Lomuscio, W. Penczek, and H. Qu. Partial order reductions for model checking temporal-epistemic logics over interleaved multi-agent systems. *Fundam. Inform.*, 101(1-2):71–90, 2010.

[50] A. Lomuscio, H. Qu, and F. Raimondi. MCMAS: An open-source model checker for the verification of multi-agent systems. *International Journal on Software Tools for Technology Transfer*, 2015. doi: 10.1007/s10009-015-0378-x. Availabe online.

[51] A. Mazurkiewicz. Concurrent program schemes and their interpretations. *DAIMI Report Series*, 6(78), 1977.

[52] A. Mazurkiewicz. Trace theory. In *Advances in Petri Nets 1986*, volume 255 of *LNCS*, pages 279–324. Springer-Verlag, 1986.

[53] A. Mazurkiewicz. Basic notions of trace theory. In *Linear Time, Branching Time and Partial Order in Logics and Models for Concurrency*, volume 354 of *LNCS*, pages 285–363. Springer-Verlag, 1988.

[54] D. Peled. All from one, one for all: On model checking using representatives. In *Proceedings of the 5th International Conference on Computer Aided Verification*, LNCS 697, pages 409–423. Springer-Verlag, 1993.

[55] D. Peled. Combining partial order reductions with on-the-fly model-checking. In *Proceedings of the 6th International Conference on Computer Aided Verification*, LNCS 818, pages 377–390. Springer-Verlag, 1994.

[56] D. Peled. Partial order reductions: Model checking using representatives. In *Proceedings of the 21st International Symposium on Mathematical Foundations of Computer Science (MFCS'96)*, volume 1113 of *LNCS*, pages 93–112. Springer-Verlag, 1996.

[57] D. Peled. Ten years of partial-order reductions. In *Proceedings of the 10th International Conference on Computer Aided Verification (CAV'98)*, volume 1427 of *LNCS*, pages 17–28. Springer-Verlag, 1998.

[58] W. Penczek, M. Szreter, R. Gerth, and R. Kuiper. Improving partial order reductions for universal branching time properties. *Fundamenta Informaticae*, 43:245–267, 2000.

[59] J. Pilecki, M. Bednarczyk, and W. Jamroga. Synthesis and verification of uniform strategies for multi-agent systems. In *Proceedings of CLIMA XV*, volume 8624 of *Lecture Notes in Computer Science*, pages 166–182. Springer, 2014.

[60] F. Raimondi. *Model Checking Multi-Agent Systems*. PhD thesis, University College London, 2006.

[61] P. Schnoebelen. The complexity of temporal model checking. In *Advances in Modal Logics, Proceedings of AiML 2002*. World Scientific, 2003.

[62] P. Y. Schobbens. Alternating-time logic with imperfect recall. *Electronic Notes in Theoretical Computer Science*, 85(2):82–93, 2004.

Pracę zgłosił Jan Mielniczuk

Adresy autorów:	Piotr Dembiński, Wojciech Jamroga Antoni Mazurkiewicz, Wojciech Penczek Instytut Podstaw Informatyki PAN ul. Jana Kazimierza 5 01-248 Warszawa, Polska
E-mail:	`piotrd,jamroga@ipipan.waw.pl` `amaz,penczek@ipipan.waw.pl`

Symbol klasyfikacji rzeczowej: D.2.4, F.3.1, F.4.1, H.2.4

Na prawach rękopisu
Printed as manuscript

Nakład 80 egzemplarzy. Papier kserograficzny klasy III. Oddano do druku w grudniu 2016 r. Wydawnictwo IPI PAN. ISSN 0138-0648.